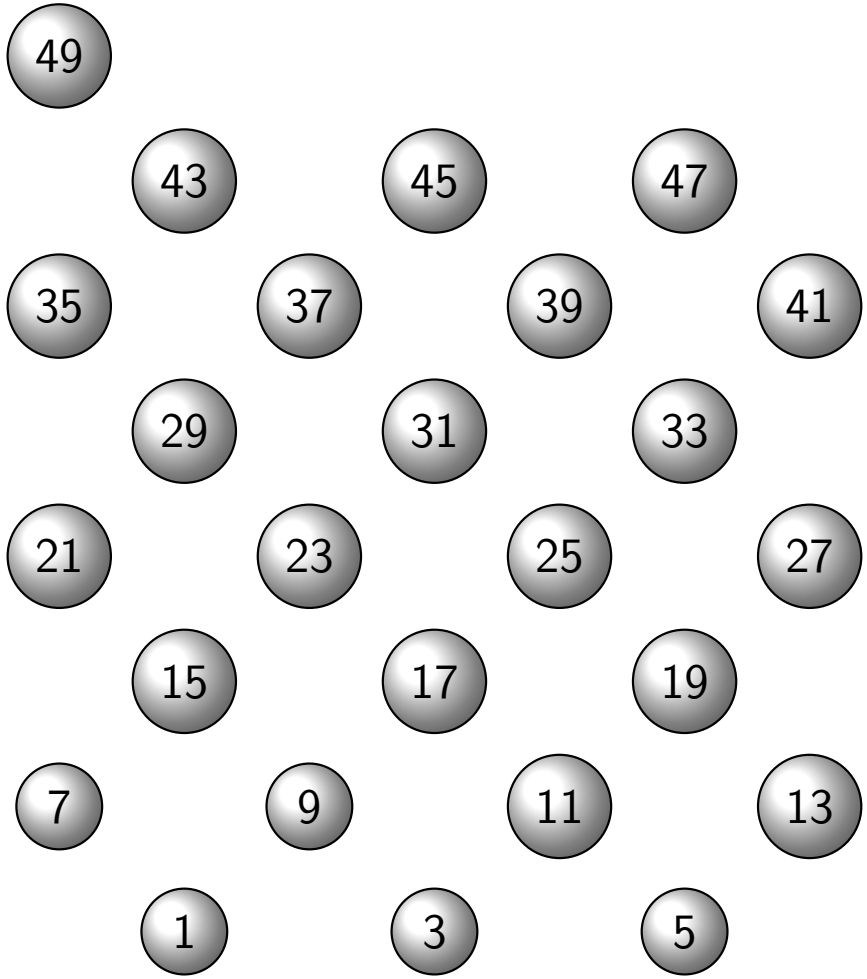
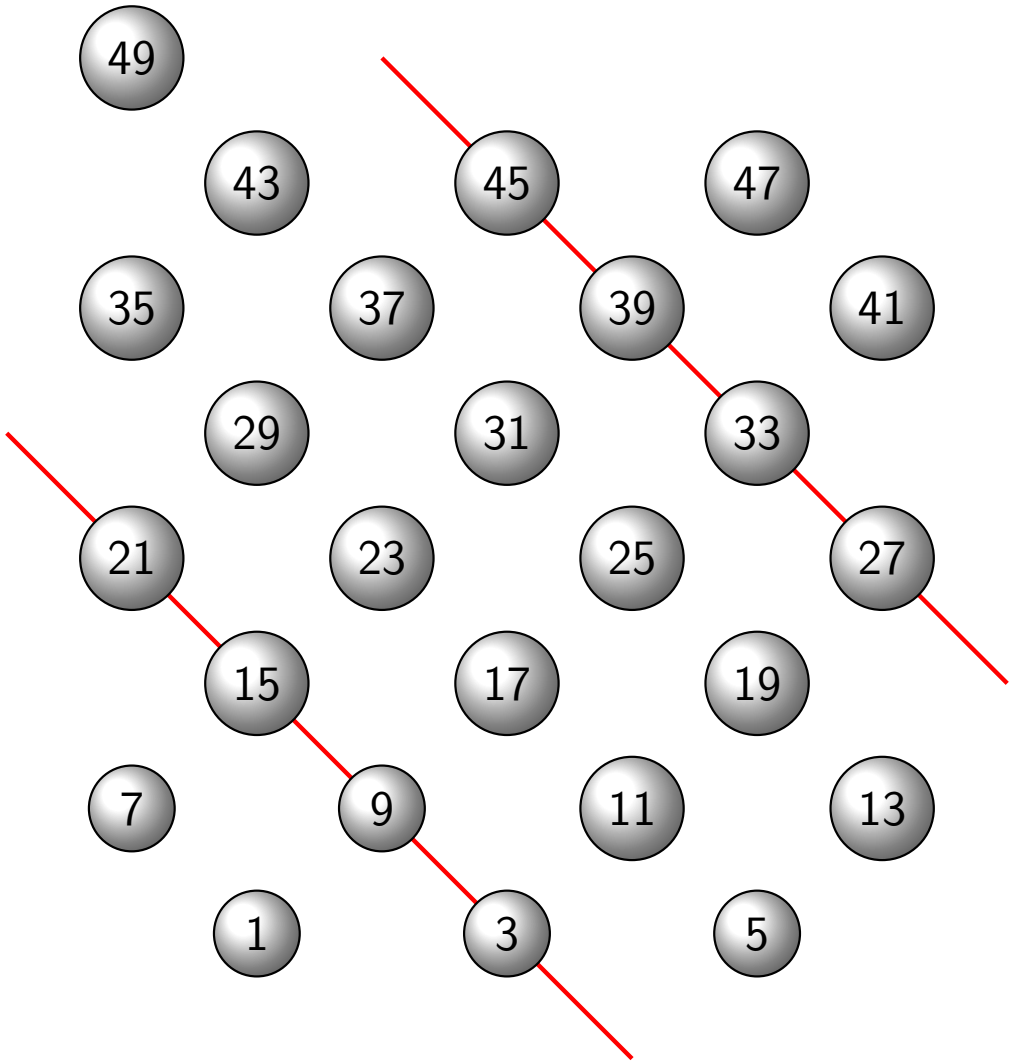
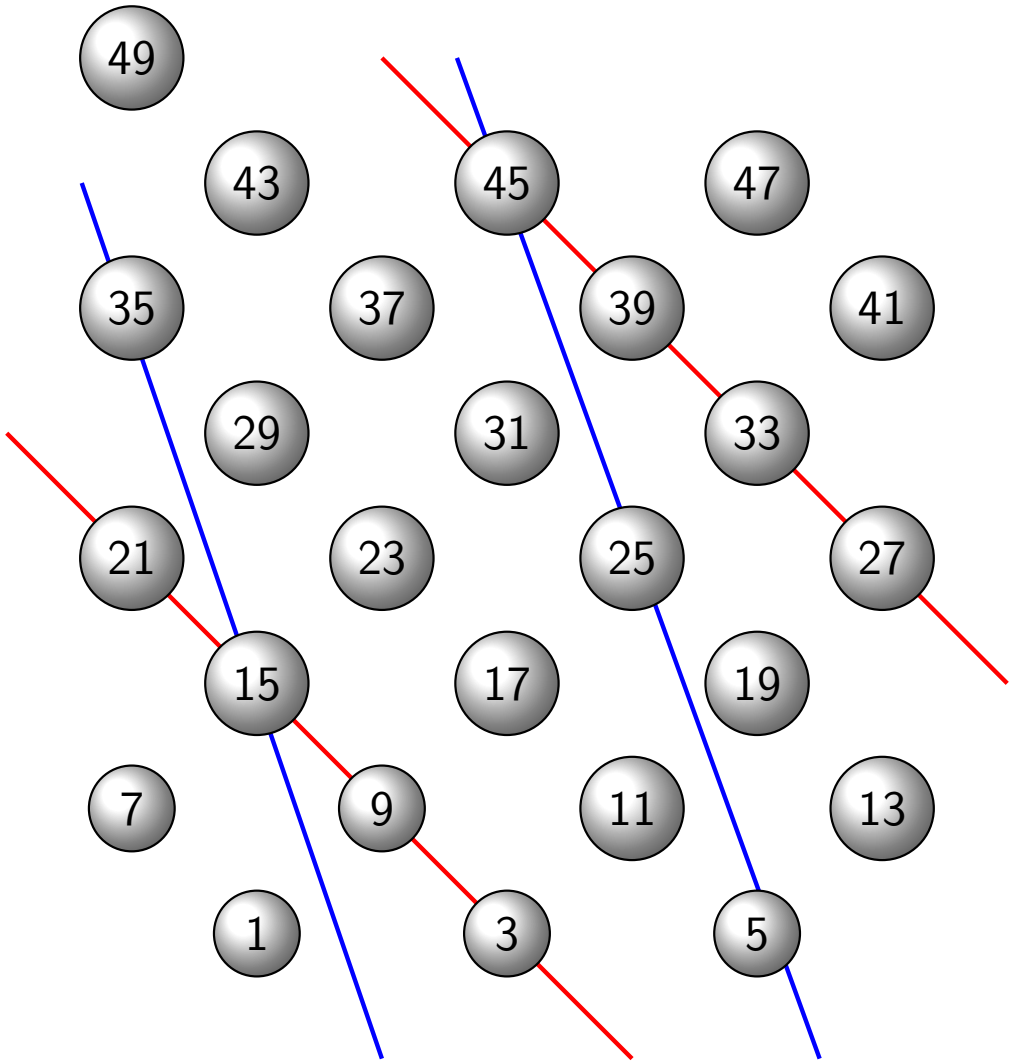


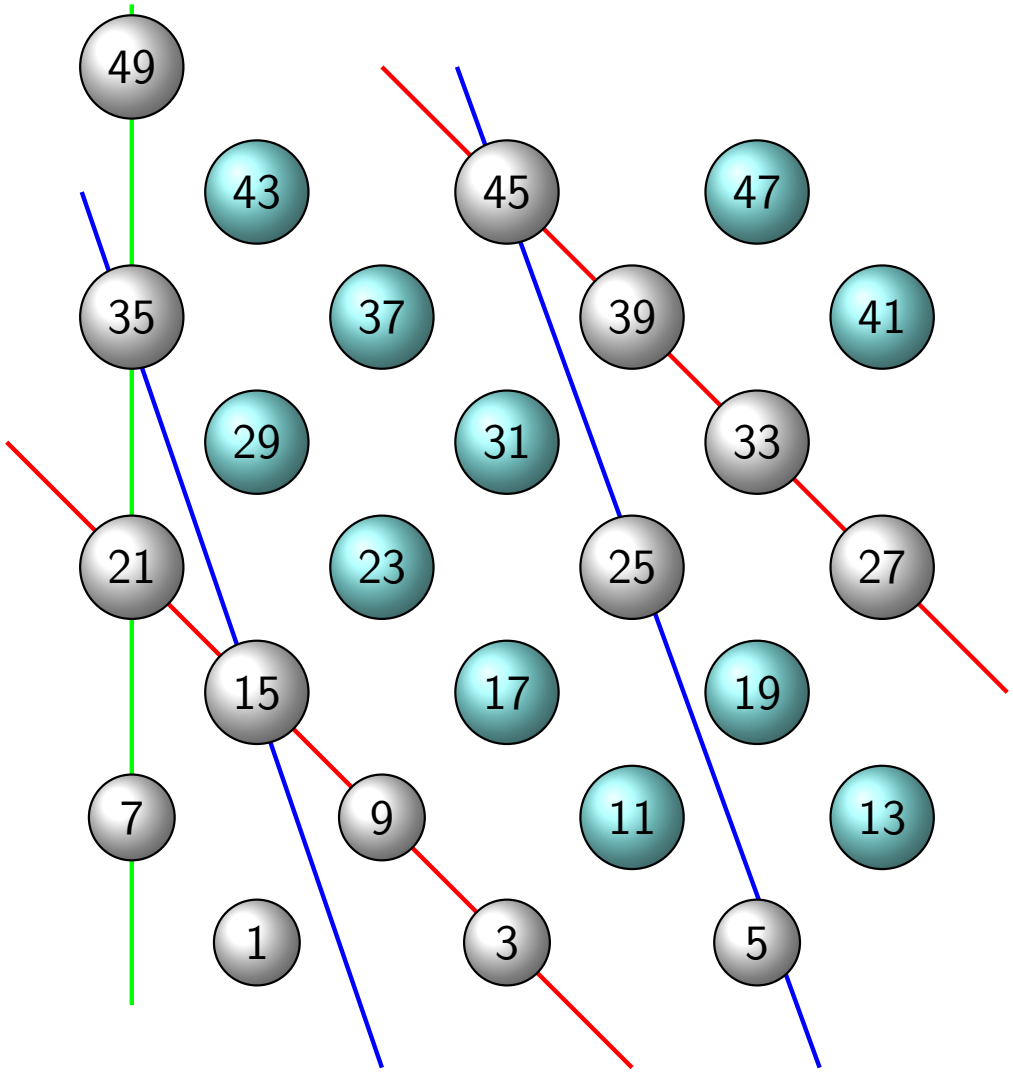
- on cherche les décomposants de Goldbach de 98 ;
- on écrit les nombres impairs de 1 à 49.



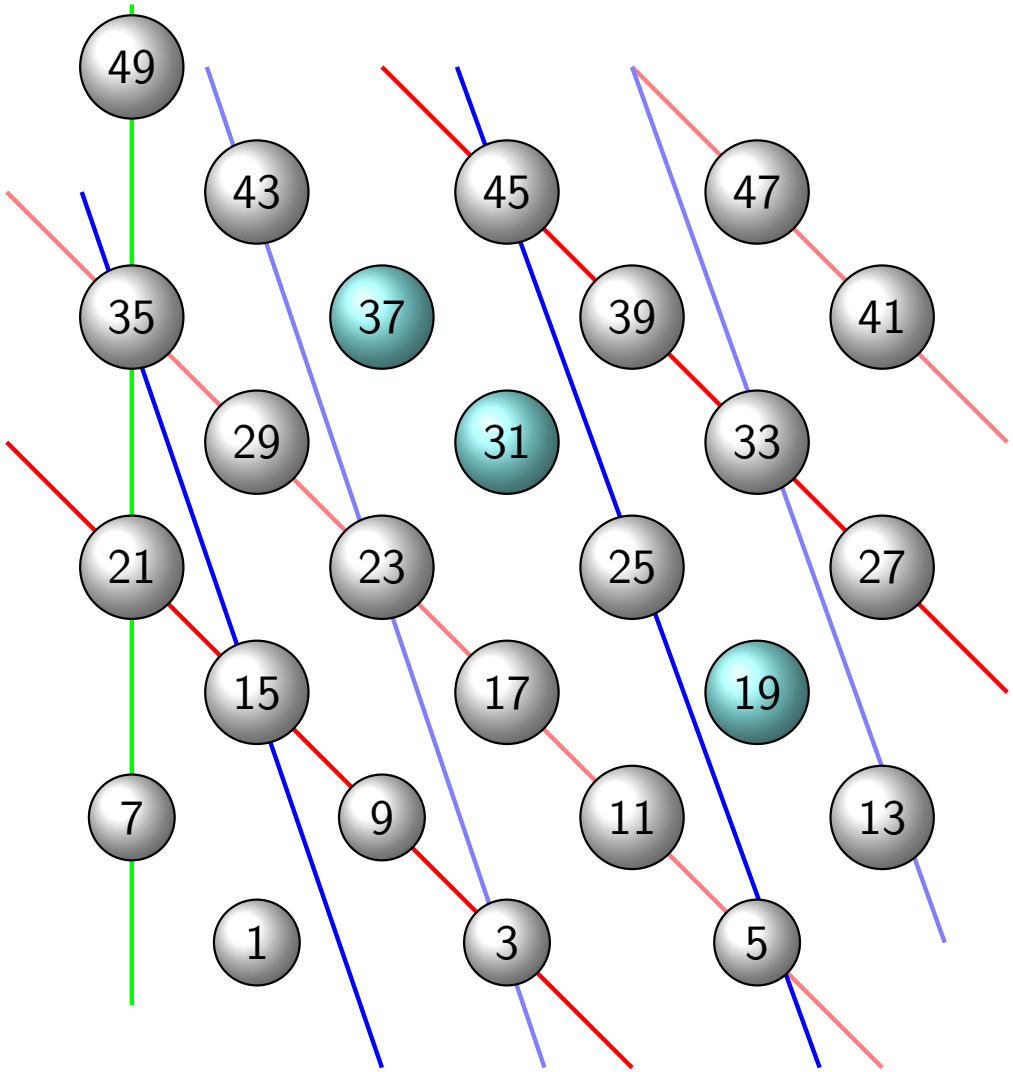
- on crible les multiples de 3 ;



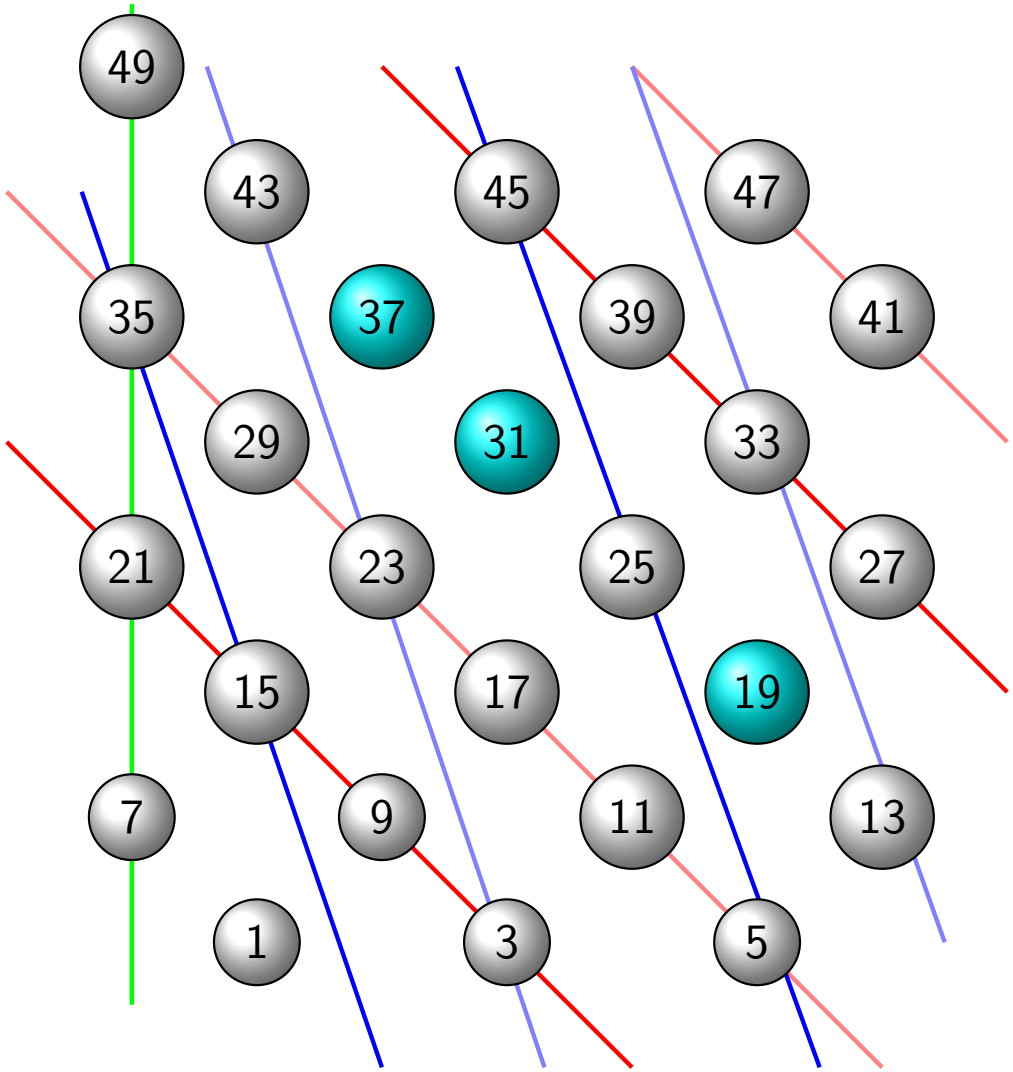
- on crible les multiples de 5 ;



- on crible les multiples de 7 ;
- ne restent (n'appartiennent à aucune droite colorée) que des nombres premiers $> \sqrt{98}$ (et le nombre 1) puisqu'on a criblé tous les multiples de nombres premiers inférieurs à $\sqrt{98}$.

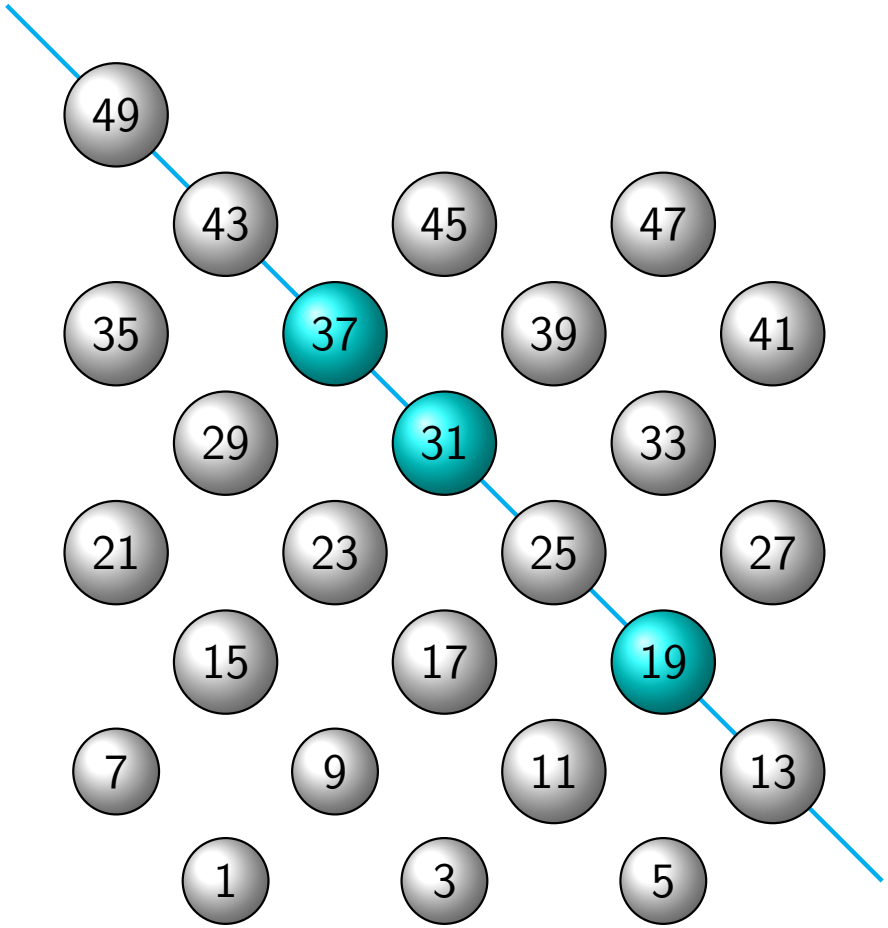


- on crible aussi les $3x + 2$ et les $5x + 3$ car $n = 98$ en est un, pour obtenir un nombre premier dont le complémentaire à n est premier.

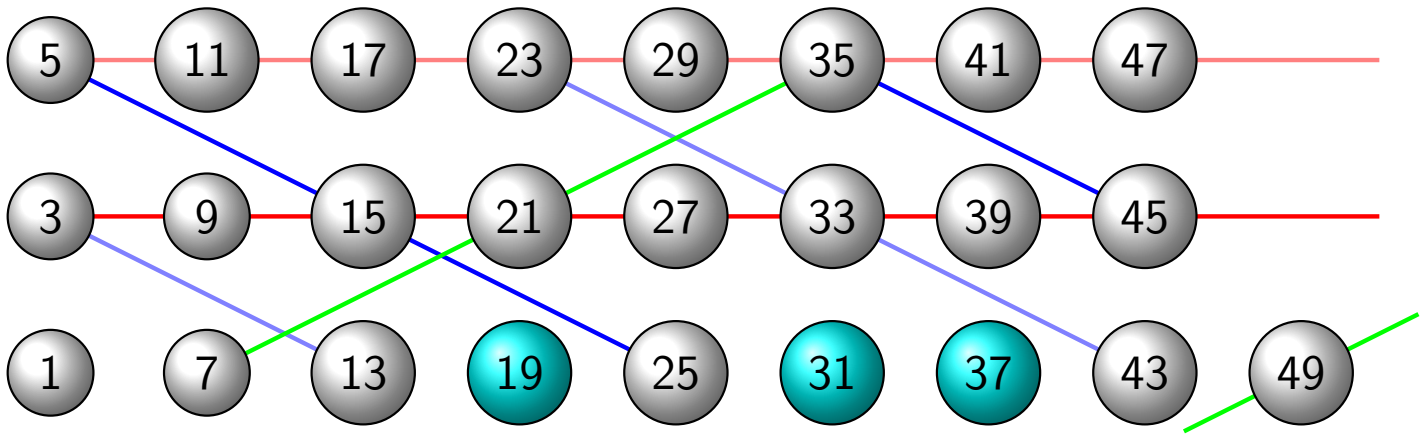


- seuls restent, qui sont passés au travers des mailles du crible, les décomp. de Goldbach de 98 ;

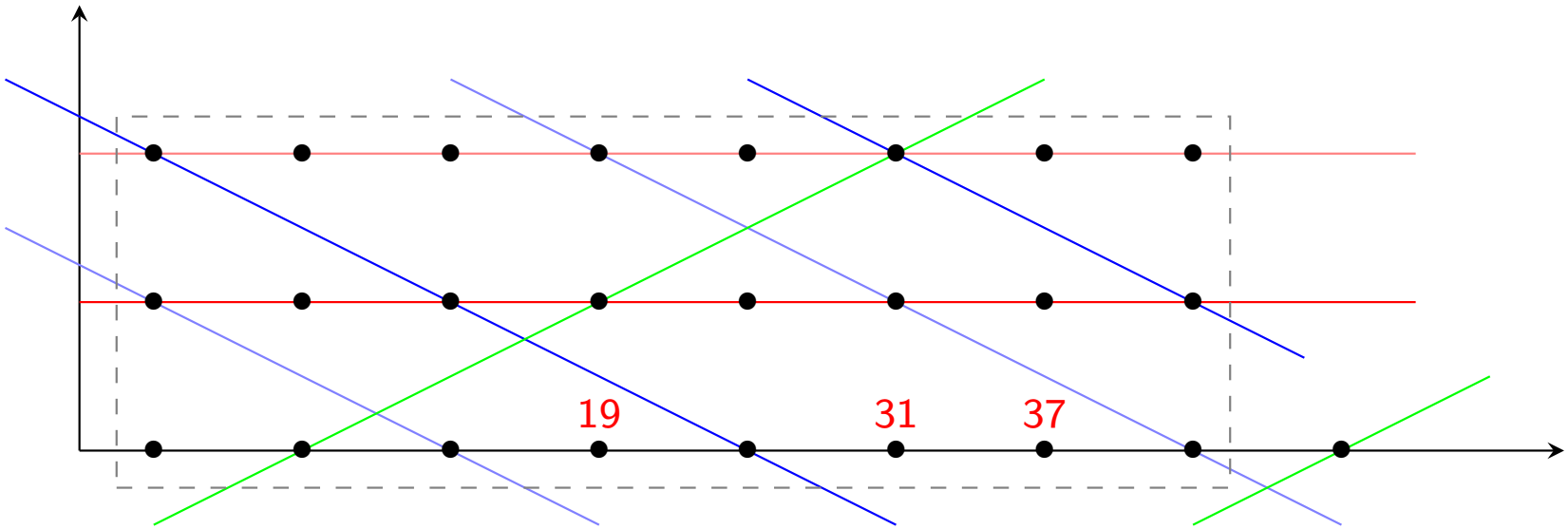
- on crible selon $\pi(\sqrt{n})$ directions ; le nombre de droites parallèles dans une direction donnée est fonction du choix de la base qui affecte les positions des nombres dans le plan cartésien.



- les décomposants de Goldbach de 98 sont tous de la forme $6k+1$ car 98 est un $6k+2$;
- les nombres pairs de la forme $6k$ ont environ deux fois plus de décomposants de Goldbach que les $6k+2$ ou les $6k+4$ car les $6k+2$ n'ont que des décomposants de forme $6k+1$ tandis que les $6k+4$ n'ont que des décomposants de la forme $6k+5$.



- cylindre (infini) plus simple à lire car il contient moins de lignes ;
- principe identique : à la recherche des décomposants de Goldbach de n, selon chaque direction correspondant à un nombre premier p_k inférieur à $\sqrt{n}$, on crible selon 2 restes ($mod\ p_k$) ou bien selon un seul reste si p_k divise n. Les décomposants de Goldbach sont alignés sur l'une et / ou l'autre des deux droites des $6k+1$ et des $6k-1$ selon que le reste de x par 6 est 0, 2 ou 4 ;
- ce que je trouve très chouette : les droites $5x+k$ et les droites $7x+k'$ se trouvent avoir des orientations "symétriques" dans ce cylindre, ainsi par exemple que les droites $11x+k$ et les droites $13x+k'$ ou plus généralement les droites $(6k-1)x+k'$ et $(6k+1)x+k''$;
- pour faciliter la lecture du graphique, on a placé le nombre x à la position $(\lfloor x/6 \rfloor, x\ mod\ 6)$ (la position de x sur le cylindre dépend de son quotient et de son reste par 6).



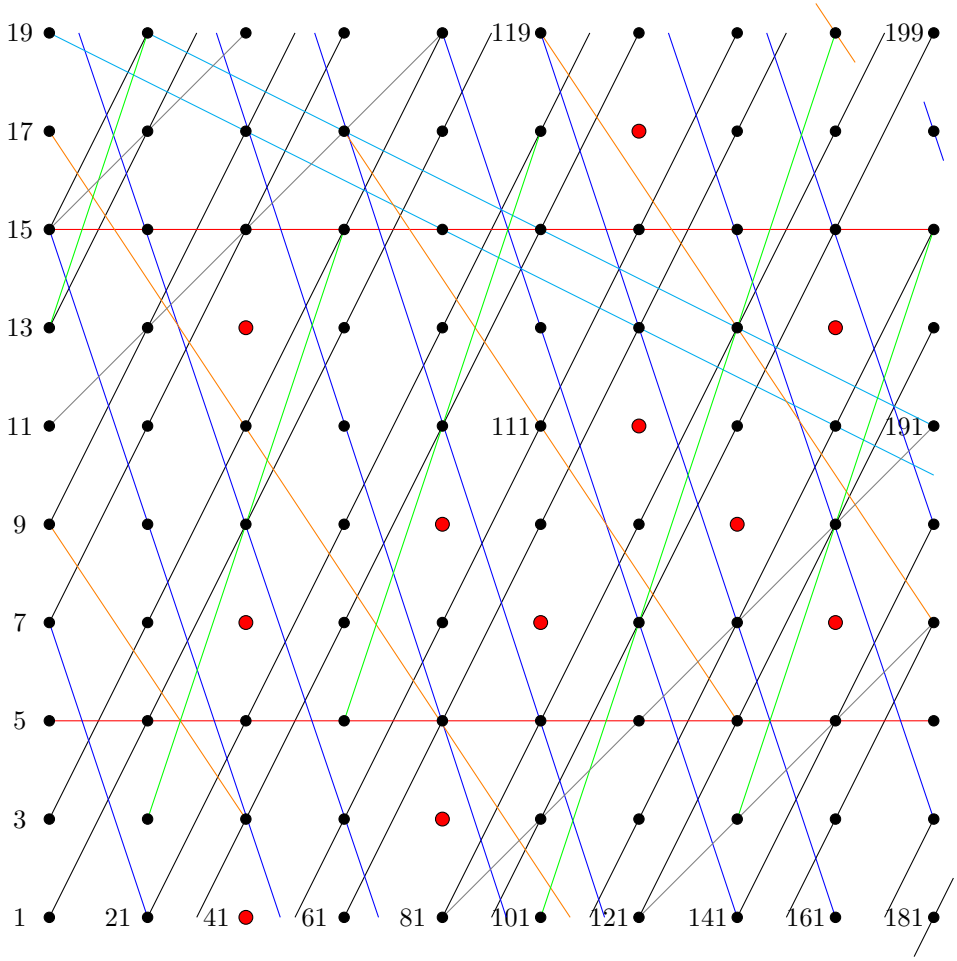
- on a transformé le problème arithmétique en problème géométrique : trouver s'il existe un point n'appartenant à aucune région délimitée par les droites associées à n ;
- problème : on ne sait pas compter les droites, et déterminer les multiplicités (ici par exemple, le point du nombre 35 est à la fois un $5x$, un $7x$ et un $3x+2$ (il est concours d'une droite verte, d'une rose et d'une bleue).

Cristal pour trouver les décomposants de Goldbach de 400 (Denise Vella-Chemla, 7.2.2020)

On trace les droites affines des nombres à cribler pour trouver les décomposants de Goldbach de 400 sur un carré de côté $20 = \sqrt{400}$.

On crible les $3a$, les $5b$, les $7c$, les $11d$, les $13e$, les $17f$ et les $19g$. Les nombres qui passent au travers des mailles du filet sont les nombres premiers supérieurs à 20.

400 appartenant à toutes les suites arithmétiques ci-après, pour que le complémentaire du nombre premier trouvé par le crible ci-dessus soit lui-aussi un nombre premier, on crible également les $3a+1$, les $5b$ (déjà criblés), les $7c+1$, les $11d+4$, les $13e+10$, les $17f+9$ et les $19g+1$.



Les décomposants de Goldbach de 400 (supérieurs à $20 = \sqrt{400}$) sont marqués en rouge (on fournit entre parenthèses leurs coordonnées) : 41 (3,1), 47 (3,4), 53 (3,7), 83 (5,2), 89 (5,5), 107 (6,4), 131 (7,6), 137 (7,9), 149 (8,5), 167 (9,4) et 173 (9,7).

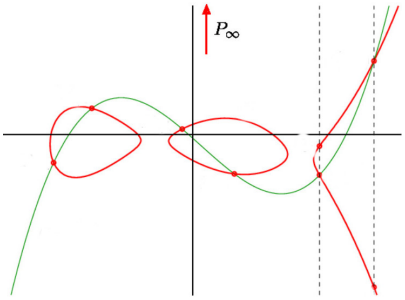
```
vella-chemla@vellachemla-X510UA:~/Desktop/2019/gb-tools-0.7.0$ ./gb-decomp -d 40
0
***** End sieve in 0..400 there are 78 primes (biggest is 397)
              3 +       397 = 400
             11 +       389 = 400
             17 +       383 = 400
             41 +       359 = 400
             47 +       353 = 400
             53 +       347 = 400
             83 +       317 = 400
             89 +       311 = 400
            107 +       293 = 400
            131 +       269 = 400
            137 +       263 = 400
            149 +       251 = 400
            167 +       233 = 400
            173 +       227 = 400
     400 has       14 decomp
```

Ô stop ! (Denise Vella-Chemla, 31.5.2019)

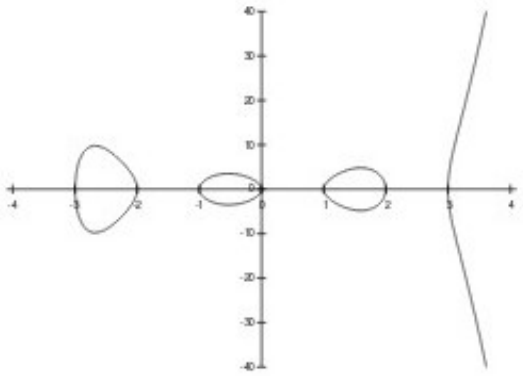
Il s'agit de garder en mémoire le fait qu'en calculant les indices de la section 53 des Recherches arithmétiques de Gauss (consultables ici http://denise.vella.chemla.free.fr/indices-RA53.pdf), on a réalisé à nouveau que la caractéristique des nombres premiers est d'avoir une solution au moins à l'équation $x^{\frac{p-1}{2}} \equiv 1 \ (mod \ p)$ (on peut réécrire cette congruence $x^{\frac{p-1}{2}} - kp - 1 = 0$), alors que cette équation n'a pas de solution pour p composé.

Cela permet d'associer à chaque nombre premier une courbe hyperelliptique de genre $\frac{p-1}{2}$ (ou une surface de Riemann à $\frac{p-1}{2}$ trous), selon les exemples ci-dessous.

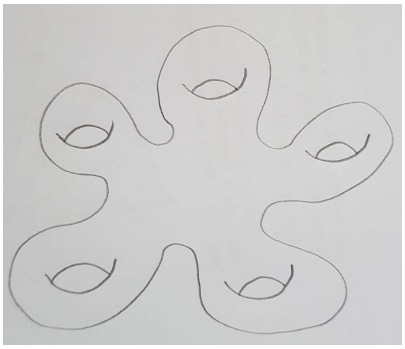
Exemple d'une courbe hyperelliptique de genre 2 associable au nombre premier 5



Exemple d'une courbe hyperelliptique de genre 3 associable au nombre premier 7



Exemple d'une surface de Riemann à 5 trous associable au nombre premier 11



Traduction d'une interview de

Donald E. Knuth

Histoire orale 332

menée par Philip L. Frana
le 8 novembre 2001
à Stanford, Californie

Institut Charles Babbage
Département d'Histoire du traitement de l'information
Université du Minnesota, Minneapolis

Résumé : Donald E. Knuth est professeur émérite d'Art de la programmation à l'Université Stanford. Dans cette histoire orale, Knuth évoque un certain nombre de sujets du développement logiciel qui incluent les brevets logiciels, la programmation structurée et la collaboration. L'histoire orale de Donald Knuth contient des éléments concernant l'écriture de L'art de la programmation ainsi que d'autres sujets comme son éducation et son héritage luthérien.

FRANA : Merci beaucoup Don d'avoir accepté cette interview. Je dois dire qu'après avoir lu toutes ces interviews qui évoquent des sujets dont un informaticien parle rarement, j'étais tenté de simplement vous poser des questions tirées aléatoirement des 328 histoires orales que nous avons dans notre collection.

KNUTH : Bonne idée.

FRANA : Mais je suis venu avec une liste de questions dont je n'avais pas la réponse. J'ai rencontré Edsger Dijkstra au cours de l'été, et vous et lui semblez être, je suppose avec juste une poignée d'autres personnes, parmi les plus anciens meneurs en informatique. Je remarque en particulier que vous étiez tous deux consultants chez Burroughs ?

KNUTH : C'est exact..

FRANA : Ou bien des boursiers Burroughs ? Étiez-vous membre ?

KNUTH : Non, non. Je n'étais qu'un consultant pour Burroughs. L'histoire, c'est que je suis allé à l'université de CalTech - j'ai commencé mes études supérieures en 1960 - et pendant cet été-là, j'ai écrit un compilateur pour Burroughs. Nous pouvons en parler davantage si vous voulez. C'était un compilateur Algol qui était destiné à leur batterie d'ordinateurs, le 205. Et quand j'ai eu terminé cela, j'ai trouvé qu'ils avaient un très bon groupe de personnes dans la section Pasadena de Burroughs. L'équipe s'appelait Division des données Electro. Et j'avais des bourses de la National Science Foundation et de la Woodrow Wilson Foundation. Mais les bourses étaient assujetties à cette restriction que vous ne pouviez rien faire sans être un étudiant diplômé ; tu ne pouvais pas faire de conseil ni avoir aucun autre revenu. Donc j'ai décroché les bourses et travaillé pour Burroughs en tant que consultant. C'est ma

première relation avec l'informatique alors que j'étais étudiant en mathématiques à Stanford. Et à Burroughs, j'ai eu une merveilleuse relation avec les gens et j'ai travaillé avec des groupes sur des logiciels et sur des matériels. Et je pense que je pourrais avoir rencontré Edsger pour la première fois quand il est venu en visite au début des années 60, environ 62 ou 63, peut-être. Et puis je l'ai rencontré plusieurs fois lors de voyages en Europe. Mais il était aussi chez Burroughs, je suppose, mais il avait aussi et surtout une chaire à Eindhoven.

FRANA : Avez-vous écrit une série de livres blancs comme Edsger ?

KNUTH : Non. Il a cette série EWD 1 qui contient 2500 ou je ne sais combien...

FRANA : Une infinité...

KNUTH : Oui. Et je les ai maintenant sur un CD, ce qui est génial.

FRANA : D'accord. Et ils sont désormais également accessibles sur le Web.

KNUTH : D'accord. Mais il ne met pas d'index dans ses livres. Alors, quand certains écrits sur l'informatique sont sortis, on a réimprimé tout un tas de ces EWD, je devais lire tout ça pour savoir ce qu'il disait à propos de moi. Je ne pouvais pas simplement regarder dans l'index et rechercher les pages. Je me demandais quelle était son opinion à mon égard, parce que c'est un critique assez fervent de ce qu'il aime et n'aime pas, et c'est sa grande force. J'ai tendance à être un peu plus insipide à mon avis. *[rires]*. Alors, j'ai rencontré Peter Naur en 1967. Je me souviens précisément que nous ne nous connaissions que par le biais de publications et correspondance pendant qu'il était rédacteur en chef du Algol Bulletin. Et c'est un autre homme de goûts et d'aversions très forts, et d'opinions fortes. Et c'était intéressant parce que je travaillais sur l'*Art de la programmation des ordinateurs*, et je l'ai en quelque sorte, à cette époque, considéré comme un assez bon plan pour un programme d'études en informatique - si l'informatique venait à s'imposer comme discipline. Je pensais que mon livre passait en revue les sujets de base qui devraient être inclus dans les travaux informatiques. Et Peter Naur et moi avions tous les deux été invités à Trondheim pour prendre la parole. L'Université de Trondheim nous a invités un jour alors qu'ils essayaient de

planifier le curriculum. C'était au printemps de 1967, et ils nous avaient invités tous les deux à aller là-haut et à parler pendant une heure de ce que nous pensions être un bon aperçu pour leurs élèves. Et il s'est avéré que Peter et moi avions indépendamment le même schéma. Nous sommes donc devenus amis.

FRANA : Avez-vous inclus un cours sur les structures de données à ce moment-là ?

KNUTH : Oui, oui. Il a publié ces choses dans un petit livret. C'était seulement une esquisse. Il n'a jamais développé des choses aux frontières comme j'ai essayé de le faire. Mais il a structuré le terrain de la même manière et il a eu ce joli mot, qu'il a introduit dans la langue danoise, *datologi*, un concept qui a été appelé "science informatique" par défaut en Amérique.

FRANA : Maintenant, vous opposez-vous à cette expression de "Science informatique", aussi fortement que le fait Edsger ?

KNUTH : J'aime ça, en fait. Une rose est une rose. Les mots prennent leur sens en ce qu'ils signifient, pas vraiment par l'étymologie. Je veux dire, considérez la "mathématique". Ce mot a probablement été très mal choisi, mais maintenant tout le monde sait ce que sont les mathématiques. Donc peu importe le terme que vous employez, dès que les gens commencent à comprendre ce que le terme signifie. Je m'oppose à certains égards à "informatique", parce que je pense que cela met l'accent sur les données sur lesquelles les algorithmes travaillent, plutôt que sur les processus. Il semble que l'information soit plus statique que dynamique. Mais alors quoi ? Je serais également satisfait de ce terme. Et pour la même raison, je ne pensais pas que "datalogie" était le mot optimal. En Suède, ils ont inventé le mot "dator" pour un ordinateur. Et c'était, je pense, un choix brillant. Mais la plupart du temps, les mots ne sont pas optimaux et simplement, nous nous arrêtons d'y penser après un certain temps. Nous ne réalisons même pas ce qu'était la signification originale.

FRANA : Maintenant, le département de l'Université d'Austin au Texas s'appelle Sciences de l'ordinateur, au pluriel.

KNUTH : Oh, d'accord. Le Texas doit être plus grand. *[rires]*

KNUTH : Je pense que le mot apparaît au pluriel à quelques endroits. Au Wisconsin peut-être... Mais il est au singulier ici.

FRANA : Trouvez-vous cela étrange ?

KNUTH : Non. Je pense simplement que c'est une bizarrerie de l'histoire.

FRANA : Trouvez-vous que les gens en Amérique soient a-mathématiques en général ?

KNUTH : Eh bien, il semblerait que les choses se passent certainement comme ça. Vous prenez tout sujet particulier qui vous intéresse et vous essayez de voir si quelqu'un l'a appris dans un lycée américain, et vous serez consterné. Vous savez, Jesse Jackson pense que les étudiants ne savent rien de la science politique, et je suis sûr que les chimistes pensent que les étudiants ne connaissent pas la chimie, etc. Mais d'une manière ou d'une autre, ils l'apprennent quand ils le doivent plus tard. Certainement, je dirais que les étudiants ont maintenant des notions plus superficielles en mathématiques qu'auparavant. Nous devons faire des corrections à Stanford que nous n'avions pas à faire il y a trente ans.

FRANA : Gio [Wiederhold] m'a dit à peu près la même chose.

KNUTH : La chose la plus scandaleuse a été que le cours de Stanford en algèbre linéaire n'a pas pu atteindre la notion de valeurs propres parce que les élèves ne connaissaient pas les nombres complexes. Maintenant, chaque cours à Stanford qui prend l'algèbre linéaire comme condition préalable le fait parce qu'ils veulent que les étudiants connaissent les valeurs propres. Mais ici, à Stanford, avec l'une des normes d'admission les plus élevées de toutes les universités, nos étudiants ne connaissent pas les nombres complexes. Nous devons donc les leur enseigner seulement quand ils arrivent à l'université. Oui, le système est définitivement en panne.

FRANA : Votre formation en mathématiques au lycée était-elle particulièrement bonne, ou est-ce que vous avez passé beaucoup de temps à faire des problèmes ?

KNUTH : Non, ma formation en mathématiques au lycée n'était pas bonne. Mes professeurs ne pouvaient pas répondre à mes questions et j'ai donc décidé d'aller en physique. Je veux dire, j'avais joué avec les mathématiques au lycée. J'ai fait beaucoup de travail comme dessiner des graphiques et tracer des points, et j'ai utilisé π comme la base d'un système de numération, et j'ai exploré à quoi ressemblerait le monde si vous vouliez faire des logarithmes et si vous aviez un système de numération basé sur π. Et j'avais joué avec des trucs comme ça. Mais mes professeurs ne pouvaient pas répondre aux questions que j'avais. J'avais prouvé que 1 = -1 ; et ils ne pouvaient rien voir de mal à cela. Et moi, j'ai pensé, eh bien, si 1 = -1, ce n'est pas si bon. Mais j'ai adoré mon professeur de physique, et j'ai décidé de me spécialiser en physique de ce fait. Il a fallu attendre de passer ma deuxième année au collège pour réaliser que les mathématiques étaient vraiment importantes pour moi. Et c'était en partie par égoïsme. J'ai aimé le fait qu'avec les mathématiques, vous pouvez savoir qu'une réponse est correcte. Avec la physique, vous ne saurez jamais. Vous pourrez passer toute votre vie sans savoir si quelque chose que vous avez fait est bien. Avec les mathématiques, vous pouvez obtenir ce réconfort. Et donc pour le genre de personne que je suis, c'était bien. Dans mon livre sur la science et la religion, je souligne que je ne me préoccupe pas de certains mystères ; cependant, j'aime savoir quelques choses certaines. C'est donc ce qui m'a attiré dans les mathématiques. Je ne pouvais pas imaginer comment un astronome pourrait être content de simplement théoriser sur ce qui se passait sur des étoiles très lointaines. Il ne pourrait jamais y aller, voilà.

FRANA : Avez-vous une réponse ? Les étudiants américains sont-ils différents aujourd'hui ? Dans l'une de vos interviews, vous discutez du problème de la créativité par rapport à l'absorption brute des connaissances.

KNUTH : Eh bien, cela en fait partie. Aujourd'hui, nous avons surtout une culture sonore solide, un certain manque d'attention et on essaye d'apprendre à passer les examens.

FRANA : Oui, je me suis soudain rendu compte que vous aviez écrit plusieurs fois que les mathématiques sont rassurantes. Mais dans votre conférence du prix de Kyoto en 1996, vous dites quelque chose comme, "La mathématique appartient à Dieu.".

KNUTH : Eh bien, c'était dans un contexte différent. C'était dans un

contexte de brevets ou quelque chose comme ça.

FRANA : D'accord, d'accord.

KNUTH : J'ai dit que vous ne revendiqueriez pas la propriété du numéro dix. Mais j'ai en fait réfléchi à de telles choses quand les chiffres grandissent.

FRANA : Quelle est votre opinion sur le brevetage de logiciels ?

KNUTH : Je suis contre les brevets logiciels sur toute idée si évidente que vous vous attendriez à ce qu'un étudiant typique l'invente. On doit pouvoir peut-être me persuader qu'il est normal d'avoir un brevet sur quelque chose de vraiment profond, comme un algorithme de point intérieur ou quelque chose de ce genre. Mais, certainement si je devais écrire le système TEX dans l'environnement d'aujourd'hui, je ne commencerais jamais, car j'aurais trop d'inquiétude sur le fait d'obtenir l'autorisation d'utiliser des centaines d'idées. Bien sûr, heureusement, nous avons intégré la plupart des algorithmes dans les logiciels avant de prendre cette décision stupide du brevetage. Mais je pense que de toute façon, les programmeurs de logiciel devraient être payés pour les services, la personnalisation, l'adaptation des programmes, pour les connaissances qu'ils ont sur la façon de maintenir les choses. Mais pas pour les algorithmes, pas pour les méthodes utilisées. Considérez l'analogie que j'ai faite avec le bureau des brevets. J'ai écrit une lettre ouverte au Commissaire aux brevets. J'ai dit que les algorithmes sont comme les mots pour l'écrivain, ou la jurisprudence pour les avocats. Que deviendrait la loi si à chaque fois que les avocats utilisaient la jurisprudence, ils devaient facturer des frais à d'autres avocats pour avoir cité leur précédent. Ce serait, je pense, très analogue à ce que les gens veulent pour les logiciels.

FRANA : Mais obliger un historien à payer un centime quand il cite le travail de quelqu'un d'autre dans une note de bas de page...

KNUTH : Ou si vous ne le citez pas, cela vaut un centime.

FRANA : Avez-vous un peu suivi le mouvement du développement de logiciels à l'étranger ?

KNUTH : Non, je reviens tout juste de Munich et la Commission européenne

analyse en ce moment toute la question des brevets logiciels. Donc ils m'ont demandé ce que j'en pensais. Et j'ai dit : "Eh bien, j'espère que vous n'allez pas y aller." J'ai été satisfait de ce que j'ai lu sur la Grande-Bretagne. Je pense que leur attitude est assez saine. Mais je suis la question de près. Concernant cela, je ne suis pas un croisé comme Richard Stallman. J'ai découvert que mes talents résident dans l'écriture de livres sur la programmation. Je peux parler aux gens lors de cocktails et dire : "Oh, oui, j'aime ceci ou je n'aime pas cela, mais je ne sors pas et ne fais pas de discours à ce sujet et tout ç. Stallman m'a encouragé à écrire une lettre ouverte à la Commission des brevets, une fois après une longue conversation téléphonique. Je cédai à contre-cœur. Il pensait que cela ferait un peu de bien. Je doute que ce soit le cas, mais de toute façon, ce qui est fait est fait.

FRANA : Maintenant au-delà de la formation brute, plusieurs informaticiens dont j'ai parlé ont décrit des expériences mystiques. Y-a-t-il une espèce de chance pour l'esprit préparé ? Est-ce vraiment de cela qu'il s'agit ?

KNUTH : Eh bien, ce serait probablement la meilleure explication d'un point de vue rationnel. Je veux dire Poincaré parle de ces choses, et bien d'autres personnes le font également, comme Hadamard par exemple. Et vous obtenez ce moment Eureka. En quelque sorte, tout d'un coup, vous savez que quelque chose fonctionne. Et c'est généralement le cas.

FRANA : La raison pour laquelle je demande cela, c'est que nous sommes assis dans cette salle ; évidemment, en audio, je ne peux pas décrire complètement la pièce, mais il y a un tapis à poils longs avec des motifs sur le sol, et vous avez des photos sur le mur.

KNUTH : C'est une sorte de pièce psychédélique.

FRANA : Oui, c'est multicolore. Il y a de très nombreuses ampoules, seize environ, rien qu'au plafond. Est-ce que cela vous inspire ?

KNUTH : Non, non. Nous n'avons pas beaucoup utilisé cette pièce en fait, mais nous voulions faire quelque chose de différent. Nous aimions l'idée d'avoir différents types d'espaces dans notre maison, pas seulement des chambres beiges. Et donc quand nous avons vu ces grands tapis en Norvège, nous sommes tombés amoureux d'eux et nous avons pensé : "Comment pouvons-

nous les utiliser à la maison ? “ Nous avons également expérimenté les panneaux de cuivre ici et des choses comme ça. Mais cette salle n'a jamais répondu à ces attentes. Les lumières allumées en question, on peut les rendre plus sombres ou plus lumineuses.

FRANA : Je pensais que c'était peut-être une salle d'idées.

KNUTH : Oui, eh bien, ce serait bien d'avoir une salle d'idées. Mais hier soir, j'ai eu une idée dans une autre pièce - pendant que je mettais la table de la salle à manger. C'était une idée stupide, mais je pourrais aussi bien vous la dire. Dans la partie de mon livre que j'écris en ce moment, l'un des exemples que j'utilise pour les algorithmes est un puzzle appelé “cryptarythme” où vous avez des mots, puis vous les ajoutez ensemble, puis vous substituez des chiffres pour les lettres, et ça marche. Alors que je mettais la table hier soir, je me suis dit que je devrais essayer “KNIFE+FORK+SPOON=DINNER” [1]. J'ai réalisé en un éclair que ces les mots ne comportent que dix lettres, parce que, vous savez, la lettre “K” est utilisée dans les deux “KNIFE” et “FORK” et ainsi de suite. Vous comptez donc et il n'y a que neuf lettres différentes entre “KNIFE” et “FORK” et “SPOON”, laissant de la place pour une lettre supplémentaire. Mais de toute façon, alors que je mettais les couverts sur la table, vous savez, je pensais aux cryptarythmes. Et donc je monte vers mon ordinateur et j'essaye. Malheureusement,“KNIFE + FORK + SPOON = SUPPER” n'a pas de solution. J'ai donc ajouté “SOUP”. Et alors, il y a exactement une solution ; c'est parfait ! Une autre variante,“KNIFE + FORK + SPOON = DINNER” avait deux solutions donc ce n'était pas si bon. J'ai pensé à ces cryptarythmes parce qu'ils enseignent des leçons intéressantes lorsque vous essayez différents algorithmes. Vous pouvez trouver une solution de la manière “brutale” où vous passez par toutes les permutations possibles des lettres par rapport aux chiffres aussi vite que vous le pouvez. Il y a trois millions et demi de permutations, mais si vous ne dépensez que cinq instructions machine sur chacune, il faut beaucoup moins qu'une seconde pour les essayer toutes. Sinon, vous pouvez être plus intelligent, et vous pouvez commencer à travailler de droite à gauche. Vous savez, d'abord les chiffres des unités, que vous devez vérifier, puis vous déterminez s'il y a une retenue, puis si le chiffre des dizaines va bien. Et en travaillant de droite à gauche, vous n'avez pas à passer par les 3,5 millions de permutations, vous n'aurez

1. COUTEAU + FOURCHETTE + CUILLÈRE = SOUPER.

peut-être qu'à passer par quelques milliers. Hier, j'avais joué avec d'autres idées ; dans la journée, j'ai écrit un programme pour une méthode de gauche à droite. Quelques choix à gauche sont impossibles : vous savez, vous ne pouvez pas faire de cette lettre un neuf parce que ce serait trop grand. C'était donc dans ma tête hier, ces cryptarythmes. J'en ai inventé quelques-uns qui sont vraiment amusants et qui seront dans mon prochain livre, vous pouvez le vérifier.

FRANA : D'accord. Ce type de résolution d'énigmes remonte à votre enfance. Vous êtes resté à la maison pendant quelques semaines au lieu d'aller à l'école. Il y avait un concours, non ?

KNUTH : Oui, quand j'étais en huitième année. Vous avez l'air d'avoir fait beaucoup de recherches sur moi. C'est impressionnant ! Je suis resté à la maison à l'époque parce qu'il y avait un concours parrainé par la Société Zeigler et ils avaient demandé : "Combien de mots pouvez-vous faire à partir des lettres du nom Zeigler Bar Geant ? J'ai réalisé que si je prenais un dictionnaire non abrégé, il y avait un assez bon moyen de le parcourir, sachant que j'allais seulement utiliser au plus trois mots, et des choses comme ça. Mais je n'aurais jamais à regarder la plupart des parties du dictionnaire. Par exemple, les mots n'avaient pas de C, donc je n'avais pas besoin de chercher dans les C et ainsi de suite. C'était un projet sur deux semaines. J'ai parcouru toutes les parties pertinentes du dictionnaire et j'ai obtenu une liste complète de mots. J'avais plus de deux fois plus de mots que les juges n'en avaient sur leur liste principale. Et j'ai oublié d'utiliser l'apostrophe.

FRANA : Avec l'apostrophe, vous en auriez trouvé encore plus.

KNUTH : Oui, à droite, pour des mots comme...

FRANA : Je veux revenir en arrière et vous poser des questions sur la programmation des langues. Vous avez mentionné que vous avez travaillé sur les compilateurs Algol pour Burroughs ?

KNUTH : Les compilateurs Algol, oui, c'est vrai.

FRANA : Pourquoi Algol 60 n'a-t-il jamais marché ?

KNUTH : Eh bien, FORTRAN était très fort. Les gens pensaient qu'ils devaient faire un gros investissement dans ce langage et ils ne se rendaient pas compte qu'ils allaient devoir changer leurs programmes encore et encore tout le temps. Mais ce n'était qu'une des raisons. La raison la plus importante était probablement que FORTRAN était plus achevé. Algol manquait d'un bon moyen d'obtenir une sortie agréable et lisible. Il n'a jamais fait partie de la norme. Il n'a jamais été considéré comme permettant une vraie programmation pour traiter les problèmes d'entrées / sorties. Et FORTRAN proposait quelque chose qui fonctionnait. Ce n'était pas élégant, mais ça existait. Et comme Algol n'a pas bien répondu à cela, je pense que ça a été l'un des principaux obstacles à l'époque. Donc Algol n'a jamais eu assez d'élan pour le porter.

FRANA : Algol 68 a-t-il insisté là-dessus ?

KNUTH : Algol 68 n'a pratiquement jamais été implémenté dans un bon compilateur. C'était un langage énorme. C'était trop compliqué d'y consacrer beaucoup de temps.

FRANA : Y avait-il un problème tel que "Trop de cuisiniers goûtent le bouillon ?"

KNUTH : Ils avaient un grand comité ; mais c'était vraiment Adriaan van Wijngaarden qui a géré les coups de feu sur Algol 68, ce n'était donc pas la raison. C'était un projet trop ambitieux, tout comme PL / I. Juste faire face à 300 pages de description du langage, je veux dire, vous ne pouvez pas le faire. Donc, Niklaus Wirth est venu avec Pascal, qui était alors très attrayant pour les gens qui voulaient un langage propre et qu'ils pourraient comprendre.

FRANA : J'entends beaucoup de gens se plaindre de la programmation structurée, et pourtant, beaucoup de gens pensent que c'est un sujet important.

KNUTH : La programmation structurée est importante, bien sûr. J'ai écrit un long paragraphe sur la programmation structurée. Ça s'appelait, "Programmation structurée avec Go To Statements", et dans cet article j'avais en quelque sorte cinquante co-auteurs, parce que j'ai incorporé un grand nombre de commentaires de personnes partout dans le monde à qui j'avais distribué le document avant sa publication. Et j'ai essayé de montrer les points forts

de la programmation structurée ainsi que ses points faibles. Le débat portait sur la question de la forme par rapport au fond. Beaucoup de gens pensaient que la programmation structurée signifiait simplement que vous avez limité votre programmation à ne pas utiliser certaines fonctionnalités. Et je détestais cela, juste la façon dont Hilbert détestait les mathématiques intuitionnistes, qui dit que vous ne pouvez pas utiliser le tiers exclus lorsque vous faites une preuve mathématique. Eh bien, je ne voulais pas être menotté avec des interdictions de toutes ces fonctionnalités dont personne n'avait jamais pu abuser, parce que l'idée circulait parmi les gens que telle ou telle instruction est à "considérer comme nuisible". Vous savez, Edsger a dit que le Go To doit être considéré comme nuisible ; et il est nocif, si vous en abusez. Mais j'ai fait un catalogue de toutes les erreurs que j'ai trouvées dans TEX, et 3 ou 4 % des erreurs étaient dues à une mauvaise utilisation d'instructions Go To, et 3 ou 4% des erreurs étaient dues à une mauvaise utilisation des variables globales, et 2-3-4 pour cent des autres étaient dus à une mauvaise utilisation des déclarations if / then, etc. Donc, si vous abolissez tout ce que je n'ai pas utilisé correctement à un moment donné du programme, vous devez retirer toutes les fonctionnalités du langage. Mon point de vue était que structurer la programmation était une merveilleuse révolution, car cela nous a appris à penser à un programme comme ayant une certaine structure et nous a donné quelques moyens intellectuellement gérables pour faire face à la complexité. Mais les gens attaquaient le style uniquement.

Frana : Vous appeliez donc à l'élégance.

Knuth : J'appelais à la compréhension et à la possibilité d'avoir un moyen de comprendre l'objectif, que les abolisseurs de Go To avaient tendance à oublier. Mais en réalité, une instruction Go To est analogue à une instruction d'affectation. Une affectation est une instruction qui donne à une variable une nouvelle valeur et une instruction Go To est une instruction qui donne au programme de contrôle une nouvelle valeur. Et ces deux concepts sont non triviaux et doivent être compris. Je pense que c'est Menabrea qui a demandé à Charles Babbage "Et si vous voulez aller à cette partie de votre programme ?" et Babbage n'avait pas la moindre idée de ce dont parlait Menabrea. C'est donc un concept non trivial pour attribuer une nouvelle valeur au contrôle. Cela signifie logiquement que vous avez une certaine abstraction, qui correspond à ce que signifie être à ce point dans le contrôle. Si vous n'avez pas une bonne signification abstraite pour cela, alors vos instructions

Go To sont très susceptibles de vous causer des ennuis parce que vous allez peut-être dans des endroits où les abstractions entrent en collision les unes avec les autres. C'est pourquoi il était très dangereux de les utiliser sans discipline. Mais je trouve encore que dans certaines circonstances, lorsqu'il est utilisé correctement, un Go To est mieux qu'un simple truc. Tout simplement parce que lorsque quelqu'un considère quelque chose comme un péché mortel, cela ne me convainc pas. C'est comme le mouvement de croissance démographique zéro : l'objectif n'est pas vraiment d'avoir une croissance démographique nulle ; l'objectif est d'améliorer la qualité de la vie des personnes. Mais ils substituent un objectif quantitatif à l'objectif qualitatif, et c'est comme voir la programmation structurée uniquement comme un type et non comme un programme, ce qui est stupide, et je suis sûr que Edsger ne voulait pas le dire de cette façon du tout. Je dois cependant vous raconter une petite blague. En fait, j'ai donné une conférence à Eindhoven au début des années 70, et j'écrivais un algorithme au tableau ; c'était un algorithme pour la correspondance de motifs dans les chaînes de caractères. Quand je suis arrivé vers la fin, j'ai dit : "Oh, Edsger, est-il permis de dire "Allez dans cette pièce ?"" Il a dit : "Je l'ai vu venir." *[rires]*

Frana : J'ai négligé de vérifier cela avant de venir ici, mais étiez-vous aux conférences de Garmisch ou de Rome sur le génie logiciel ?

Knuth : Non.

Frana : Je n'ai pas encore vérifié l'index pour voir.

Knuth : Non, j'ai arrêté d'aller aux conférences. C'était trop décourageant. La programmation informatique devient de plus en plus difficile car plus de choses sont découvertes. Je peux faire face à l'apprentissage d'une nouvelle technique par jour, mais je ne peux pas en apprendre dix à la fois en une journée. Les conférences sont donc déprimantes ; cela signifie que j'ai tellement beaucoup plus de travail à faire. Si je me cache la vérité, je suis beaucoup plus heureux.

Frana : Je suis tout à fait d'accord avec vous. Souvent lorsque je vais à une conférence, je vois une présentation et puis je pars travailler par moi-même.

Knuth : Eh bien, pour chaque journée passée à une conférence, j'ai besoin

de cinq ou six jours pour rattraper après. Bien sûr, c'est agréable de revoir de vieux amis, mais j'ai été là et j'ai fait cela au milieu des années 70.

FRANA : La notion d'ingénierie logicielle, était-ce une expression malheureuse ?

KNUTH : Beaucoup de gens aiment certainement cette expression et la trouvent inspirante. Je ne sais pas. Je n'ai jamais été enthousiasmé par l'idée de métriques logicielles, qui était l'un des piliers du génie logiciel. Parce que les gens pensaient que c'est d'une discipline d'ingénierie dont nous avons besoin pour mesurer les choses. Tu penses, "eh bien, tout ce qui fonctionne pour une branche de l'ingénierie va être utile pour les autres" et tout ce que nous devons faire, c'est de nous ressaisir et d'être plus disciplinés à ce sujet et puis les problèmes seront plus faciles. Mais je pense qu'il n'y a pas de voie royale vers la programmation. C'est l'une des plus grandes perspectives d'Edsger que de comparer la quantité d'effort intellectuel qui va dans les programmes informatiques, par rapport à d'autres types d'activités humaines. En physique, les choses échouent, mais elles échouent lentement ; elles échouent en douceur. Mais avec la logique, il n'y a pas de continuité ; ça va juste mal et tout est en désordre. C'est tout autre chose. Il y a beaucoup de grandes réalisations - vous savez, construire un énorme avion est une réalisation époustouflante, faire un film comme Blanche-Neige et les Sept Nains est une réalisation extraordinaire, toutes sortes de choses - envoyer des gens sur la lune - nécessitent une coopération massive de nombreuses personnes, etc. Pourtant, ces réalisations sont différentes de ce qui se passe lorsque vous avez dix millions de lignes de programme. Et donc le génie logiciel espère changer cela en faisant ces lignes de code beaucoup plus comme la physique, de sorte qu'elles échouent en douceur ou quelque chose comme ça et il y a toutes sortes de redondance. En fait, ça aide un peu, mais cela n'atteint pas l'objectif.

FRANA : Vous vous retrouvez avec un ensemble de problèmes différents ?

KNUTH : Vous repoussez le problème, oui, vous le repoussez simplement sur les genoux de quelqu'un d'autre. Ça ne vous donne pas le... C'est très tentant de penser que si nous enseignons à tout le monde comment utiliser correctement les déclarations, on sera en mesure de proposer des programmes qui font des choses subtiles. Eh bien non. Il faut réfléchir dur et il y a quelques

lignes - il y a une déclaration - si je devais facturer le programme TEX, il y a une déclaration qui est passée par douze révisions, et j'ai dû y réfléchir très dur un certain temps. Pour cette déclaration, je dirais : "D'accord, payez-moi pour celle-là." Mais je ne connais aucune technique de génie logiciel qui m'aurait amené à ce niveau sans une certaine angoisse. Voilà ce qui se cache dans mes commentaires. Il n'y a pas de boules magiques.

FRANA : Cela ressemble beaucoup à quelque chose que Duane Whitlow m'a dit à propos de SyncSort. Il y a cette seule ligne, dit-il, c'est vraiment la ligne qui m'a fait me poser le plus de questions, et la ligne sur laquelle j'ai dû travailler le plus.

KNUTH : Vous l'avez interviewé alors ?

FRANA : Il était en fait un des participants du Xerox PARC l'année dernière, cette conférence durant laquelle on a discuté de l'histoire de l'industrie des produits logiciels. Il a donné un papier.

KNUTH : Oh, d'accord. C'était donc l'un des brevets ? Oui, j'ai finalement découvert...

FRANA : Étiez-vous dans le public là-bas ?

KNUTH : Non, non, mais j'ai eu à le joindre. Je veux dire que j'essayais de découvrir ce que le L. représentait dans Duane L. Whitlow, et je suppose que c'est Leroy, de toute façon, puis j'ai voulu savoir s'il fallait mettre le R en majuscule. J'ai finalement pris contact avec lui, mais je ne l'ai jamais rencontré, non. Mais j'ai aimé l'idée de SyncSort, dont j'ai parlé dans le volume 3 il y a quelques années.

FRANA : Que pensez-vous de la "physique des logiciels" ?

KNUTH : Je ne sais pas ce que c'est.

FRANA : C'est une idée de Ken Kolence.

KNUTH : Je n'en sais rien. Mais je pense qu'il cherche un balle magique.

FRANA : Ça n'existe tout simplement pas ?

KNUTH : Non, ça pourrait exister, mais ça n'existe pas dans mon univers. Je sais ce que je suis bon dans mon domaine étroit. Je ne dis pas que tous dans le monde devraient être comme moi. Je sais seulement que tout ce qui procède comme je le fais est en bonne corrélation avec le fait de faire faire de bonnes choses aux ordinateurs. Mais il y a beaucoup des gens qui ne me ressemblent pas et qui n'aiment pas les analogies que j'aime. Et si j'essaie de leur expliquer quelque chose, ils diront : "Oh, c'est beaucoup plus facile si vous comprenez et faites cela en fonction de la loi de la thermodynamique ou quelque chose du même style. D'accord, mais cela entre par une oreille et sort par l'autre pour moi. Donc quand vous dites "physique du logiciel", ça me semble provenir de ce type qui a cette autre façon de voir le monde. Sa voie pourrait être meilleure que la mienne, mais ce n'est pas ma manière de voir, et je doute que je puisse y contribuer d'une manière ou d'une autre.

FRANA : Pour changer un peu de direction, et pardonnez-moi si je me trompe, mais j'ai lu ou entendu qu'à Stanford, pendant la guerre du Vietnam, vous avez rejoint les manifestants. Est-ce correct ?

KNUTH : Eh bien, un jour, Bob Floyd et moi nous sommes assis devant le bâtiment d'informatique, en nous joignant aux étudiants qui faisaient des piquets de grève. C'était quand Nixon avait envahi le Cambodge. Nous avons dit : "Eh bien, c'est une sorte de dernière goutte. Nous ne pouvons pas continuer à faire comme d'habitude." Bien qu'en fait, nous parlions entre nous d'algorithmes de tri. Et Bob et moi, en fait, inventions des réseaux de tri alors que nous étions assis là. Mais nous étions à l'extérieur du bâtiment parce que nous voulions être comptés dans les statistiques des gens qui disent telle ou telle chose, beaucoup de gens ont protesté aujourd'hui. Et les actions du gouvernement nous semblaient juste outrageuses. En général, je prends rarement position sur quoi que ce soit, mais je fais en sorte que lorsque j'agis, mon action exceptionnelle soit significative. Ce n'est pas comme si j'y allais à chaque souffle de vent.

FRANA : Vous avez également dénoncé SDI dans les années Reagan, n'est-ce pas ?

KNUTH : Je ne l'ai pas fait. Seulement en privé. Je ne pense pas avoir jamais

pris position publiquement à propos de ça.

FRANA : Ça doit être une conversation que j'ai eue avec quelqu'un à l'époque.

KNUTH : Je pense que toute personne qui savait quoi que ce soit sur les programmes de fiabilité était contre le SDI. Je n'étais pas exceptionnel à cet égard, c'était simplement un gâchis.

FRANA : Vous savez que Gio [Wiederhold] à Stanford a fait beaucoup de recherche sur les missiles ?

KNUTH : Je ne l'ai jamais su.

FRANA : Oh, tu ne le savais pas ? C'était son premier travail. Il a passé beaucoup d'années sur ces problème.

KNUTH : Je savais seulement qu'il était entré en médecine.

FRANA : Des fusées à combustible liquide, je suppose.

KNUTH : Oh, d'accord.

FRANA : Est-ce un problème ? Trouvez-vous cela troublant rétrospectivement, les recherches sur la guerre froide que les informaticiens ont faites ?

KNUTH : Eh bien, j'ai une expérience très limitée avec cela. J'ai passé un an travailler à l'IDA en cryptographie et toutes les personnes que j'ai rencontrées là-bas était engagées à fournir du bon travail. Je n'avais pas de retours sur ce que mes collègues là-bas faisaient. Je savais aussi qu'il n'était pas naturel pour moi de garder des secrets. J'étais le genre de personne qui veut être professeur et qui veut tout apprendre aux étudiants, alors qu'en travaillant pour eux, je n'étais pas autorisé à dire à ma femme ce sur quoi je travaillais. Tu sais, je ne lui en ai toujours pas parlé. je n'aimais pas être dans cette situation. J'ai donc pensé que c'était une année de service national, et que je partirais - non pas parce que je n'aimais pas le travail, c'était juste parce que je savais que ma propre contribution serait meilleure dans quelque chose où je ne dois pas garder le secret.

FRANA : Eh bien, passons à quelque chose de plus amusant. Vous et John von Neumann, Leo Szilard et plusieurs autres personnes êtes tous des luthériens.

KNUTH : Attendez une minute. Je ne peux pas le croire. John von Neumann ? Es-tu ...

FRANA : John von Neumann est allé dans un lycée luthérien.

KNUTH : Non. Je pensais qu'il était... d'accord, il n'est pas juif ? Tu sais, j'ai rencontré son frère Nicholas.

FRANA : Non, il n'est pas luthérien, mais il était élève au lycée luthérien de Budapest.

KNUTH : Oh, je ne le savais pas.

FRANA : Et je me demande s'il y a quelque chose de luthérien dans la science informatique.

KNUTH : C'est une question à laquelle il est très difficile de répondre. Ce serait comme dire : oh, une certaine quantité de gens boivent du lait pasteurisé quand ils sont jeunes, je ne sais pas. Je doute qu'il y en ait. Luther défend l'idée d'une activité intellectuelle indépendante. Dans la religion, il voulait que sa tête et son cœur soient là et non... il ne voulait pas être détaché de la logique, puis juste devenir... juste dire qu'il mérite d'être sauvé parce qu'il peut croire le plus de choses impossibles. Il a dit... non... continuons d'interroger les choses. Et c'est sûr qu'il y a des corrélations. Mais je ne dirais pas que les luthériens sont en avance sur les presbytériens, ou les musulmans, ou quoi que ce soit.

FRANA : J'ai aussi été élève dans un lycée luthérien, et c'était aussi mon impression, que Luther était un érudit et que c'était une bonne chose.

KNUTH : Oui, je viens d'une tradition où l'on peut remettre en question les choses. Mais alors, vous avez aussi ceci, "J'ai la foi", et votre conscience, et ainsi de suite. Et c'est très bien. Mais je dirais que la façon principale dont

ma foi intervient dans mon travail scientifique est le modèle que Dieu sait ce que je pense. Donc je ne ressens pas de besoin d'intimité comme le font beaucoup de gens. Dans un certain sens, je ne pense même pas avoir de vie privée. Malheureusement, Edsger a eu de très mauvaises expériences avec la religion dès le début ; il est devenu très sensible à ce sujet par la négative.

FRANA : Nous n'en avons jamais parlé.

KNUTH : Eh bien, bien. Il a insulté un de mes étudiants une fois parce que le gars parlait de programmation et Edsger a dit : "Vous êtes catholique ?"

FRANA : Wow. Avait-il raison ?

KNUTH : Oui, mais bien sûr qu'il l'était, le gars était français. Mais Edsger pensait qu'il y avait quelque chose de catholique dans sa façon de programmer, ce qui n'avait aucun sens. Quand j'ai écrit le livre 3 : 16, j'ai découvert à ma grande surprise que certaines personnes avaient des sentiments vraiment amers à l'égard de l'Eglise à cause de quelque chose qui leur était arrivé avec leurs parents, ou autre chose. Et c'était un aspect que je n'avais tout simplement pas connu auparavant. Plusieurs dizaines de personnes avec des antécédents différents se sont portés volontaires pour lire le premier projet de ce livre. J'avais donc des athées, qui écrivaient des lettres de dix pages, faisaient des commentaires sur le travail, car ils étaient également intéressés par les idées, mais ils m'ont également écrit des éléments de leur histoire et j'apprenais d'eux. Et Edsger a définitivement réagi contre la religion ; contre la religion organisée de toute façon. Mais je vais essayer de vous expliquer de quelle manière mon éducation religieuse a vraiment interagi avec ma vie scientifique. Et je pense que ce n'était pas parce que j'ai eu les meilleurs professeurs de sciences, ou des choses comme ça. Mon éducation religieuse m'a donné le sentiment de faire partie d'un mouvement mené par Dieu de l'univers ; cela a affecté les modèles que j'ai de ma propre vie d'une manière qui a probablement eu un certain effet. Je doute que cela me rende meilleur programmeur, ou pire programmeur. Cela affecte simplement quel type de programmeur je suis ; vous savez, je ne pense pas que je serais apte à faire un excellent travail en cryptographie parce que je ne suis pas assez sournois pour comprendre les attaques que quelqu'un pourrait faire, et aussi parce que je ne suis pas si convaincu que ça que j'ai besoin de secret.

FRANA : Vous acceptez la bonté fondamentale de la plupart des êtres humains ?

KNUTH : Eh bien, oui.

FRANA : Vous avez de l'espoir ou de la foi, je suppose.

KNUTH : Je peux comprendre la nécessité d'un faible niveau de sécurité, mais je ne peux pas être très facilement convaincu que la cryptographie forte est importante. Et peut-être que je pourrais, peut-être que je ne pourrais pas, mais de toute façon, à cause de ma nature basique, je ne pense pas que je suis apte à être une personne de haut niveau dans ce genre de travail.

FRANA : Maintenant, la raison pour laquelle je vous ai posé des questions sur Luther et l'informatique est qu'il y a une longue histoire des métiers de l'imprimerie dans la foi luthérienne. C'est l'histoire de la manière dont ils ont communiqué la Réforme aux gens.

KNUTH : Oui, c'est vrai.

FRANA : Cela a-t-il eu un impact sur votre décision de faire de la typographie numérique ?

KNUTH : Cela aurait pu être implicite parce que mon père travaillait beaucoup avec le domaine de l'impression.

FRANA : Oh, il l'a fait ?

KNUTH : Il a fait beaucoup de tels travaux pour toutes les églises autour de Milwaukee. Mon père avait une machine de miméographie à la maison et il faisait les programmes pour des événements spéciaux et ce genre de choses.

FRANA : Vous voulez dire des programmes de type dépliants, pas des programmes informatiques ?

KNUTH : Oui. C'est vrai. Je dois faire attention. Vous avez raison. Et des programmes musicaux, surtout pour les chœurs, et des choses comme ça ; il passait beaucoup de temps en fabrication à la main de matériaux à faible

coût pour chœurs. Et ce n'était pas de la haute technologie du tout que ces travaux d'impression. C'était à peu près le genre de travaux que vous faites quasiment à perte. C'était une mission pour lui ; ce n'était pas un moyen de gagner de l'argent. Il faisait ce travail mais il n'en a pas fait de profit. Il l'a fait comme service. En réalité, il a appelé ça "service", Erv's Service. Son nom était Ervin.

FRANA : Comment était votre père ? Je n'ai jamais lu aucun récit le concernant, si ce n'est qu'il était organiste.

KNUTH : Oh, c'était un bon cuisinier, un professeur charismatique, un directeur de chœur, et il avait un bon sens de l'humour. Il faisait tout avec beaucoup de responsabilité et a travaillé dur pour faire beaucoup de choses différentes. Il était trésorier du lycée luthérien, et il payait les professeurs de sa poche si l'argent du budget n'était pas là. Et il prenait cette responsabilité très au sérieux. Il s'est mis au service de la population locale, ou des gens du monde. Je ne fais pas autant pour mes propres amis qu'il a pu le faire. Il a tout fait pour ses amis. Mais il y a des gens en Russie, en Pologne et en Chine qui m'écrivent tout le temps en me remerciant pour ce que j'ai fait. Et ma vie est complètement différente de celle de mon père, parce qu'il était le genre de personne qui n'a jamais voulu se faire un nom. Il était très heureux de savoir qu'il y avait une centaine de personnes, des amis proches, dont il améliorait leur vie et c'est tout. Mais moi, je reçois de la gloire pour des choses qui ne prennent pas plus de temps, mais qui se trouvent juste être connectées d'une autre façon.

FRANA : La région de Palo Alto, cependant, n'est pas comme Milwaukee non plus.

KNUTH : Non. C'était le genre de gars qui fait vibrer l'Amérique. Il était vraiment l'un des rouages bien huilé qui fait fonctionner les choses. Tout ce qui devait être fait, il le faisait. Et ma maman est très similaire.

FRANA : Elle est toujours en vie ?

KNUTH : Elle a 89 ans maintenant.

FRANA : Votre père a-t-il aussi vécu jusqu'à un âge avancé ?

KNUTH : Non. Je pense qu'à son décès, il était plus jeune que moi maintenant, 63 peut-être, je pense, 62. Il est décédé très soudainement. Mais ma maman n'est toujours pas à la retraite. Elle travaille dans l'immobilier au centre-ville de Milwaukee. Et tous les deux faisaient toujours du bénévolat et diverses choses. Pas le genre de chose qui est acclamée. L'éducation luthérienne était vraiment l'objectif principal de mon père dans la vie. Et il avait été au collège à River Forest à Chicago, puis est devenu professeur en école élémentaire à Milwaukee, puis il est allé au lycée là-bas. Il considérait l'éducation comme une mission qui pouvait être accomplie dans une atmosphère aimante.

FRANA : Oui. Maintenant, vous avez choisi de compartimenter votre foi luthérienne, du moins au début. Était-ce par choix ?

KNUTH : Je ne comprends pas ce que vous voulez dire.

FRANA : Vous étiez "luthérien" le dimanche matin.

KNUTH : Oh, dans ce livre, j'ai dit...

FRANA : Je pense que c'est le mot que vous avez utilisé, "compartimenté".

KNUTH : Eh bien, j'ai utilisé le mot quand quelqu'un m'a posé une question concernant Bill Clinton. Mais j'ai dit cela parce que c'était juste, vous savez, que c'était un mot à la mode. Je dirais que quand j'étais jeune, j'étais à peu près moi-même à tous égards. J'étais une machine à faire des tests. J'ai appris les sujets afin que je puisse passer des examens. Mais je n'étais que dans des choses quantitatives ; quant à ce qui concerne les questions d'élégance, de beauté, et de grande littérature, et de telles choses, je ne connaissais pas tout ça... j'étais... comment diriez-vous cela ? J'ai été défié par le développement dans ces domaines jusqu'à ce que j'aie peut-être 30 ans. Et puis j'ai commencé à lire de grands livres, et à comprendre et vraiment apprécier la bonne musique. Avant ça, je savais faire plaisir à mon professeur et je pouvais fournir la bonne formulation, mais je n'avais pas cette âme. Mais quelque chose m'est arrivé alors que j'étais plus âgé.

FRANA : Qu'est-il arrivé ?

KNUTH : Je n'en ai aucune idée. Peut-être que j'étais trop occupé ou quelque chose comme ça. En tous cas, dans tous les aspects de ma vie, j'ai essayé d'obéir aux règles. Et je pense que j'ai toujours eu un complexe d'infériorité. À certains égards, je dirais bien, peut-être, que je dois prouver que je peux le faire. Au fond de moi, je savais que j'étais beaucoup plus intelligent que les gens ne le pensaient, mais j'essayais toujours de faire mes preuves, et je ne faisais pas des trucs parce que je voulais les faire, je les faisais parce que j'étais censé les faire. Non pas que je détestais faire le travail ; j'étais simplement respectueux. Et donc s'il était temps d'étudier les mathématiques, j'étudiais les mathématiques. Ainsi, j'agissais donc plutôt de façon mécanique. Et c'est plus tard que j'ai commencé à voir la partieplus émotionnelle de la vie. Qui sait pourquoi ? C'est probablement ma femme qui m'a donné ça.

FRANA : Où l'avez-vous rencontrée ?

KNUTH : Nous étions tous les deux étudiants. J'étais à Case et elle était à Reserve, et je sortais avec sa colocataire et nous sortions parfois ensemble. Et j'ai découvert qu'elle était une personne plutôt sympa. Alors je suis allé lui parler un jour de certains problèmes que je rencontrais avec sa colocataire. Et elle a donné de si belles réponses que j'ai décidé que je l'aimais mieux.

FRANA : La plus vieille histoire du monde. Sortez avec la colocataire et trouvez l'autre plus attrayante.

KNUTH : Oui. Elle avait un an de retard sur moi à l'école. Mais nous avons passé beaucoup de temps dans les bibliothèques à étudier ensemble parce que nous avons constaté qu'il y avait beaucoup de sujets dont nous aimions parler.

FRANA : Je voudrais vous poser quelques autres questions sur la technologie numérique avant de nous éloigner trop de cela. Il y a cette tentative de commentaire que vous faites dans l'un de vos livres, qu'une bonne typographie aide à écrire de bons programmes, ou de meilleurs programmes. Qu'est-ce que vous entendez par là ?

KNUTH : Je ne sais pas si j'ai dit cela au sujet des programmes... *[marque une pause]* Mais certainement, l'une des idées globales de la programmation structurée est que vous devez être capable de comprendre un ensemble

compliqué : “Quel est le programme.” Vous avez donc besoin de quelques aides pour aboutir à cette compréhension, et la typographie fait partie de ces aides. Avec une bonne typographie, vous pouvez percevoir la structure, au lieu d'imaginer le texte comme une chaîne chaotique de caractères. C'est beaucoup mieux lorsque ces caractères sont disposés sur une page d'une manière raisonnable. Donc, la typographie est en partie l'arrangement de ces choses, vous savez, comme l'indentation par exemple. Des choses qui sont soulignées ou des choses qui sont, vous savez, en petits caractères ou en italique. Et sans cette typographie, l'impact est négatif sur la façon dont je peux percevoir ce qui se passe.

Frana : Dont je peux percevoir l'intégralité d'un document.

Knuth : Mais je ne dis pas que la typographie soit une solution miracle. Je dis simplement que cela aide beaucoup. J'ai travaillé, en fait, avec le Journal de l'ACM dans les années 60, en partie pour aider à améliorer leurs normes de composition des programmes informatiques parce que c'était une nouveauté. Ils n'avaient pas été confrontés à cela avant. Et donc j'ai dit : “Pensons sérieusement à la bonne typographie pour les programmes, pour la compréhension des idées, pour faire passer le message que le programme signifie”. Dans ce nouveau domaine, nous devons rechercher des présentations adéquates.

Frana : Voici une question beaucoup plus précise : j'en sais beaucoup plus sur l'algorithme de Dijkstra que j'en connais sur de nombreux autres algorithmes, parce que nous avons eu une discussion approfondie à ce sujet. Mais est-ce que votre algorithme est similaire à celui permettant de trouver le plus court chemin entre la partie la plus haute et la partie la plus basse d'un paragraphe ?

Knuth : C'est pareil. La façon dont j'ai utilisé la méthode équivaut à l'Algorithme de Dijkstra, sauf que je dois construire le graphe au fur et à mesure. Je convertis le paragraphe en un problème de graphe, mais je découvre certaines parties du graphe, au fur et à mesure de ma lecture, pour faire court. C'est son algorithme, mais je gagne du temps en ne montrant pas les choses qui pourraient le ralentir. Par exemple, il est inutile d'explorer ce qui se passerait si vous arrêtiez le paragraphe en cassant une ligne après le premier mot. Droite. Je ne cherche donc que les points d'arrêt réalisables. Et connaître quels sont les points d'arrêt réalisables en appliquant l'algorithme

de Dijkstra, et comme je l'applique, je sais que certaines choses ne sont pas réalisables, donc je ne les considère pas. Il s'agit donc d'une extension de la méthode de Dijkstra. Mais vous pouvez également le voir comme ce qu'ils appellent la programmation dynamique, car discrète : la programmation dynamique est essentiellement identique à la recherche du plus court chemin. Donc de nombreux points de vue se résument au même problème sous-jacent.

FRANA : D'accord. Il y a quelques années, dans une interview, vous avez dit que la science des ordinateurs, comme la plupart des autres sciences, se développe principalement par beaucoup de petites étapes plutôt que par des bonds essentiels, principaux.

KNUTH : Ah bon, j'ai dit ça ? Oui bien. *[rire]*

FRANA : Je pense que vous avez utilisé le mot "principaux" comme qualificatif. Y a-t-il eu quelques bonds essentiels ?

KNUTH : Oui, l'idée d'une programmation structurée est un pas de géant, mais ceux-ci sont rares. Et donc regardez ce soir pendant la session des questions / réponses. Quelqu'un va dire : "Quels sont d'après vous les cinq algorithmes les plus importants, vous savez, ou quelque chose de similaire. Je déteste ce genre de question, parce que ce n'est pas vraiment la façon dont la science se développe. L'important n'est pas quels sont les cinq premiers ; parce que ce sont les cinq derniers. C'est juste qu'il y en a toujours cinq de plus. Et toute la structure se modifie par petites étapes. Malheureusement, je ne peux pas expliquer cela à un membre du Congrès qui va financer la NSF. Nous devons dire aux membres du Congrès que nous avons ce grand projet, et que nous nous sommes fixés tels objectifs, et tout ça. Mais vraiment, si vous prenez juste chaque science et que vous dites : "Faites ce que vous pensez être intéressant", alors vous obtenez la meilleure science qui soit.

FRANA : Donc, vous en savez un peu plus que la plupart des scientifiques sur l'Histoire des sciences, je suppose.

KNUTH : Sur certaines parties.

FRANA : Voyez-vous cela comme un processus cumulatif ? Sommes-nous tous debout sur les épaules de géants ? Ou existe-t-il des paradigmes kuh-

niens ?

KNUTH : Je pense que nous apprenons mieux en étudiant le passé et en voyant comment d'autres personnes ont trouvé leurs idées, et en apprenant le processus d'apprentissage, par osmose, après avoir vu tous ces exemples. Et certainement que la chose la plus frappante pour moi est que les êtres humains sont capables de revenir à leur histoire et de voir ce que les humains précédents ont fait et sont également capables de modifier leurs actions en fonction de cela. Les animaux font cela très lentement. Et donc c'est la clé pour moi, que de regarder des sources de matériaux et de voir comment étaient les idées quand elles étaient brutes. Je n'aurais pas travaillé aussi efficacement si je n'avais pas lu beaucoup de sources des nombreux siècles précédents. Si c'était nécessaire, je demandais à un ami "aidez-moi avec le chinois, le japonais ou le hongrois, ou quoi que ce soit", mais je regardais toujours des écrits des temps anciens qui étaient liés à ce que je faisais à un moment donné. Parce que pour moi, la façon dont les humains ont évolué est centrale à l'histoire. C'est peut-être juste parce que je suis égoïste et que je veux que des gens lisent mes trucs à l'avenir, donc je lis les trucs des autres pour payer ma dette.

FRANA : Eh bien, vous savez très bien, cependant, que le passé peut être un endroit étrange. En astronomie, par exemple, nous avons l'univers géocentrique.

KNUTH : Vous devez vous mettre dans l'esprit des anciens. En 1972, pendant deux semaines, j'écrivais un article sur les algorithmes babyloniens, et je me suis entouré de textes babyloniens. Je me suis immergé suffisamment, en lisant autant de tablettes que possible, de façon à être capable de dire : "Oh, oui, ceci est inhabituel", ou bien "celle-ci est du même auteur que celle-là" en regardant telle ou telle tablette. Et je pouvais commencer à essayer d'entrer dans l'esprit de ces auteurs. Ils avaient une certaine façon d'exprimer les choses. Ils n'utilisaient pas l'algèbre, mais ils avaient un équivalent. J'essayais donc d'apprendre ce que signifiaient leurs symboles et d'interpréter leur langue. C'est cette raison qui empêche beaucoup de gens de lire les sources : ils ne veulent pas ou ne peuvent pas réapprendre certaines notations. Cela signifie que, comme avec la musique, vous devez soit décider que vous êtes capable d'apprendre une autre notation ou bien que vous êtes capable de la convertir en votre propre truc. Avec les mathématiques, je peux

généralement le faire sans trop d'effort. Mais ça nécessite quelques heures d'adaptation. Avec un peu d'expérience, cela deviendrait plus facile pour vous aussi. La musique, je pense, serait plus difficile. C'est certain car il y a en musique quelques notations qui sont si illogiques qu'elles ne peuvent être apprises qu'après de nombreuses années de formation.

FRANA : Vous avez fait plusieurs fois le commentaire que peut-être 1 personne sur 50 a l'esprit du "scientifique de l'ordinateur".

KNUTH : Oui.

FRANA : Je me demande si un grand nombre de ces personnes sont des bibliothécaires professionnels ? *[rires]* Il y a là une certaine étrangeté. Mais pouvez-vous mettre en évidence ce qu'est l'esprit de l'informaticien...

KNUTH : C'est différent ?

FRANA : Quelles sont ses caractéristiques ?

KNUTH : Deux choses : la première est la capacité de gérer une structure non uniforme, où vous avez le cas un, le cas deux, le cas trois, le cas quatre. Par exemple, quand vous avez un modèle de quelque chose dont le premier composant est un entier, le composant suivant est un booléen, et le composant suivant est un nombre réel, ou quelque chose dont vous savez que sa structure est non uniforme. Gérer couramment ces types d'entités est critique en informatique, ce qui n'est pas typique dans d'autres branches des mathématiques. Et l'autre capacité caractéristique est de changer rapidement de niveau, de regarder quelque chose à grande échelle et la regarder à petite échelle, et avec beaucoup de niveaux de granularités entre les deux, en passant d'un niveau d'abstraction à un autre. Vous savez que, lorsque vous ajoutez un nombre à un certain nombre, ce que vous obtenez est réellement plus proche de l'objectif global. Ces compétences, être en mesure de gérer les objets non uniformes et de voir à travers les choses du niveau supérieur vers le bas niveau, sont des compétences très essentielles à la programmation informatique, il me semble. Mais peut-être que je me trompe parce que j'en suis trop proche.

FRANA : C'est la chose la plus difficile à comprendre que ce qui te caractérise vraiment au plus profond.

KNUTH : Oui.

FRANA : J'avais une question complémentaire tout à l'heure mais je l'ai maintenant perdue.

KNUTH : Est-ce à propos des 2 % ? Je pense que nous devons réaliser qu'il y a des gens que nous ne sommes pas en mesure de bien comprendre à 98 %, et ils ne sont pas en mesure de nous comprendre très bien. Mais nous pouvons construire des ponts. Nous pouvons nous faire des amis parmi ceux qui sont plus proches de nous, et obtenir un réseau de personnes qui rassemblent les choses. Je suis sûr que je ne peux pas écrire le bon manuel d'utilisation pour ma mère ; mais je pourrais collaborer avec quelqu'un qui pourrait le faire de la bonne façon. Certaines personnes ont cette capacité de changer pour penser différemment. Mais c'est seulement vrai dans une petite mesure. Voici un exemple qui a fait l'objet d'une récente discussion en Allemagne : et si l'informatique quantique prenait le dessus ? Et qu'est-ce qui se passerait si tout à coup, les gens pouvaient fabriquer ces choses, qui nécessitent une manière complètement différente d'écrire des programmes ? Eh bien, il se pourrait très bien que je ne sois absolument pas bon en informatique quantique et que je ne sois pas capable de changer. (Et il serait facile pour moi d'écrire l'*Art de la programmation des ordinateurs* si personne ne faisait des programmes à l'ancienne, donc je n'aurais pas à suivre la littérature). Mais j'ai une certaine façon de broyer des trucs pour lesquels je suis bon, et ce n'est pas seulement que je les ai étudiés au collège ; c'est que j'ai cette façon excentrique de penser.

FRANA : Comme la façon dont certaines personnes ont du mal à comprendre l'architecture client / serveur. Les vieux gars du mainframe qui, me disent-ils, ne comprennent tout simplement pas cela.

KNUTH : Oui. Je peux comprendre. C'est comme votre gars des logiciels de physique. Mais vous savez, ils doivent y parvenir d'une autre manière ou d'une autre, alors laissons-les être contents et faire ça à leur manière. Et peut-être que la nouvelle façon de faire sera mieux, mais je ne pourrai jamais procéder ainsi parce que je suis différent. Je pense qu'en informatique, il n'y a aucune voie étroite. C'est assez différent de nombreux autres domaines, car nous sommes sélectionnés par nos compétences, par notre profil de capacités, pas par notre mission de calcul. Les gens vont en médecine avec de nombreux

types de compétences ; leur carrière est basée sur l'objectif de sauver des vies, rendre la vie plus saine, peu importe. Mais ma carrière est différente. Il y a ce domaine dans lequel je suis bon et d'autres personnes sont heureuses que je sois bon dans ce domaine, donc je vais le faire. Mais je ne fais pas avancer une cause, comme la médecine.

FRANA : Maintenant, pensez-vous qu'il y a davantage - je déteste ce mot - mais davantage de science "normale" en cours en informatique ? Je veux dire, l'informatique quantique, qui sait si ça va marcher, c'est encore dans le futur.

KNUTH : Eh bien, il y a toutes sortes de collaborations en cours. Et le fait est que, comme les domaines deviennent plus spécialisés, je pense que la tendance va être que les gens se définiront davantage en ayant deux sous-sous-spécialités. Autrement dit, je pense que je l'ai déjà dit, il y aura des personnes qui parviennent à être bons dans un domaine : cela pourrait être l'informatique, et dans un autre domaine : cela pourrait être la chimie.

FRANA : Donc, chacun sera défini par ses deux sous-spécialités ?

KNUTH : Oui, c'est vrai, et cela fera une sorte de toile de disciplines. Et ça va être difficile pour le contrôle de la qualité car il n'y aura pas beaucoup de personnes ayant les deux mêmes sous-spécialités qui pourront faire passer des examens les autres. Il sera plus difficile de faire partie des comités à ce moment-là. Mais ça semble inévitable que le monde va devoir aller dans cette voie où les gens auront deux champs de compétence, une combinaison de capacités qui les rendra uniques ; ils se rendront compte que c'est ce pourquoi ils sont nés, car ils peuvent aider à combler cette lacune. Et il y a tellement de choses différentes à combler en science. Je trouve qu'il y a de nombreuses parties de la science qui pourraient m'intéresser d'un point de vue extérieur, mais je ne sens jamais ce que je pourrais y faire moi-même. Et il y a d'autres parties, où je sens que c'est ma mission de le faire, et j'en fais partie. Nous revenons toujours à Dijkstra parce qu'il est l'une des personnes les plus universelles que je connaisse. Sur presque tous les sujets dont nous parlons, il en connaîtrait un rayon incroyable.

FRANA : Mais si vous me le permettez, je dois dire que vous existez dans l'autre monde, parmi les 98 autres pour cent d'entre nous, bien mieux que lui.

KNUTH : Eh bien, il est intéressant que vous disiez cela. Les écrits de Dijkstra peuvent être plus spécialisés, mais pas ses conversations informelles.

FRANA : Vous semblez être capable de faire ce saut.

KNUTH : Eh bien, je ne suis pas sûr. En tant qu'étudiant, j'ai découvert que tout ce que j'étais à cette époque, me semblait-il, était un mélange de mathématiques et d'écriture. Une fois que j'avais ces compétences, je les appliquais simplement dans différentes proportions.

FRANA : Vous avez dit dans votre conférence pour l'obtention du prix Turing que "la science est ce que nous comprenons suffisamment bien pour l'expliquer à l'ordinateur, et que l'art, c'est tout le reste".

KNUTH : Oui, oui.

FRANA : Mais alors pourquoi appeler votre série de livres "*L'art de programmer les ordinateurs*" ?

KNUTH : Parce que nous ne le comprenons pas, nous ne l'avons pas rendu automatique. Nous n'avons pas encore eu d'algorithme pour écrire les programmes. Nous essayons de convertir la programmation en science, mais comme nous le faisons alors, l'art va de l'avant. Au fur et à mesure que la science progresse, l'art reste peut-être encore un peu en avance. Et puis je crois aussi que je décris l'art de la programmation informatique à cause de son élégance, de sa beauté, de son esthétique.

FRANA : Donc, vous y voyez vraiment à la fois une science et un art ?

KNUTH : Il y a de l'art, au sens des beaux-arts, et puis il y a de l'art dans le sens de l'artificiel, pas dans la nature. Dans ma conférence pour le prix Turing, j'ai essayé de méditer sur la raison pour laquelle je l'appelle l'art de la programmation informatique, et qu'est-ce que cela signifie. Je suis allé à la bibliothèque et j'ai trouvé cinquante livres qui avaient à la fois les mots art et science dans leurs titres. Et j'ai regardé le mot art dans l'histoire pour savoir comment il a été utilisé. Et je suis arrivé à la conclusion que la bonne façon de le comprendre est que l'art devient science quand nous atteignons un niveau auquel nous n'avons pas besoin de penser à ce sujet, nous pouvons

le programmer. Et c'est le plus grand mystère, du moins en science maintenant, de comprendre ce que signifie savoir quelque chose, avoir la cognition. C'est un mystère que de comprendre ce qu'est la conscience, mais si nous savions ce qu'était la conscience alors ma définition ne fonctionnerait pas. Je dis en quelque sorte qu'une fois que nous nous serons écartés de cette nécessité d'avoir un cerveau pour le comprendre, une fois que l'ordinateur le comprendra, cela deviendra alors une science.

Frana : Ce ne serait pas très amusant si nous comprenions vraiment la conscience, ne trouvez-vous pas ?

Knuth : Eh bien, qui sait ? *[rires]*

Frana : Je pense que nous pourrions peut-être passer à un niveau de conscience différent.

Knuth : Oui, c'est vrai, peut-être que nous pourrions encore avancer.

Frana : Je ne sais pas si vous avez lu L'homme de Turing de Jay David Bolter et sa critique de la vision cybernétique de la société. Sommes-nous devenus nos machines ?

Knuth : Ce genre de choses, le plus que j'y aie jamais pensé est dans mon livre, dans les conférences du MIT. La perspicacité que j'ai obtenue en étudiant le jeu de la vie était aussi proche qu'il est possible de comprendre certains de ces problèmes. Et d'autres personnes sont en avance sur moi sur ces choses. Mais tout ça mérite d'être exploré. Encore une fois, c'est comme l'astronomie, vous ne pouvez jamais y aller. Je suis donc très heureux d'avoir la plupart de mes questions telles que je sais que j'ai les réponses.

Frana : Mais vous ne voyez pas que la société est devenue numérique ?

Knuth : Certes, le monde a changé beaucoup plus rapidement à certains égards que j'ai jamais pensé que ce serait le cas, et les ordinateurs sont devenus beaucoup plus pertinents pour le monde que je ne l'aurais cru possible. C'est encore un oxymore pour parler comme "un célèbre informaticien", pourtant je pense que nous recevons trop d'attention ; trop de crédit pour certains trucs par rapport aux autres sciences. Mais cela s'arrêtera.

FRANA : C'est une question terriblement injuste, mais j'étais dans la Vallée il y a à un an, puis à nouveau cette année. Et à en juger par le nombre de devantures de magasins vides, les choses ont vraiment changé ici depuis la dernière fois l'année passée. Qu'est-il arrivé ?

KNUTH : Eh bien, l'année dernière a certainement été le sommet du boom. Mais si vous regardez en Iowa, ou n'importe où, vous constaterez qu'il y a beaucoup de chiffre d'affaires partout où vous allez et l'économie continue de se réinventer tout le temps. Il y a des cycles. Il y en a donc eu un, depuis longtemps, et maintenant l'économie mondiale va se réajuster. Les futurs historiens pourraient penser que l'année dernière était le moment où le boom était à son apogée. En d'autres termes, il y avait une telle pénurie de main-d'œuvre qualifiée, où tout ce que vous vouliez, vous aviez plus d'idées de choses à faire que les gens n'avaient de temps pour les faire. Donc, en quelque sorte, ce genre de système s'entretient lui-même.

FRANA : Mais alors nous avons manqué d'idées ?

KNUTH : Non. Non. Vous devez arriver au point où vous devez évaluer qui sont les gens de qualité, plutôt que ceux qui sont juste là pour la balade. Mais je suis écrivain et je ne connais pas grand chose à l'argent.

FRANA : Je sais que c'est une question terriblement injuste. Ils disent qu'il y a eu huit cycles d'explosion depuis la Seconde Guerre mondiale dans cette seule industrie, et c'en est juste un autre. Mais il y a des gens qui disent que c'est vraiment différent, et nous avons construit quelque chose au cours des années 90 qui était une sorte de schéma pyramidal.

KNUTH : Non. Je ne pense pas ça. Le monde est différent en quelque sorte, c'est-à-dire vous ne reviendrez pas à l'époque pré-internet. Nous devons être capable de décider combien et comment les gens vont payer pour ce nouveau monde où certaines choses sont possibles qui ne l'avaient jamais été auparavant. Alors maintenant, vous devons penser à ce qui est un moyen équitable de compenser cela, et ce que nous pouvons gérer, et comment le faire. Personne n'a de meilleur moyen de décider que par essai et erreur. C'est un peu comme à l'intérieur de notre corps, il y a toutes sortes de cellules qui en attaquent tout le temps d'autres ou qui les combattent, et cela semble être la

meilleure façon de faire face à des changements complexes. C'est chaotique, ces corpuscules vont ensuite combattre d'autres trucs et c'est une guerre là-dedans, et ce n'est pas d'appliquer simplement l'algorithme de Dijkstra pour trouver le chemin le plus court. Vous êtes en compétition.

FRANA : Nous rétablissons l'équilibre et retrouvons l'homéostasie ?

KNUTH : Oui, quelque chose comme ça. Mais le fait est qu'il y a un grand potentiel, qui prendra des années de travail, qui n'est pas réalisé par les machines que nous avons maintenant. Et il est difficile de faire ces travaux et cela demande beaucoup de travail. Mais je ne dirai jamais que c'est une idée stupide que de faire ce travail et que nous devrions tous nous arrêter de travailler là-dessus car personne ne va l'acheter. Ce serait la chose la plus stupide possible. Toutes ces compétences que les gens ont ici sont vitales et, en fait, nous allons avoir besoin de millions de personnes supplémentaires, et ma règle des 2 % prédit qu'il n'y en aura pas assez. Il va toujours y avoir une pénurie de compétences informatiques, et nous sommes loin d'exploiter les ordinateurs actuels pour tout ce qu'ils pourraient faire pour nous.

FRANA : Maintenant, je dois vérifier ici, mais avez-vous supervisé vingt-huit Ph.D. ?

KNUTH : Oui.

FRANA : Je sais que Stanford fournit des capitaux de démarrage aux étudiants qui ont des idées. Mais l'un de vos élèves a-t-il obtenu ce genre d'aide pour commencer et sont-ils pour la plupart des universitaires aujourd'hui ou sont-ils partis travailler dans l'industrie privée ?

KNUTH : Je pense qu'ils sont à peu près à moitié entre l'industrie et l'académie. Je crois que mon étudiant brésilien a fait une sorte de création d'entreprise. J'ai perdu le contact avec un ou deux d'entre eux, mais je pense que dans l'industrie, ils ont principalement travaillé dans des laboratoires de recherche - ne fondant rien eux-mêmes, mais en faisant partie d'une équipe que quelqu'un d'autre gère. À l'Université ils ont présidé certains départements et des choses comme ça, mais ce n'est pas la même chose que le démarrage de Yahoo ! Mais je dirais que ces vingt-huit élèves étaient vingt-huit jeunes gens complètement différents. Cela prouve mon idée que les informaticiens

savent comment traiter différents cas.

FRANA : C'est un bel exemple du monde réel. Je comprends que les gens sont imprévisibles ainsi. Certains d'entre eux sont-ils devenus vos plus proches confidents ?

KNUTH : Nous sommes proches de différentes manières. Je dirais que j'ai perdu de vue certains d'entre eux, mais avec d'autres, nous nous entraidons lorsque notre ordinateur tombe en panne, ou bien nous travaillons ensemble sur des projets éditoriaux, ou quoi que ce soit.

FRANA : Juste pour répéter la question, avec l'informatique dans son ensemble, ou le milieu universitaire dans son ensemble, pouvez-vous nommer les personnes qui sont vos plus proches confidents. Les gens que vous pensez pouvoir appeler pour faire rebondir une idée ?

KNUTH : Ma femme. De plus, comme j'écris des ébauches, quelles que soient les personnes expertes dans la section sur laquelle je travaille, je leur envoie toujours du matériel pour vérifier. Par exemple, la semaine dernière, j'ai rejeté une idée du professeur Carla Sauvage en Caroline du Nord. Je ne l'ai rencontrée qu'une seule fois et je ne l'appellerais pas ma confidente la plus proche ; mais d'une certaine manière, sur le sujet sur lequel je travaille maintenant, c'est un domaine dans lequel elle a de nombreuses publications. Et donc j'ai dit, "Carla, qu'est-ce que vous pensez de ce problème ? Je viens de travailler dessus pendant quatre heures. Je n'arrive pas à le résoudre, mais je ne veux pas le mettre à la poubelle, donc je ne pourrai même pas avoir le temps de le mettre dans mon livre." Puis elle a dit :" Hé Don, c'est un beau problème et je pense que je peux le faire.". Le lendemain, elle m'envoie une réponse et je regarde la réponse et j'ai dit : "Oui, il suffit de patcher de cette façon et maintenant je pense que c'est correct, et je vais le mettre dans mon livre, c'est un problème élégant et beau.". C'est juste un exemple de la façon dont je travaille maintenant. Donc vous voyez, je travaille par lots. Chaque six semaines, je suis dans une sous-culture différente de l'informatique et j'écris une partie différente des matériaux pour *L'art de la programmation informatique*. Et au cours de ces six semaines, j'ai un ensemble différent de confidents les plus proches.

FRANA : Je vous comprends. Vous avez donc vraiment un ensemble très

large de collaborateurs ?

KNUTH : Oui. C'est fantastique de voir combien de personnes m'ont aidé dans cette tâche. Et puis j'ai mis de nouveaux trucs sur Internet, et en une semaine, j'ai reçu une centaine de lettres. Un garçon de quatorze ans en Allemagne a récemment souligné que j'ai mal orthographié Nuremberg. (J'ai écrit Nuremburg.). Je reçois énormément d'appels à l'aide.

FRANA : Et c'est l'une des raisons pour lesquelles vous n'utilisez pas davantage le courrier électronique ?

KNUTH : Oh, je ne ferais jamais rien.

FRANA : De cette façon, vous pouvez toujours envoyer autant de messages que vous le souhaitez.

KNUTH : Je peux utiliser le Web, où les personnes intéressées peuvent trouver ceci ou cela. Je ne les leur envoie pas, mais ils les trouvent s'ils les recherchent. Mais avec Carla, je lui ai envoyé le message parce que je savais qu'elle était experte dans ce sujet et qu'elle ne serait pas gênée d'être interrompue.

FRANA : Ceci est la dernière question que j'ai, mais il peut y avoir d'autres choses que vous souhaiteriez ajouter à cette interview. Et peut-être que je trouverai les réponses ce soir au Xerox PARC : Comment est-ce d'être le porte-parole en chef de l'informatique ?

KNUTH : Je ne sais pas si je le suis.

FRANA : D'autres l'ont dit.

KNUTH : Je peux être un écrivain, qui essaie d'organiser les idées des autres en une sorte de structure plus cohérente pour qu'il soit plus facile de mettre les choses ensemble. Je peux voir que je peux être considéré comme un universitaire qui fait de son mieux pour vérifier les sources de matériel, afin que les gens obtiennent du crédit là où il est dû. Et pour vérifier les faits, non seulement pour abstraire quelque chose, mais pour voir quelles étaient les méthodes qui ont aidé à cette abstraction et pour combler les trous si néces-

saire. Je considére mon rôle comme étant celui de quelqu'un qui est capable de comprendre les motivations et les logiques d'un groupe de spécialistes et de les résumer dans une certaine mesure afin que les gens dans d'autres parties du domaine puissent les utiliser. J'essaye d'écouter les théoriciens et de sélectionner ce qu'ils ont fait d'important pour le programmeur de la rue ; j'élimine le jargon technique lorsque cela est possible. Mais je n'ai jamais été bon dans tout type de rôle qui serait de faire de la politique ou de conseiller les gens sur les stratégies, ou sur quoi faire. J'ai toujours été le meilleur pour raffiner des choses qui sont là et mettre de l'ordre dans le chaos. Je soulève parfois de nouvelles idées qui pourraient stimuler les gens, mais pas vraiment d'une manière à contrôler le débit. La seule fois où j'ai défendu quelque chose avec force, c'était quand j'ai inventé la notion de programmation littéraire ; mais je le fais toujours avec la mise en garde que cela fonctionne pour moi, mais que je ne sais pas si cela fonctionnerait pour quelqu'un d'autre. Quand je travaille avec un système que j'ai créé moi-même, je peux toujours le changer si je le souhaite. Mais tous ceux qui travaillent avec mon système doivent travailler avec ce que je leur donne. Je ne suis donc pas en mesure de juger mes propres affaires de manière impartiale. De toute façon, je me suis toujours senti mal lorsque quelqu'un me dit : "Don, pourriez-vous prévoir le futur" ou "Don, veuillez voter pour ou contre quelque chose.". Mais je me suis toujours senti à l'aise avec, "Don, pouvez-vous écrire un exposé à ce sujet, ou pouvez-vous vérifier ces faits ?". C'est ce que je pense faire assez bien. Maintenant, bien sûr, il y a des gens qui pensent que mes livres sont complètement impossibles à comprendre. Je peux accepter ça. Quand j'étais étudiant, j'écoutais les autres étudiants et ils disaient : "Oh, voici un livre de Williams sur les probabilités, il est vraiment difficile à comprendre ". Et donc je ne croyais jamais que je serais en mesure de comprendre un mot de ce qu'il avait dit et je ne lirais pas ses livres jusqu'à une dizaine d'années plus tard. Enfin quand j'ai eu le courage de les ouvrir, j'ai trouvé un exposé merveilleux. Mais quand même, je ne peux pas dire que mes livres sont faciles à comprendre. Tout ce que je peux dire, c'est qu'ils sont beaucoup plus faciles à comprendre que ce avec quoi j'ai travaillé. Je me suis rapproché d'une présentation simple, mais je n'y suis peut-être pas parvenu jusqu'au bout. J'essaie d'éviter le jargon sauf quand il est nécessaire. Je n'ai donc pas encore utilisé le mot "abjection" dans *L'Art*. Je pourrais, mais pourquoi ? Un jour si j'en ai besoin, je le ferai, mais je n'utilise pas de terminologie effrayante. Je connais des gens qui pensent que des mots comme ça sont hyper-mathématiques. Donc je fais bouillir des trucs et j'utilise le niveau de mathématiques que je dois, le cas

échéant, mais j'essaie de traduire à partir des différentes sous-cultures que mes sources sont dans un autre langage. J'ai reçu hier une lettre d'un mec, il a dit qu'il ne savait pas ce que je voulais dire quand j'ai dit "parité". Et donc j'ai réalisé, vous savez, je pensais que tout le monde savait que la parité signifiait la distinction entre pair ou impair. Mais non, ce gars ne le savait pas et il était motivé, vraiment, alors je l'ai aidé. Mais ce que j'essaie de dire, c'est que l'écriture technique, c'est ce que je fais bien. Mais porte-parole de l'informatique ? Je suis plutô un porte-parole pour les informaticiens...

Frana : Plutôt que pour l'informatique.

Knuth : Oui, essayer de régurgiter les choses que mes collègues ont faites d'une manière qui les rend plus mémorisables ou plus faciles à assembler. Donc ça signifie que je dois lire des trucs à partir de nombreuses sources - comme tous ces dossiers que je vous ai montrés à l'étage - et certains viennent de physiciens, et certains viennent d'ingénieurs électriciens, certains proviennent du travail en IA, certains proviennent de la théorie de la complexité, etc. Et j'apprends leurs mots à la mode, mais je ne les utilise pas moi-même, sauf si je le dois. C'est ce que je fais le mieux.

Frana : Espérons que nous nous retrouvions plus de dix mille jours dans le futur, avez-vous pensé à une épitaphe appropriée ?

Knuth : Épitaphe. Non. Non. Je suis toujours en vie. *[rires]* C'est intéressant, les gens qui ont écrit leur propre épitaphe. Non, je laisserai quelqu'un d'autre faire ça. Mais peut-être que ça devrait être un cryptarythme.

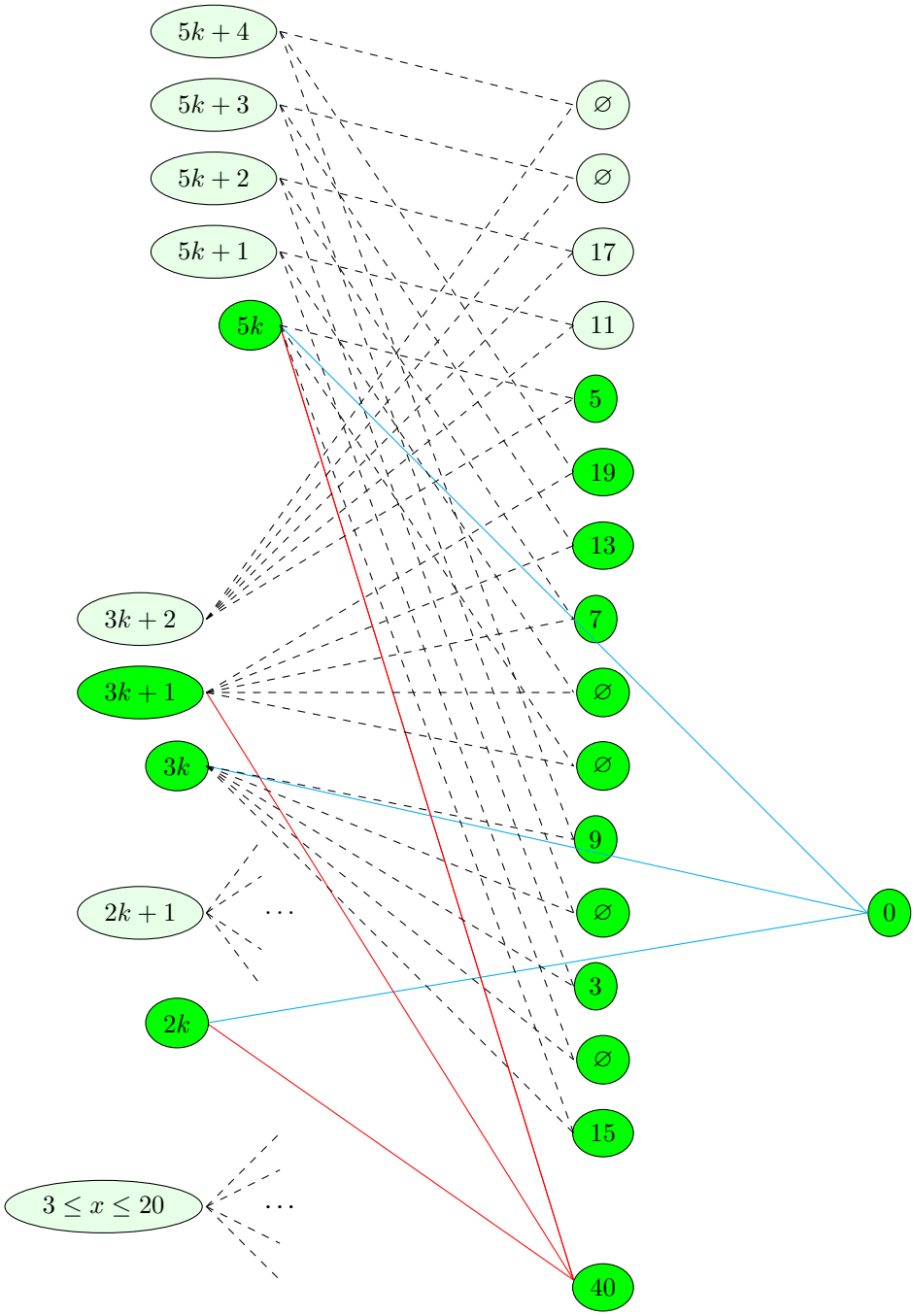
Orthogonalité

- On note $\mathcal{F}(x) = \{y \in \mathbb{N}^\times \mid 3 \leq y \leq x\}$
- On note $98^\perp$ l'ensemble des décomposants de Goldbach de 98.
- $98 \in (2\mathbb{N}) \cap (3\mathbb{N}+2) \cap (5\mathbb{N}+3) \cap (7\mathbb{N})$.
- $98^\perp \in \mathcal{F}(49) \cap [(2\mathbb{N}+1) \cap (3\mathbb{N}+1) \cap [(5\mathbb{N}+1) \cup (5\mathbb{N}+2) \cup (5\mathbb{N}+4)] \cap [(7\mathbb{N}+1) \cup \ldots \cup (7\mathbb{N}+6)]]$.

  Attention : Bien avoir à l'esprit que les règles de développement de la multiplication $\cap$ sur l'addition $\cup$ se font comme habituellement en algèbre, i.e. l'union $(7\mathbb{N}+1) \cup \ldots \cup (7\mathbb{N}+6)$ ne représente pas tous les entiers sauf les multiples de 7 par exemple. On développe par distributivité d'une opération sur l'autre.

- Plus généralement, si on note $\mathcal{P}$ l'ensemble des nombres premiers.
- $n \in \bigcap_{a \in \mathcal{F}(\sqrt{n}) \cap \mathcal{P}^\times, b \in \{0,\ldots,a-1\}} \{ax+b\}$ et
- $n^\perp \in \mathcal{F}(n/2) \cap \bigcap_{a \in \mathcal{F}(\sqrt{n}) \cap \mathcal{P}^\times, b' \in \{1,\ldots,a-1\}} \{ax+b', \text{ avec } b' \neq b, \forall a\}$.

Treillis distributif pour décomposants de Goldbach de 40 (Denise Vella-Chemla, 13.6.2020)



Sont notées en vert clair (resp. en vert vif) les solutions potentielles (resp. tout ce que le filtre élimine). On notera que le filtre élimine 3 alors que 3 est un décomposant de Goldbach de 40 car on fait le choix de ne conserver que les nombres premiers décomposants de Goldbach supérieurs à la racine carrée du nombre pair considéré.

Note : Pour ne pas surcharger le dessin, on n'a pas noté les liens de relation d'intersections entre les $\{2k+1\}$ et tous les nœuds à droite (on se concentre à la recherche de décomposants de Goldbach d'un nombre pair sur les nombres impairs, qui sont tous potentiellements solutions (i.e. notés en vert clair)). C'est la borne (la restriction à l'intervalle $[3, n/2]$, symbolisée sur le schéma par le nœud $3 \leq x \leq 20$) (vert clair) qui pose peut-être problème pour prouver l'existence d'un décomposant de Goldbach pour tout nombre pair.

On trouvera ici, là, là, ou encore là des éléments de référence concernant les treillis distributifs, les espaces de Stone, les espaces booléens. Un décomposant de Goldbach de n appartient au complémentaire

ou ensemble orthogonal de l'ensemble $\{n\} \cup \{0\}$. On peut se reporter également aux articles de wikipedia suivants : Filtre topologique, Espace de Stone, Treillis, Algèbre de Boole.

Pour bien décrire l'espace dans lequel on se place, on recopie texto un extrait de l'article de wikipedia *Théorème de représentation de Stone pour les algèbres de Boole* :
https://fr.wikipedia.org/wiki/Th%C3%A9or%C3%A8me_de_repr%C3%A9sentation_de_Stone_pour_les_alg%C3%A8bres_de_Boole :
Exemple : L'algèbre des parties finies ou cofinies
Soit E un ensemble infini. Posons $FC(E)$ l'ensemble des parties finies ou cofinies[1] *$FC(E)$ forme une algèbre de Boole avec les opérations ensemblistes usuelles (union, intersection, complémentaire). Les ultrafiltres de $FC(E)$ sont :*

- *Les ultrafiltres principaux U_x , dont les éléments sont les parties finies ou cofinies de E contenant un élément x donné de E,*
- *L'ultrafiltre U_∞, dont les éléments sont les parties cofinies de E.*

Par conséquent, l'espace de Stone $S(FC(E))$, ensemble des ultrafiltres précédents, peut être assimilé à l'ensemble E auquel on a ajouté un élément ∞. Topologiquement, il s'agit du compactifié d'Alexandrov de E, lorsque ce dernier est muni de la topologie discrète. Les parties ouvertes-fermées de $S(FC(E))$ sont d'une part les parties finies incluses dans E, d'autre part les parties cofinies contenant ∞. L'ensemble de ces parties ouvertes-fermées, avec les opérations ensemblistes usuelles, redonnent une algèbre de Boole isomorphe à $FC(E)$[2]*.*

Transposons au problème qui nous intéresse :
Soit $\mathbb{N}$ l'ensemble infini des entiers naturels. Posons $FC(\mathbb{N})$ l'ensemble des parties finies ou cofinies de $\mathbb{N}$. $FC(\mathbb{N})$ forme une algèbre de Boole avec les opérations ensemblistes usuelles (union, intersection, complémentaire). Les ultrafiltres de $FC(\mathbb{N})$ sont :

- Les ultrafiltres principaux U_x, dont les éléments sont les parties finies ou cofinies. de $\mathbb{N}$ contenant un élément x donné de $\mathbb{N}$,
- L'ultrafiltre U_∞, dont les éléments sont les parties cofinies de $\mathbb{N}$.

Par conséquent, l'espace de Stone $S(FC(\mathbb{N}))$, ensemble des ultrafiltres précédents, peut être assimilé à l'ensemble $\mathbb{N}$ auquel on a ajouté un élément ∞. Topologiquement, il s'agit du compactifié d'Alexandrov de $\mathbb{N}$, lorsque ce dernier est muni de la topologie discrète. Les parties ouvertes-fermées de $S(FC(\mathbb{N}))$ sont d'une part les parties finies incluses dans $\mathbb{N}$, d'autre part les parties cofinies contenant ∞. L'ensemble de ces parties ouvertes-fermées, avec les opérations ensemblistes usuelles, redonnent une algèbre de Boole isomorphe à $FC(\mathbb{N})$.

La notion d'ultrafiltre principal permet de calculer les intersections du complémentaire de $\{n\} \cup \{0\}$ avec les intervalles de la forme $[0, n/2]$ pour trouver les décomposants de Goldbach ; on ne sait pas si le fait d'avoir mieux "délimité" l'espace sur lequel il convient de se placer permet de prouver l'existence d'un décomposant de Goldbach pour tout entier pair supérieur ou égal à 4 ou pas.

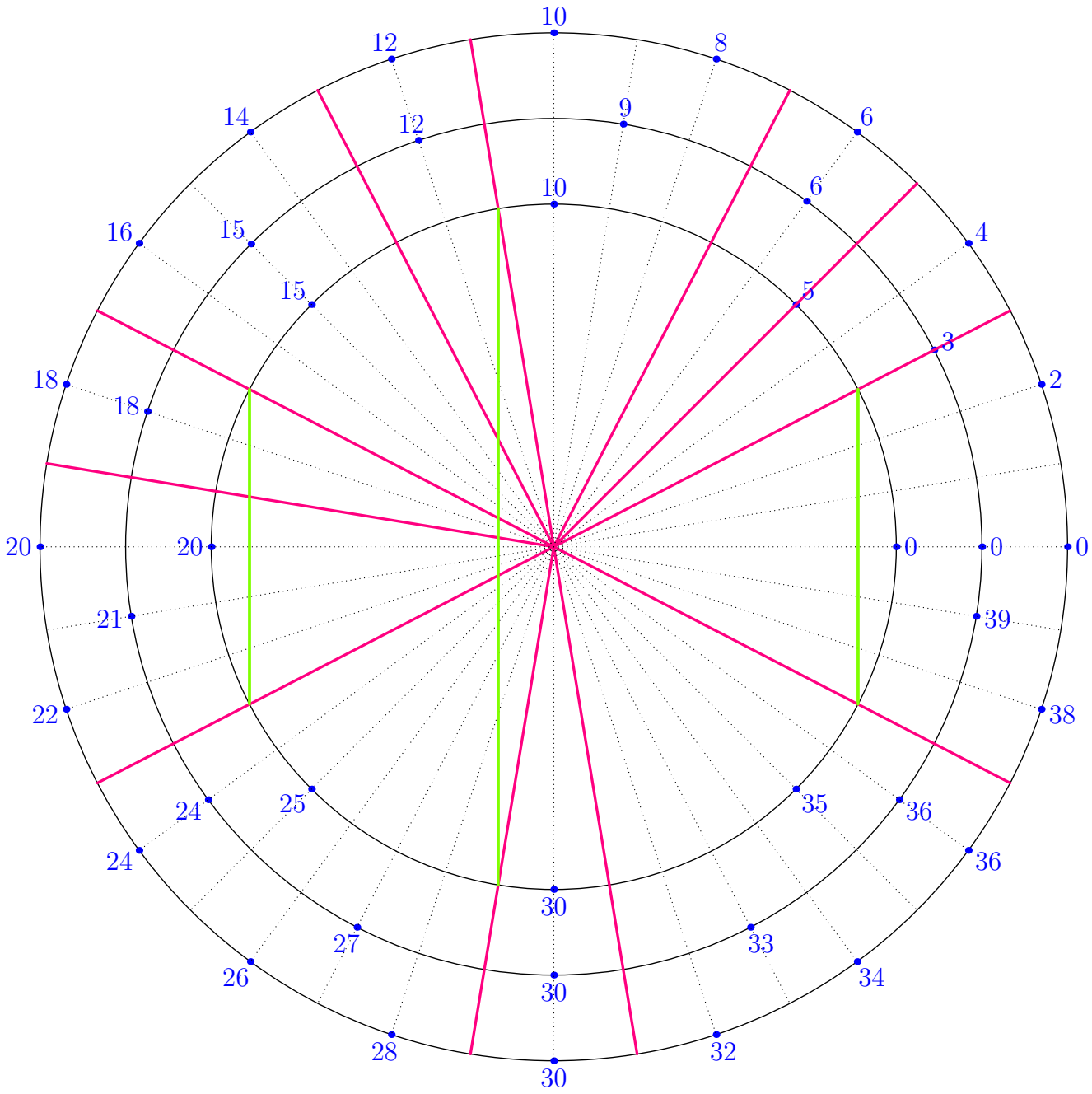
1. cofini = dont le complémentaire est fini. Par exemple, $[20, \infty]$ est une partie cofinie de $\mathbb{N}$ de E.
2. Il suffit d'"oublier" l'élément ∞ pour retrouver $FC(\mathbb{E})$.

Décomposants de Goldbach de $n = 40$ (Denise Vella-Chemla et le fan d'Asymptote, 13.7.2020)

Sur chaque cercle concentrique, on note les multiples des nombres premiers inférieurs à $\sqrt{n}$, ici 2, 3 et 5 (plus le cercle est petit, plus le nombre premier est grand). Les rayons fuchsia montrent les nombres premiers.

Les traits verticaux verts lient deux nombres premiers complémentaires décomposants de Goldbach de n.

$40 = 3 + 37 = 11 + 29 = 17 + 23$.

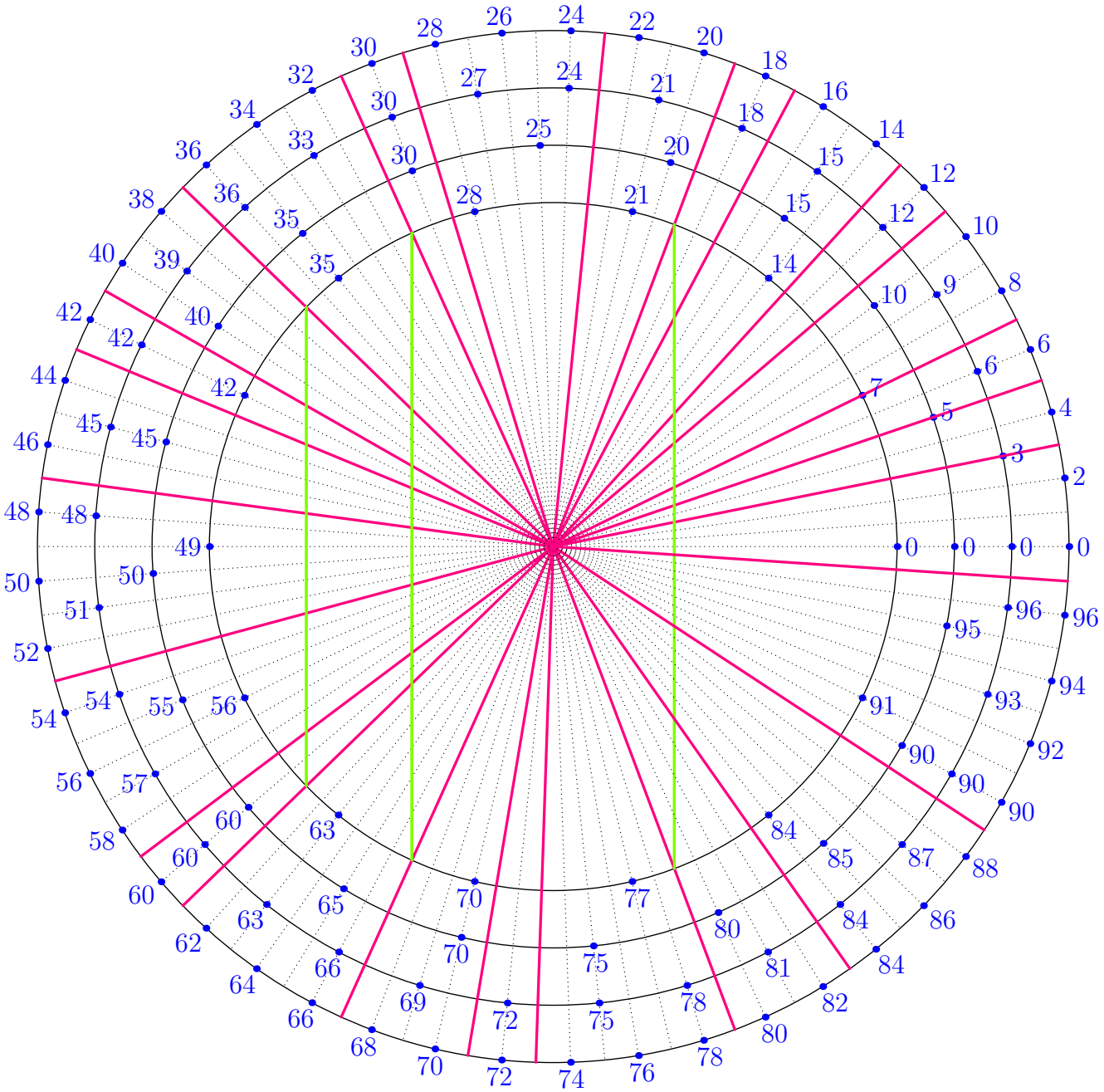


Décomposants de Goldbach de $n = 98$ (Denise Vella-Chemla et le fan d'Asymptote, 13.7.2020)

Sur chaque cercle concentrique, on note les multiples des nombres premiers inférieurs à $\sqrt{n}$, ici 2, 3, 5 et 7 (plus le cercle est petit, plus le nombre premier est grand). Les rayons fuchsia montrent les nombres premiers.

Les traits verticaux verts lient deux nombres premiers complémentaires décomposants de Goldbach de n.

$98 = 19 + 79 = 31 + 67 = 37 + 61$.

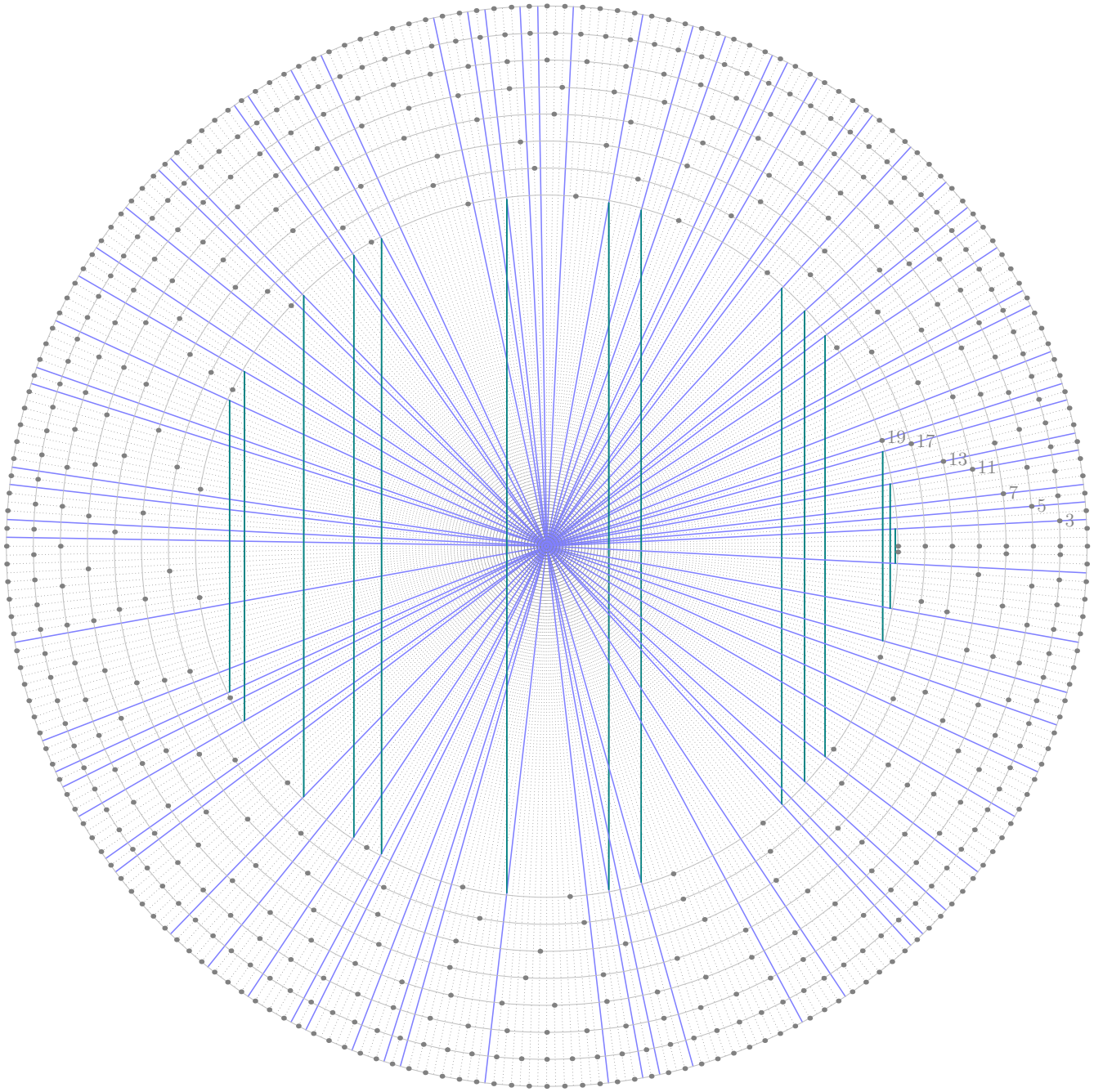


Décomposants de Goldbach de $n = 400$ (Denise Vella-Chemla et le fan d'Asymptote, 14.7.2020)

Sur chaque cercle concentrique, on note (par des petits points ronds gris) les multiples des nombres premiers impairs inférieurs à $\sqrt{n}$, ici 2, 3, ..., 19 (plus le cercle est petit, plus le nombre premier est grand). Les rayons bleu ciel montrent les nombres premiers.

Les traits verticaux verts foncés lient deux nombres premiers complémentaires décomposants de Goldbach de n.

$400 = 3 + 397 = 11 + 389 = 17 + 383 = 41 + 359 = 47 + 353 = 53 + 347 = 83 + 317 = 89 + 311 = 107 + 293 = 131 + 269 = 137 + 263 = 149 + 251 = 167 + 233 = 173 + 227.$



Triangulaires (Denise Vella-Chemla, 20.7.2020)

On s'intéresse aux nombres de nombres premiers inférieurs aux puissances de 10 successives qu'on trouve par programme. $\pi(x) = 25$, le nombre de nombres premiers inférieurs à 100 "ressemble" à 28, le septième nombre triangulaire $\Delta_7 = \frac{7 \times 8}{2}$.

On cherche de quels nombres triangulaires les nombres de nombres premiers sont les plus proches. Jusqu'à 10^8, on trouve les nombres triangulaires fournis dans le tableau ci-dessous. On note en regard les différences entre la fonction $li(x)$ de Gauss et les nombres $\pi(x)$ ou bien entre "nos" nombres triangulaires et les $\pi(x)$ en question. Ces calculs sympathiques nous ramène au Eureka que Gauss nota dans son journal, lorsqu'il découvrit que tout nombre est la somme de trois nombres triangulaires [1] le 10 juillet 1796.

x	$\pi(x)$	Δ_k	$li(x) - \pi(x)$	$\Delta_k - \pi(x)$
10^2	25	$\Delta_7 = 28$	5	3
10^3	168	$\Delta_{18} = 171$	10	3
10^4	1 229	$\Delta_{49} = 1225$	17	4
10^5	9 592	$\Delta_{138} = 9\ 591$	38	1
10^6	78 498	$\Delta_{396} = 78\ 606$	130	208
10^7	664 579	$\Delta_{1152} = 664\ 128$	339	451
10^8	5 761 455	$\Delta_{3394} = 5\ 761\ 315$	754	140
10^9	50 847 534	$\Delta_{10\ 084} = 50\ 848\ 570$	1701	1036

On cherche alors dans OEIS une séquence d'entiers qui ressemblerait le plus possible à la séquence 7,18,49,138, etc. On trouve la séquence $a(n) = \frac{1 + 2^n + 3^n}{2}$ mais cette séquence aurait amené au nombre triangulaire $T_{397} = 79\ 003$ au lieu de T_{396}, puis à $T_{1\ 158} = 671\ 061$ au lieu de $T_{1\ 152}$, puis à $T_{3\ 409} = 5\ 812\ 345$ au lieu de $T_{3\ 394}$ et $T_{10\ 098} = 50\ 989\ 851$ au lieu de $T_{10\ 084}$ et alors, les écarts faibles à $\pi(x)$ pour les petites valeurs de x s'envolent d'une manière complètement rédhibitoire : $505 \gg 130$ pour 10^6, $6482 \gg 339$ pour 10^7, $50890 \gg 754$ pour 10^8, $142317 \gg 1701$ pour 10^9.

Il faudrait trouver une façon plus judicieuse de passer d'un nombre triangulaire au suivant.

On utilise la calculatrice en ligne à l'adresse https://fr.planetcalc.com/5992/ pour approximer la fonction qui à 10^2 associe 7, à 10^3 associe 18, à 10^4 associe 49, à 10^5 associe 138, à 10^6 associe 396, à 10^7 associe 1152 et à 10^8 associe 3394. La calculatrice en ligne fournit la fonction $f(x) = 0.8207x^{0.4492}$, avec un coefficient de corrélation de 0.9983, une erreur relative moyenne de 3.8473%, et un coefficient de détermination de 0.9966. Malheureusement, les triangulaires calculés par cette fonction $T_6, T_{18}, T_{51}, T_{144}, T_{406}, T_{1144}, T_{3219}$ sont bien trop éloignés des valeurs de $\pi(x)$ pour convenir. On a à nouveau donné un coup d'épée dans l'eau.

1. cf. http://denisevellachemla.eu/eureka3t.jpg et http://www.persee.fr/doc/rhs$_0$048 − 7996$_1$956$_n$um$_9$14346.

Paquets (Denise Vella-Chemla, 22.7.2020)

On voudrait utiliser une caractérisation simple, puérile, des nombres premiers. On utilise la fonction

$$F(n) = \sum_{k=1}^{n} \left\lfloor \frac{n}{k} \right\rfloor .$$

On caractérise alors les nombres premiers par

$$n \text{ est premier } \iff F(n) - F(n-1) = 2.$$

Voyons cette caractérisation élémentaire dans un tableau contenant les $\left\lfloor \frac{n}{k} \right\rfloor$ pour n de 1 à 13.

n	$k=1$	2	3	4	5	6	7	8	9	10	11	12	13	Σ	Δ	1^{er}
1	1													1	1	
2	2	1												3	2	$*$
3	3	1	1											5	2	$*$
4	4	2	1	1										8	3	
5	5	2	1	1	1									10	2	$*$
6	6	3	2	1	1	1								14	4	
7	7	3	2	1	1	1	1							16	2	$*$
8	8	4	2	2	1	1	1	1						20	4	
9	9	4	3	2	1	1	1	1	1					23	3	
10	10	5	3	2	2	1	1	1	1	1				27	4	
11	11	5	3	2	2	1	1	1	1	1	1			29	2	$*$
12	12	6	4	3	2	2	1	1	1	1	1	1		35	6	
13	13	6	4	3	2	2	1	1	1	1	1	1	1	37	2	$*$

Il faudrait alors utiliser la fonction suivante

$$g(n) = n - \frac{1}{2} + \sum_{k=1}^{\infty} \frac{sin(2k\pi n)}{k\pi}$$

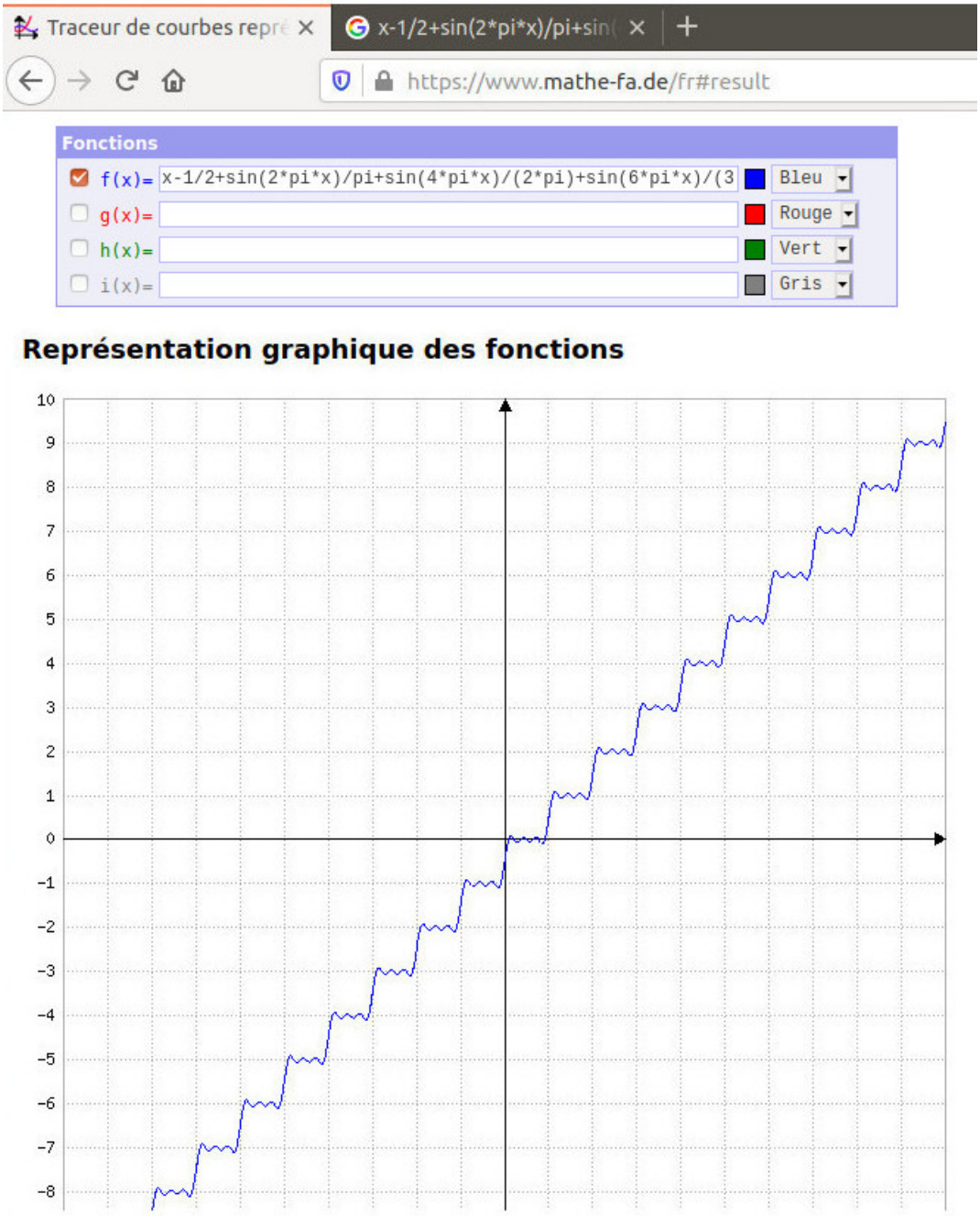
pour approximer la fonction *floor* (notée $\lfloor x \rfloor$).

En prenant seulement les 3 premières valeurs de k, en calculant plutôt que $g(n)$ la fonction

$$g_3(n) = n - \frac{1}{2} + \frac{sin(2\pi n)}{\pi} + \frac{sin(4\pi n)}{2\pi} + \frac{sin(6\pi n)}{3\pi},$$

on a déjà une fonction assez "ressemblante" de la fonction *floor*.

Utilisons un logiciel de visualisation en ligne de fonctions https://www.mathe-fa.de/frresult [1] :



1. Elle passe par les points intermédiaires médians pour les entiers, comme la fonction zeta de Riemann.

Si on appelle d le nombre de termes de la somme de sinus, on a :

$$\begin{aligned} F(n) &= \sum_{k=1}^{n} \left\lfloor \frac{n}{k} \right\rfloor \\ &= \sum_{k=1}^{n} \left[\frac{n}{k} - \frac{1}{2} + \frac{1}{\pi} \sum_{l=1}^{d} \frac{sin(2\pi l \frac{n}{k})}{l} \right] \\ &= n \sum_{k=1}^{n} \frac{1}{k} - \frac{n}{2} + \frac{1}{\pi} \sum_{k=1}^{n} \sum_{l=1}^{d} \frac{sin(2\pi l \frac{n}{k})}{l} \\ &= n \sum_{k=1}^{n} \frac{1}{k} - \frac{n}{2} + \frac{1}{\pi} \sum_{l=1}^{d} \frac{1}{l} \sum_{k=1}^{n} sin\left(\frac{2\pi l n}{k}\right) \end{aligned}$$

De toute façon, comme indiqué dans l'article wikipedia concernant la fonction *floor*, la formule d'approximayion n'est pas valide lorsque k divise n et cette idée s'avère être une très mauvaise idée, un nouveau coup d'épée dans l'eau.

```
from math import *
from matplotlib import *
from matplotlib.pyplot import *

def vecteur(point1, point2):
  return [y - x for x, y in zip(point1, point2)]

def add(vecteur1, vecteur2):
  return [x + y for x, y in zip(vecteur1, vecteur2)]

def norme(vecteur):
  return sqrt(prodscal(vecteur, vecteur))

def prodscal(vecteur1, vecteur2):
  return sum([x*y for x, y in zip(vecteur1, vecteur2)])

def determ(vecteur1, vecteur2):
 return vecteur1[0]*vecteur2[1]-vecteur1[1]*vecteur2[0]

def angle(vecteur1, vecteur2):
  cosinus=prodscal(vecteur1, vecteur2)/(norme(vecteur1)*norme(vecteur2))
  sinus=determ(vecteur1, vecteur2)/(norme(vecteur1)*norme(vecteur2))
  return atan2(sinus,cosinus)

def rotation(u, theta):
  return [u[0]*cos(theta)-u[1]*sin(theta),u[0]*sin(theta)+u[1]*cos(theta)]

def intersecte(x, y, z, t):
  a1 = (y[1]-x[1])/(y[0]-x[0])
  b1 = x[1]-a1*x[0]
  a2 = (t[1]-z[1])/(t[0]-z[0])
  b2 = z[1]-a2*z[0]
  return [(b2-b1)/(a1-a2),((b2-b1)/(a1-a2))*a1+b1]

a=[8.83, 47.89]
b=[72.74, 8.55]
c=[90.72, 86.63]
ab = vecteur(a,b) ; ac = vecteur(a,c)
ba = vecteur(b,a) ; bc = vecteur(b,c)
ca = vecteur(c,a) ; cb = vecteur(c,b)
anglea=angle(ab,ac)  ; angleb=angle(bc,ba)  ; anglec=angle(ca,cb)
aprime = add(a, rotation(ab,anglea/3.0))  ; aseconde = add(a,
rotation(ab,anglea*2.0/3.0))
bprime = add(b, rotation(bc,angleb/3.0)) ; bseconde = add(b,
rotation(bc,angleb*2.0/3.0))
cprime = add(c, rotation(ca,anglec/3.0)) ; cseconde = add(c,
rotation(ca,anglec*2.0/3.0))
p=intersecte(a,aseconde,b,c) ; q=intersecte(a,aprime,b,c)
r=intersecte(c,cseconde,a,b) ; s=intersecte(c,cprime,a,b)
t=intersecte(b,bseconde,c,a) ; u=intersecte(b,bprime,c,a)
n=intersecte(a,aseconde,c,cprime)
o=intersecte(c,cseconde,b,bprime)
x=intersecte(b,bseconde,a,aprime)
print('Normes des cotes %3.15f '% norme(vecteur(n,o)))
print('Normes des cotes %3.15f '% norme(vecteur(o,x)))
print('Normes des cotes %3.15f '% norme(vecteur(x,n)))

fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
matplotlib.pyplot.plot([a[0],b[0],c[0],a[0]],[a[1],b[1],c[1],a[1]], 'g',
alpha=0.7)
matplotlib.pyplot.fill([a[0],p[0],q[0]],[a[1],p[1],q[1]], 'g', 2, alpha=0.4)
matplotlib.pyplot.fill([c[0],r[0],s[0]],[c[1],r[1],s[1]], 'g', 2, alpha=0.4)
matplotlib.pyplot.fill([b[0],t[0],u[0]],[b[1],t[1],u[1]], 'g', 2, alpha=0.4)
```

```
matplotlib.pyplot.fill([n[0],o[0],x[0]],[n[1],o[1],x[1]], 'g', 2, alpha=0.8)
matplotlib.pyplot.xlim(0,100)
matplotlib.pyplot.ylim(0,100)
ax.set_aspect('equal')
matplotlib.pyplot.show()
```

Poursuite de petites recherches au sujet de certains décomposants de Goldbach en lien avec l'algorithme d'Euclide étendu (Denise Vella-Chemla, 9.8.2020)

1) Expérimentations

On a effectué par ordinateur quelques calculs expérimentaux utilisant l'algorithme d'Euclide étendu pour tenter de mettre en relation un décomposant de Goldbach d'un nombre pair $n \geq 6$ (un nombre premier p_1 dont le complémentaire $p_2 = n - p_1$ à n est premier lui aussi) avec un autre décomposant de Goldbach de n.

Pour rappel, l'algorithme d'Euclide étendu permet de trouver pour deux nombres entiers a et b deux autres nombres entiers x et y solutions de l'équation $ax + by = 1$.

On identifie par ces expériences de programmation au moins deux sortes de nombres pairs (on note n le nombre pair considéré) :

- les cas, très nombreux (97% des nombres jusqu'à 10000), pour lesquels l'algorithme d'Euclide étendu appliqué entre un premier nombre a et n d'une part, et un second nombre b et n d'autre part, établit une relation symétrique entre a et b et où l'on a simultanément a et b qui sont tous les deux des décomposants de Goldbach de n : il existe une configuration de nombres a, b, z tels que

$$\text{PGCD}_e(a, n) = (\pm b, z) \text{ et } \text{PGCD}_e(b, n) = (\pm a, z)$$

  et $a, b, n - a$ et $n - b$ sont tous premiers.
  Exemple : $n = 92$.
  $\text{PGCD}_e(3, 92) = (31, -1)$ (car $3 \times 31 + (-1) \times 92 = 1$) et
  $\text{PGCD}_e(31, 92) = (3, -1)$ et
  3 est un décomposant de Goldbach de 92 (89 est premier) et
  31 est lui aussi un décomposant de Goldbach de 92 (61 est premier). Les 2 nombres en question sont deux inversibles du groupe $((\mathbb{Z}/n\mathbb{Z})^*, \times)$.

- les nombres pairs pour lesquels aucun cas de symétrie tel que celui décrit ci-dessus n'existe. Jusqu'à 100, cela concerne les nombres 20, 28, 32, 44, 52, 64, 76 et 88.

2) Rappel de l'application de l'algorithme d'Euclide étendu sur un exemple

Les premier et second éléments d'une ligne sont les second et troisième éléments de la ligne qui la précède dans le tableau. Les colonnes x et y sont utilisés pour la seconde représentation matricielle présentée plus loin.

Dividende (a)	*Diviseur* (b)	*Reste* (r)	*Quotient* (q)	Relation invariante ($r = a - bq$)	x	y
$a = 385$	$b = 156$	73	2	$73 = a - 2b$	1	-2
156	73	10	2	$10 = b - 2 \times 73 = -2a + 5b$	-2	5
73	10	3	7	$3 = 15a - 37b$	15	-37
10	3	1	3	$1 = -47a + 116b$	-47	116
3	1	0	3	$-47 \times 385 + 116 \times 156 = 1$		

Pour une représentation matricielle, on utilise des matrices de la forme $\begin{pmatrix} a & b \\ 0 & 1 \end{pmatrix}$ (cf. [1] et [2]).

L'exécution de l'algorithme peut être mise en correspondance avec la multiplication des matrices :

$$\begin{aligned}
&\begin{pmatrix} 2 & 73 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 10 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 7 & 3 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 3 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 3 & 0 \\ 0 & 1 \end{pmatrix} \\
&= \begin{pmatrix} 2 & 73 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 10 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 7 & 3 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 9 & 1 \\ 0 & 1 \end{pmatrix} \\
&= \begin{pmatrix} 2 & 73 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 10 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 63 & 10 \\ 0 & 1 \end{pmatrix} \\
&= \begin{pmatrix} 2 & 73 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 126 & 30 \\ 0 & 1 \end{pmatrix} = N \\
&= \quad \begin{pmatrix} 252 & 133 \\ 0 & 1 \end{pmatrix} = M
\end{aligned}$$

La somme des éléments de la première ligne de la matrice M est $252 + 133 = 385$, le nombre a dividende initial, tandis que la somme des éléments de la première ligne de la matrice N est $126+30 = 156$, le nombre b diviseur initial. On peut également obtenir les nombres $b = 156$ et $a = 385$ en multipliant à droite la

séquence de matrices de la forme $\begin{pmatrix} q_i & r_i \\ 1 & 0 \end{pmatrix}$ par la matrice $\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$ (séquence complète pour obtenir a, séquence sans son premier élément pour obtenir b).

On préfère cependant la formulation matricelle trouvée dans [5], qui bien que faisant intervenir des matrices de 2 tailles différentes (des 2×2, et des 2×3) fait voir plus aisément les valeurs des variables calculées. Les matrices 2×2 sont de la forme $\begin{pmatrix} 0 & 1 \\ 1 & -q_i \end{pmatrix}$ tandis que les matrices 2×3 sont de la forme $\begin{pmatrix} a_i & x_i & y_i \\ b_i & x_{i+1} & y_{y+1} \end{pmatrix}$ ainsi :

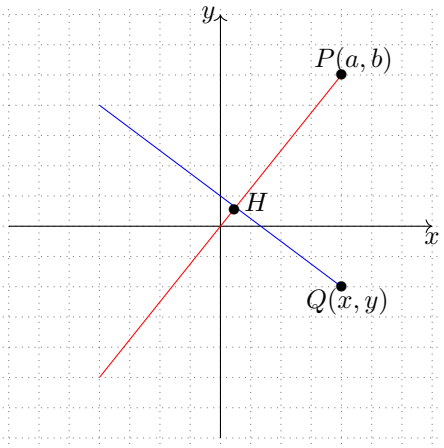
$$\begin{aligned}
&\begin{pmatrix} 0 & 1 \\ 1 & -3 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -3 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -7 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -2 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -2 \end{pmatrix}\begin{pmatrix} 385 & 1 & 0 \\ 156 & 0 & 1 \end{pmatrix} \\
&= \begin{pmatrix} 0 & 1 \\ 1 & -3 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -3 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -7 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -2 \end{pmatrix}\begin{pmatrix} 156 & 0 & 1 \\ 73 & 1 & -2 \end{pmatrix} \\
&= \begin{pmatrix} 0 & 1 \\ 1 & -3 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -3 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -7 \end{pmatrix}\begin{pmatrix} 73 & 1 & -2 \\ 10 & -2 & 5 \end{pmatrix} \\
&= \begin{pmatrix} 0 & 1 \\ 1 & -3 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -3 \end{pmatrix}\begin{pmatrix} 10 & -2 & 5 \\ 3 & 15 & -37 \end{pmatrix} \\
&= \begin{pmatrix} 0 & 1 \\ 1 & -3 \end{pmatrix}\begin{pmatrix} 3 & 15 & -37 \\ 1 & -47 & 116 \end{pmatrix} \\
&= \begin{pmatrix} 1 & -47 & 116 \\ 0 & 156 & 385 \end{pmatrix}
\end{aligned}$$

Remarque : On a à tout étape de l'algorithme respect de la relation invariante habituelle $a = bq + r$ mais également dans cette deuxième version $x_i y_{i+1} - x_{i+1} y_i = (-1)^i$.

3) Représentation géométrique pour l'algorithme d'Euclide étendu

Représentons ci-dessous le fait que l'algorithme étendu permette de trouver les coefficients $x = 4$ et $y = -3$ pour les valeurs fournies $a = 4$ et $b = 5$, qui permettent que l'identité de Bézout $4 \times 4 - 3 \times 5 = 1$ soit vérifiée. Le point $Q\begin{pmatrix} x \\ y \end{pmatrix}$ est sur la droite perpendiculaire à la droite (OP) qui passe par le point P' avec $\overrightarrow{OP'} = \dfrac{\overrightarrow{OP}}{OP^2}$.

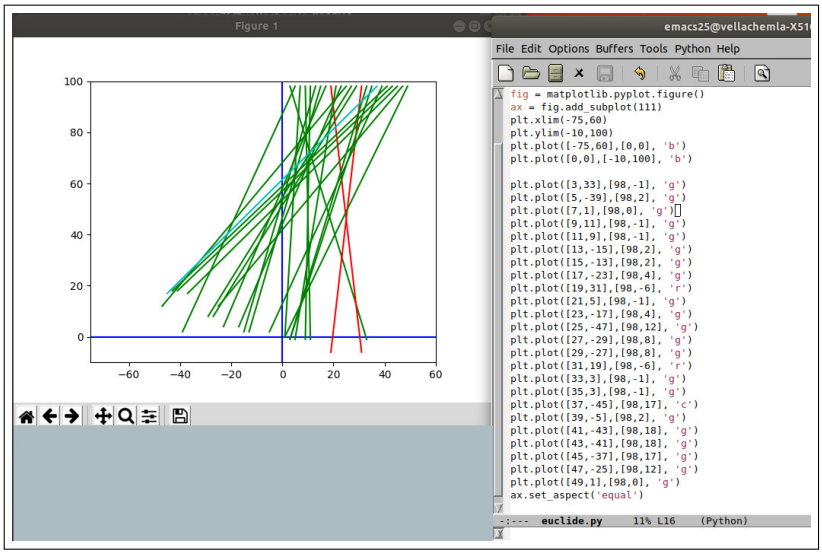
Il faudrait réfléchir à la possibilité que cette représentation géométrique puisse peut-être permettre de "voir" les décomposants de Goldbach *directement*, par opposition à une méthode de visualisation comme celle qu'on a proposée par exemple en [5] et qui nécessite de cribler les multiples.



4) Présenter graphiquement les points trouvés par l'algorithme d'Euclide étendu dans un seul cas particulier

Pour cependant visualiser à la fois les résultats de l'algorithme d'Euclide étendu et les décomposants de Goldbach des nombres pairs compris entre 6 et 100, on décide de dessiner des sortes de spaghetti (fournis en annexe). Voici le résultat d'une telle visualisation pour le nombre pair 98, pour lequel l'algorithme d'Euclide étendu fournit :

$(3, 98) = (33, -1)$	$(95, 98) = (-33, 32)$	
$(5, 98) = (-39, 2)$	$(93, 98) = (39, -37)$	
$(7, 98) = (1, 0)$	$(91, 98) = (-1, 1)$	
$(9, 98) = (11, -1)$	$(89, 98) = (-11, 10)$	
$(11, 98) = (9, -1)$	$(87, 98) = (-9, 8)$	
$(13, 98) = (-15, 2)$	$(85, 98) = (15, -13)$	
$(15, 98) = (-13, 2)$	$(83, 98) = (13, -11)$	
$(17, 98) = (-23, 4)$	$(81, 98) = (23, -19)$	
$(19, 98) = (31, -6)$	$(79, 98) = (-31, 25.)$	Décomp. de Goldbach.
$(21, 98) = (5, -1)$	$(77, 98) = (-5, 4)$	
$(23, 98) = (-17, 4)$	$(75, 98) = (17, -13)$	
$(25, 98) = (-47, 12)$	$(73, 98) = (47, -35)$	
$(27, 98) = (-29, 8)$	$(71, 98) = (29, -21)$	
$(29, 98) = (-27, 8)$	$(69, 98) = (27, -19)$	
$(31, 98) = (19, -6)$	$(67, 98) = (-19, 13.)$	Décomp. de Goldbach.
$(33, 98) = (3, -1)$	$(65, 98) = (-3, 2)$	
$(35, 98) = (3, -1)$	$(63, 98) = (-3, 2)$	
$(37, 98) = (-45, 17)$	$(61, 98) = (45, -28.)$	Décomp. de Goldbach.
$(39, 98) = (-5, 2)$	$(59, 98) = (5, -3)$	
$(41, 98) = (-43, 18)$	$(57, 98) = (43, -25)$	
$(43, 98) = (-41, 18)$	$(55, 98) = (41, -23)$	
$(45, 98) = (-37, 17)$	$(53, 98) = (37, -20)$	
$(47, 98) = (-25, 12)$	$(51, 98) = (25, -13)$	
$(49, 98) = (1, 0)$	$(49, 98) = (1, 0)$	



On a noté les décomposants de Goldbach 19 et 31 qui sont comme symétriques en rouge (et le troisième décomposant 37 en cyan), les autres impairs inférieurs à 49 la moitié de 98 en vert.
Les "spaghetti" des décomposants de Goldbach descendent plus bas que les autres ; est-ce un hasard ?...

Bibliographie

[1] Connes, Alain, *A new proof of Morley's theorem*, Publications Mathématiques de l'IHÉS, 1998, S88 : 43-46. (traduction : http://denisevellachemla.eu/Alain-Connes-Theoreme-Morley.pdf)

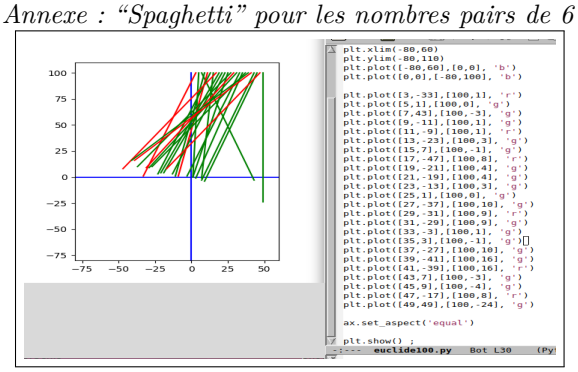
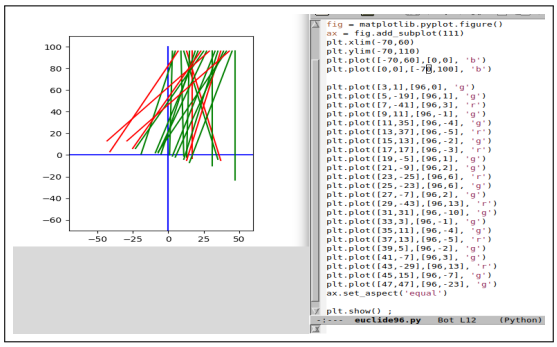
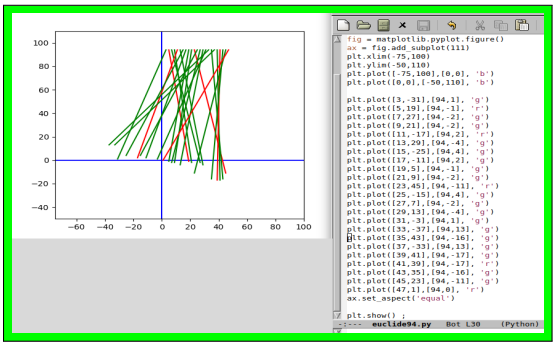
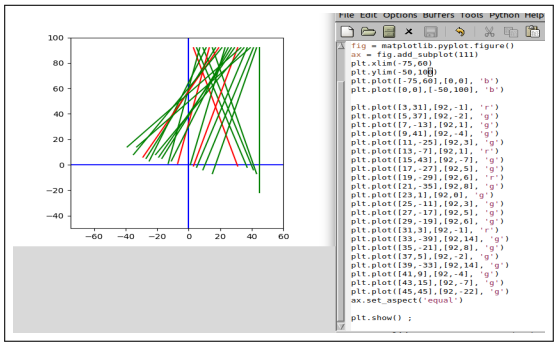
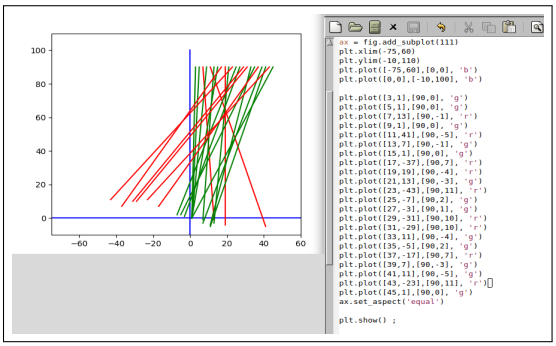
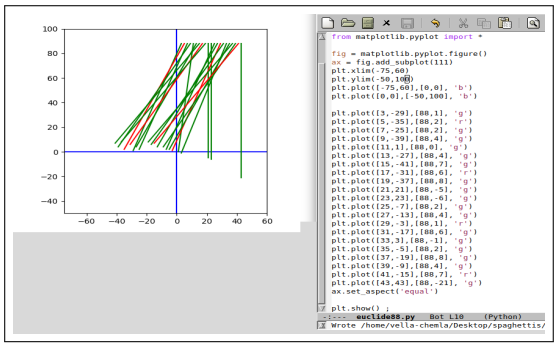
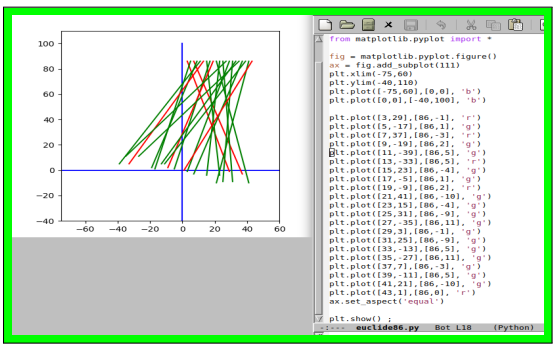
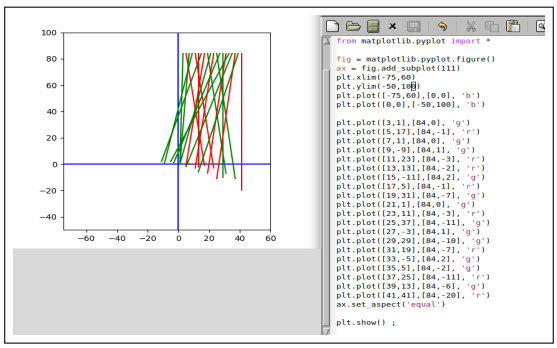
[2] Connes, Alain, *Noncommutative Geometry and the Riemann zeta function*, Mathematics : Frontiers and Perspectives 2000. IMU, AMS, V. Arnold, M. Atiyah, P. Lax, B. Mazur Editors, 2000, p. 35-55.

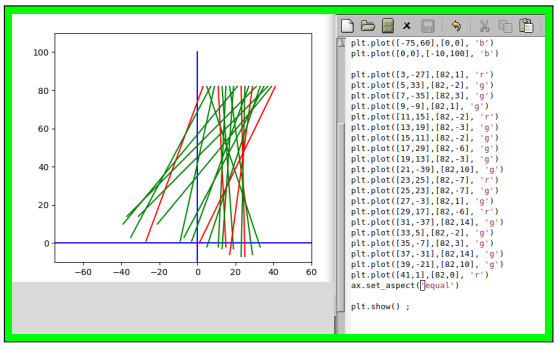
[3] *L'identité de Bézout*, dans Histoire d'algorithmes, du caillou à la puce, Jean-Luc Chabert, Évelyne Barbin, Michel Guillemot, Anne Michel-Pajus, Jacques Borowczyk, Ahmed Djebbar, Jean-Claude Martzloff, éditions Belin, Collection Regards sur la Science, 1994, p. 139.

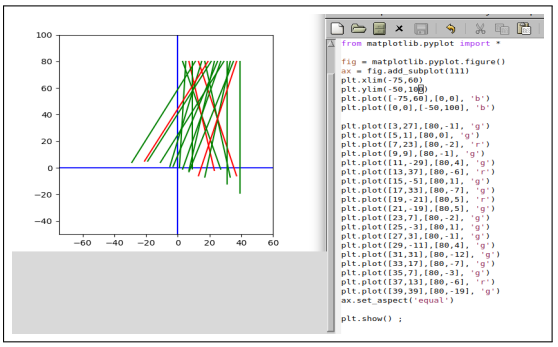
[4] Demazure, Michel, Cours d'Algèbre, primalité, divisibilité, codes„ Nouvelle bibliothèque mathématique, éd. Cassini, 1997.

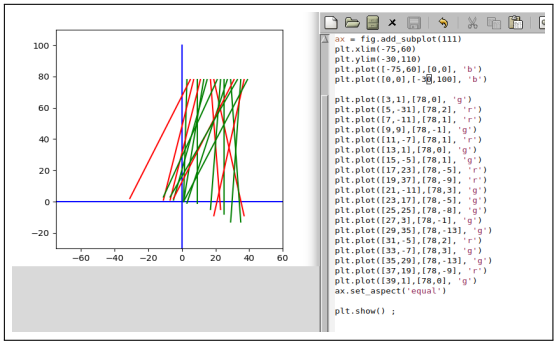
[5] Vella-Chemla, Denise, http://denisevellachemla.eu/concentriques400.pdf.

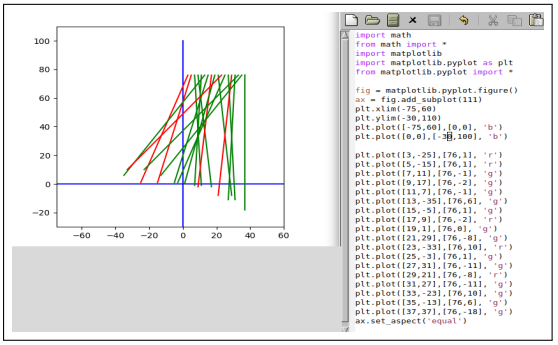
Annexe : "Spaghetti" pour les nombres pairs de 6 à 100

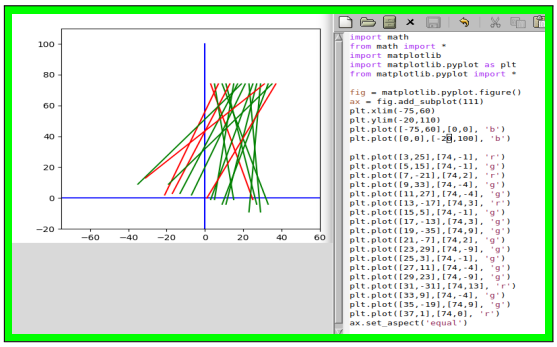









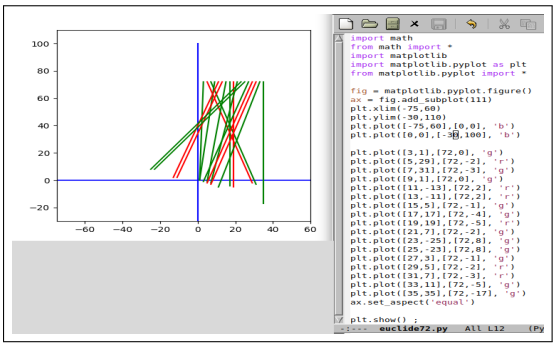


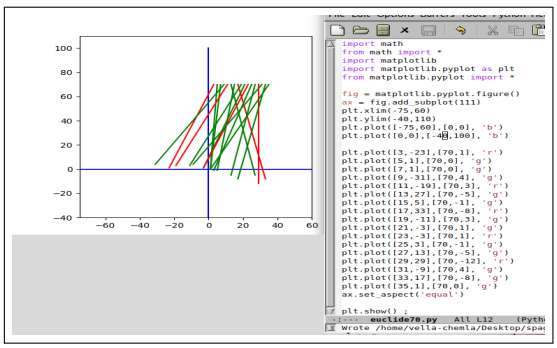


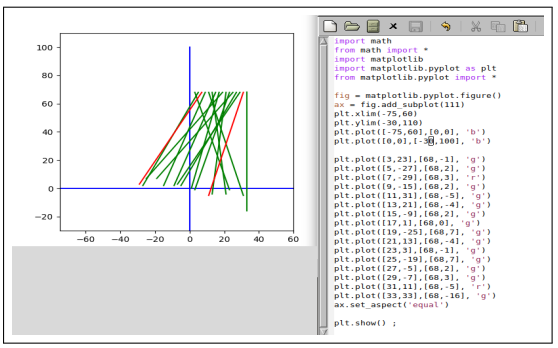


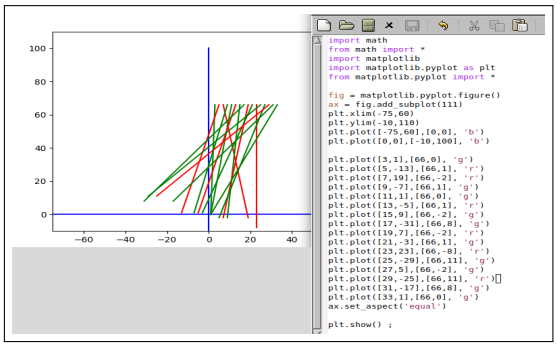


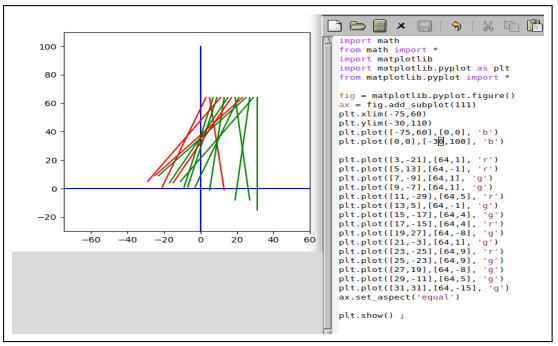


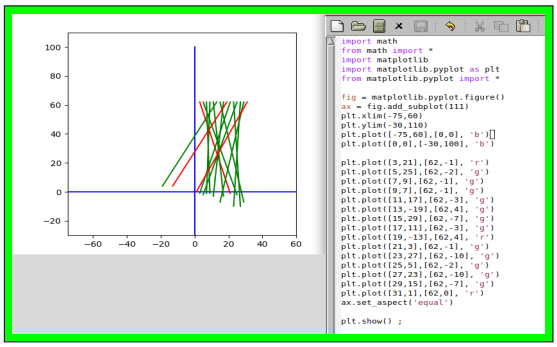




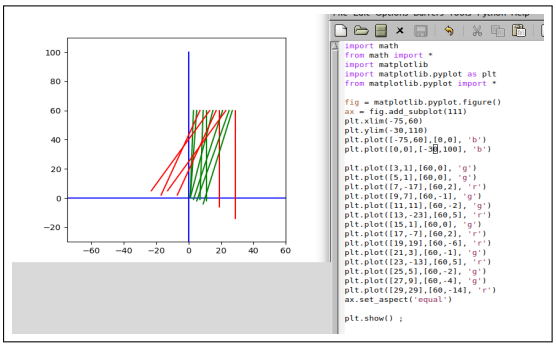




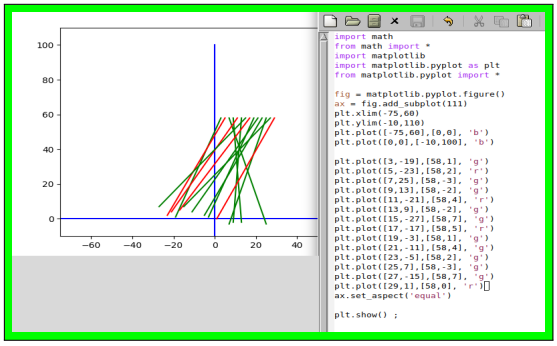




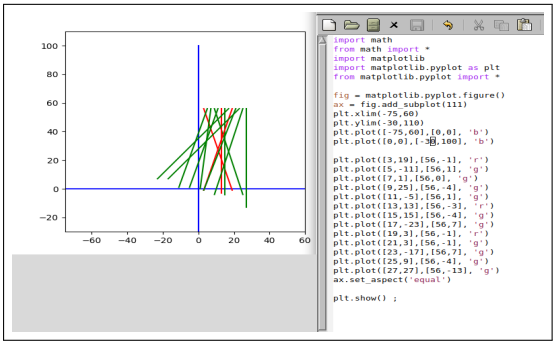
```
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *

fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-30,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-30,100], 'b')

plt.plot([3,21],[62,-1), 'r')
plt.plot([5,25],[62,-2], 'g')
plt.plot([7,9],[62,-1], 'g')
plt.plot([9,7],[62,-1), 'g')
plt.plot([11,17],[62,-3], 'g')
plt.plot([13,-19],[62,4], 'g')
plt.plot([15,29],[62,-7], 'g')
plt.plot([17,11],[62,-3], 'g')
plt.plot([19,-13],[62,4], 'r')
plt.plot([21,3],[62,-1], 'g')
plt.plot([23,27],[62,-10], 'g')
plt.plot([25,5],[62,-2], 'g')
plt.plot([27,23],[62,-10], 'g')
plt.plot([29,15],[62,-7], 'g')
plt.plot([31,1],[62,0], 'r')
ax.set_aspect('equal')

plt.show() ;
```



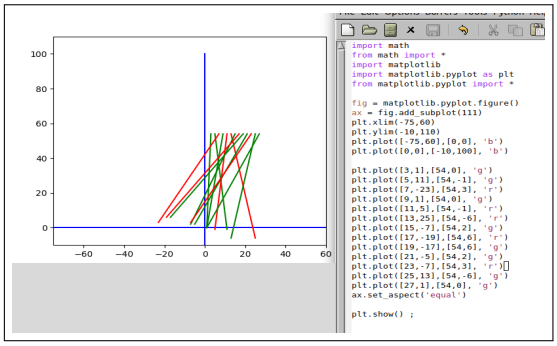
```
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *

fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-30,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-30,100], 'b')

plt.plot([3,1],[60,0], 'g')
plt.plot([5,1],[60,0], 'g')
plt.plot([7,-17],[60,2], 'r')
plt.plot([9,7],[60,-1], 'g')
plt.plot([11,11],[60,-2], 'g')
plt.plot([13,-23],[60,5], 'r')
plt.plot([15,1],[60,0], 'g')
plt.plot([17,-7],[60,2], 'r')
plt.plot([19,19],[60,-6], 'r')
plt.plot([21,3],[60,-1], 'g')
plt.plot([23,-13],[60,5], 'r')
plt.plot([25,5],[60,-2], 'g')
plt.plot([27,9],[60,-4], 'g')
plt.plot([29,29],[60,-14], 'r')
ax.set_aspect('equal')

plt.show() ;
```



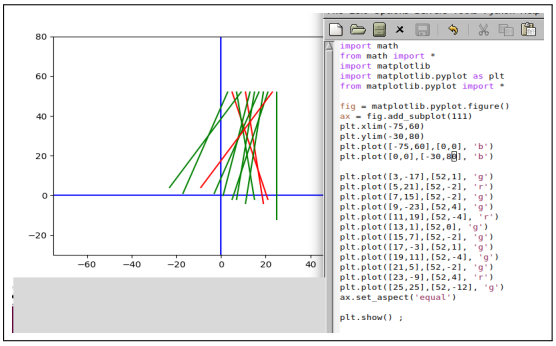
```
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *

fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-10,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-10,100], 'b')

plt.plot([3,-19],[58,1], 'g')
plt.plot([5,-23],[58,21, 'r')
plt.plot([7,25],[58,-3], 'g')
plt.plot([9,13],[58,-2], 'g')
plt.plot([11,-21],[58,4], 'r')
plt.plot([13,9],[58,-2], 'g')
plt.plot([15,-27],[58,7], 'g')
plt.plot([17,-17],[58,5], 'r')
plt.plot([19,-3],[58,11], 'g')
plt.plot([21,-11],[58,4], 'g')
plt.plot([23,-5],[58,2], 'g')
plt.plot([25,7],[58,-3], 'g')
plt.plot([27,-15],[58,7], 'g')
plt.plot([29,1],[58,0], 'r')
ax.set_aspect('equal')

plt.show() ;
```



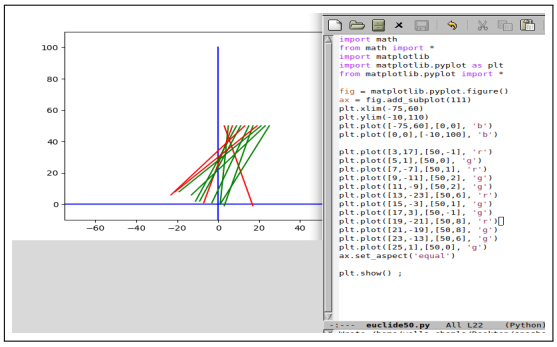
```
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *

fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-30,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-30,100], 'b')

plt.plot([3,19],[56,-1], 'r')
plt.plot([5,-11],[56,1], 'g')
plt.plot([7,11],[56,0], 'g')
plt.plot([9,25],[56,-4], 'g')
plt.plot([11,-5],[56,1], 'g')
plt.plot([13,13],[56,-3], 'r')
plt.plot([15,15],[56,-4], 'g')
plt.plot([17,-23],[56,7], 'g')
plt.plot([19,3],[56,-1], 'r')
plt.plot([21,3],[56,-1], 'g')
plt.plot([23,-17],[56,7], 'g')
plt.plot([25,9],[56,-4], 'g')
plt.plot([27,27],[56,-13], 'g')
ax.set_aspect('equal')

plt.show() ;
```



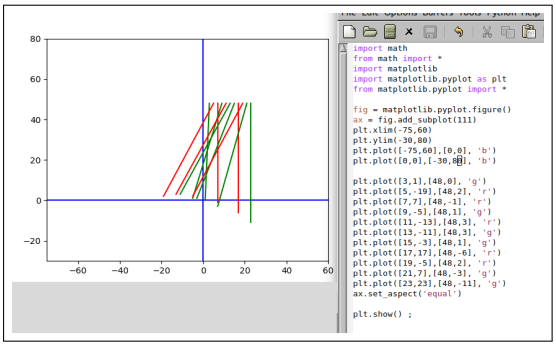
```
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *

fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-10,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-10,100], 'b')

plt.plot([3,1],[54,0], 'g')
plt.plot([5,111],[54,-1], 'g')
plt.plot([7,-23],[54,3], 'r')
plt.plot([9,1],[54,0], 'g')
plt.plot([11,5],[54,-1], 'r')
plt.plot([13,25],[54,-6], 'r')
plt.plot([15,-7],[54,2], 'g')
plt.plot([17,-19],[54,6], 'r')
plt.plot([19,-17],[54,6], 'g')
plt.plot([21,-5],[54,2], 'g')
plt.plot([23,-7],[54,3], 'r')
plt.plot([25,13],[54,-6], 'g')
plt.plot([27,1],[54,0], 'g')
ax.set_aspect('equal')

plt.show() ;
```



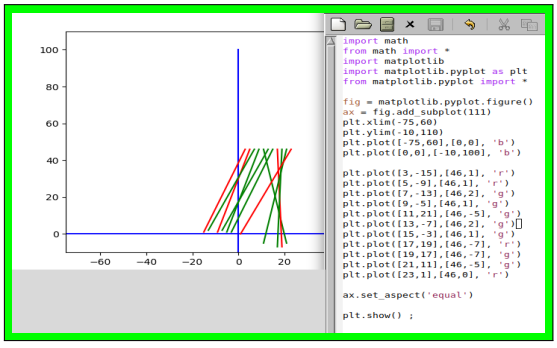
```
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *

fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-30,80)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-30,80], 'b')

plt.plot([3,-17],[52,1], 'g')
plt.plot([5,21],[52,-2], 'r')
plt.plot([7,15],[52,-2], 'g')
plt.plot([9,-23],[52,4], 'g')
plt.plot([11,19],[52,-4], 'r')
plt.plot([13,1],[52,0], 'g')
plt.plot([15,7],[52,-2], 'g')
plt.plot([17,-3],[52,1], 'g')
plt.plot([19,11],[52,-4], 'g')
plt.plot([21,5],[52,-2], 'g')
plt.plot([23,-9],[52,4], 'r')
plt.plot([25,25],[52,-12], 'g')
ax.set_aspect('equal')

plt.show() ;
```



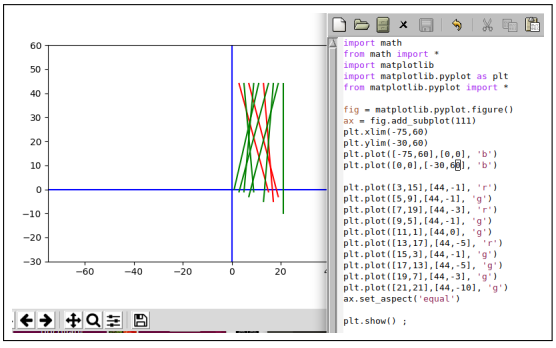
```
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *

fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-10,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-10,100], 'b')

plt.plot([3,17],[50,-1], 'r')
plt.plot([5,1],[50,0], 'g')
plt.plot([7,-7],[50,1], 'r')
plt.plot([9,-11],[50,2], 'g')
plt.plot([11,-9],[50,2], 'g')
plt.plot([13,-23],[50,6], 'r')
plt.plot([15,-3],[50,1], 'g')
plt.plot([17,3],[50,-1], 'g')
plt.plot([19,-21],[50,6], 'r')
plt.plot([21,-19],[50,8], 'g')
plt.plot([23,-13],[50,6], 'g')
plt.plot([25,1],[50,0], 'g')
ax.set_aspect('equal')

plt.show() ;
```



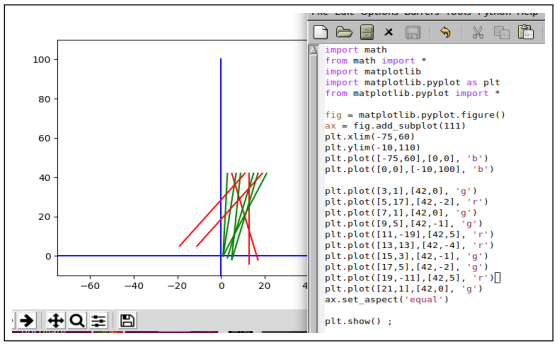
```
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *

fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-30,80)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-30,80], 'b')

plt.plot([3,1],[48,0], 'g')
plt.plot([5,-19],[48,2], 'r')
plt.plot([7,7],[48,-1], 'r')
plt.plot([9,-5],[48,1], 'g')
plt.plot([11,-13],[48,3], 'r')
plt.plot([13,-11],[48,3], 'g')
plt.plot([15,-3],[48,1], 'g')
plt.plot([17,17],[48,-6], 'r')
plt.plot([19,-5],[48,2], 'r')
plt.plot([21,7],[48,-3], 'g')
plt.plot([23,23],[48,-11], 'g')
ax.set_aspect('equal')

plt.show() ;
```

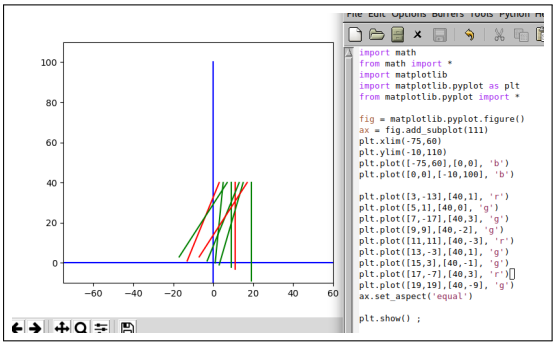


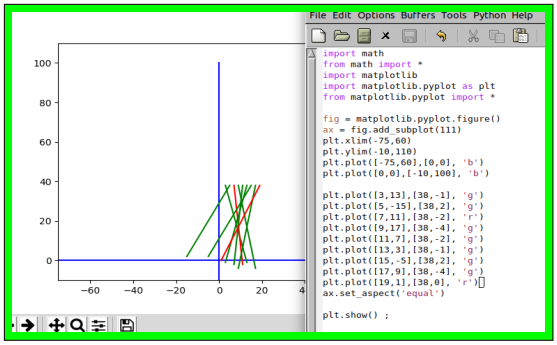
```
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *

fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-10,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-10,100], 'b')

plt.plot([3,-15],[46,1], 'r')
plt.plot([5,-9],[46,1], 'r')
plt.plot([7,-13],[46,2], 'g')
plt.plot([9,-5],[46,1], 'g')
plt.plot([11,21],[46,-5], 'g')
plt.plot([13,-7],[46,2], 'g')
plt.plot([15,-3],[46,1], 'g')
plt.plot([17,19],[46,-7], 'r')
plt.plot([19,17],[46,-7], 'g')
plt.plot([21,11],[46,-5], 'g')
plt.plot([23,1],[46,0], 'r')

ax.set_aspect('equal')

plt.show() ;
```

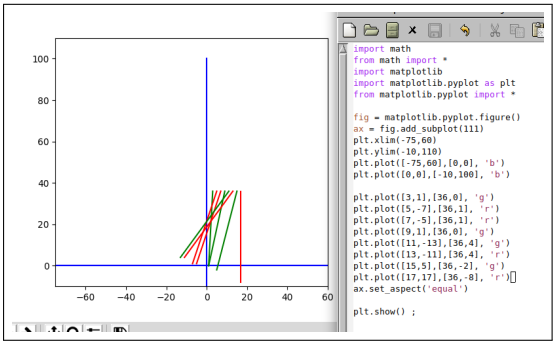


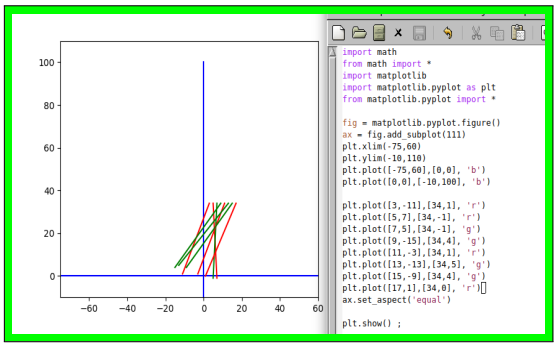
```
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *

fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-30,60)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-30,60], 'b')

plt.plot([3,15],[44,-1], 'r')
plt.plot([5,9],[44,-1], 'g')
plt.plot([7,19],[44,-3], 'r')
plt.plot([9,5],[44,-1], 'g')
plt.plot([11,1],[44,0], 'g')
plt.plot([13,17],[44,-5], 'r')
plt.plot([15,3],[44,-1], 'g')
plt.plot([17,13],[44,-5], 'g')
plt.plot([19,7],[44,-3], 'g')
plt.plot([21,21],[44,-10], 'g')
ax.set_aspect('equal')

plt.show() ;
```

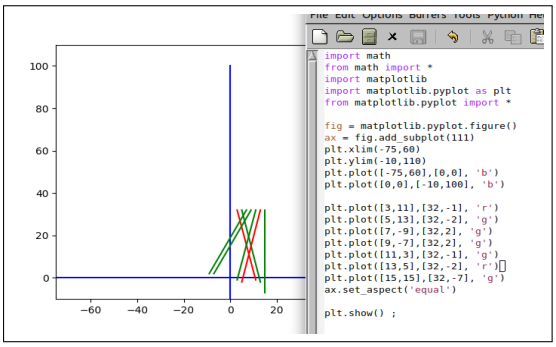

import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *
fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-10,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-10,100], 'b')
plt.plot([3,1],[42,0], 'g')
plt.plot([5,17],[42,-2], 'r')
plt.plot([7,1],[42,0], 'g')
plt.plot([9,5],[42,-1], 'g')
plt.plot([11,-19],[42,5], 'r')
plt.plot([13,13],[42,-4], 'r')
plt.plot([15,3],[42,-1], 'g')
plt.plot([17,5],[42,-2], 'g')
plt.plot([19,-11],[42,5], 'r')
plt.plot([21,1],[42,0], 'g')
ax.set_aspect('equal')
plt.show() ;

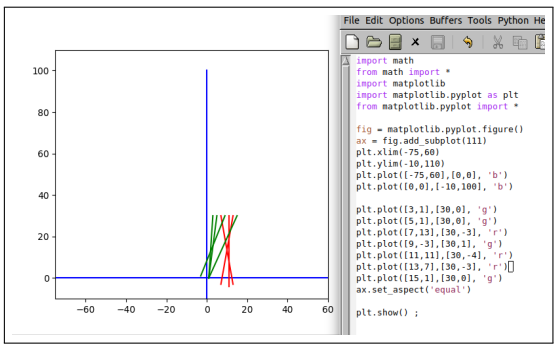

import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *
fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-10,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-10,100], 'b')
plt.plot([3,-13],[40,1], 'r')
plt.plot([5,1],[40,0], 'g')
plt.plot([7,-17],[40,3], 'g')
plt.plot([9,9],[40,-2], 'g')
plt.plot([11,11],[40,-3], 'r')
plt.plot([13,-3],[40,1], 'g')
plt.plot([15,3],[40,-1], 'g')
plt.plot([17,-7],[40,3], 'r')
plt.plot([19,19],[40,-9], 'g')
ax.set_aspect('equal')
plt.show() ;

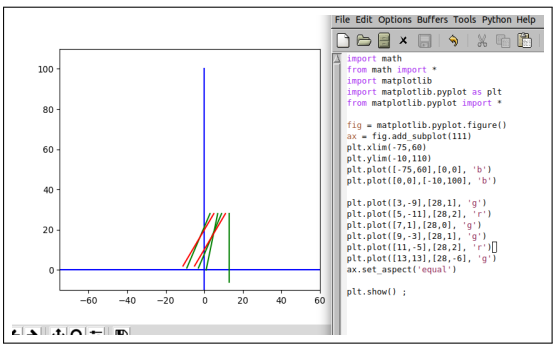

File Edit Options Buffers Tools Python Help
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *
fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-10,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-10,100], 'b')
plt.plot([3,13],[38,-1], 'g')
plt.plot([5,-15],[38,2], 'g')
plt.plot([7,11],[38,-2], 'r')
plt.plot([9,17],[38,-4], 'g')
plt.plot([11,7],[38,-2], 'g')
plt.plot([13,3],[38,-1], 'g')
plt.plot([15,-5],[38,2], 'g')
plt.plot([17,9],[38,-4], 'g')
plt.plot([19,1],[38,0], 'r')
ax.set_aspect('equal')
plt.show() ;

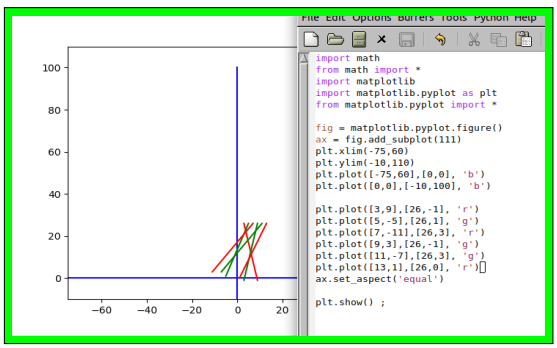

import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *
fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-10,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-10,100], 'b')
plt.plot([3,1],[36,0], 'g')
plt.plot([5,-7],[36,1], 'r')
plt.plot([7,-5],[36,1], 'r')
plt.plot([9,1],[36,0], 'g')
plt.plot([11,-13],[36,4], 'g')
plt.plot([13,-11],[36,4], 'r')
plt.plot([15,5],[36,-2], 'g')
plt.plot([17,17],[36,-8], 'r')
ax.set_aspect('equal')
plt.show() ;

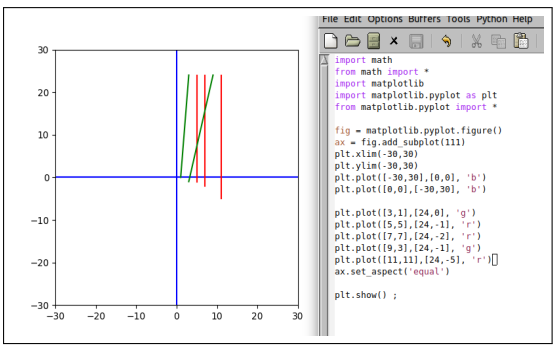

import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *
fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-10,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-10,100], 'b')
plt.plot([3,-11],[34,1], 'r')
plt.plot([5,7],[34,-1], 'r')
plt.plot([7,5],[34,-1], 'g')
plt.plot([9,-15],[34,4], 'g')
plt.plot([11,-3],[34,1], 'r')
plt.plot([13,-13],[34,5], 'g')
plt.plot([15,-9],[34,4], 'g')
plt.plot([17,1],[34,0], 'r')
ax.set_aspect('equal')
plt.show() ;

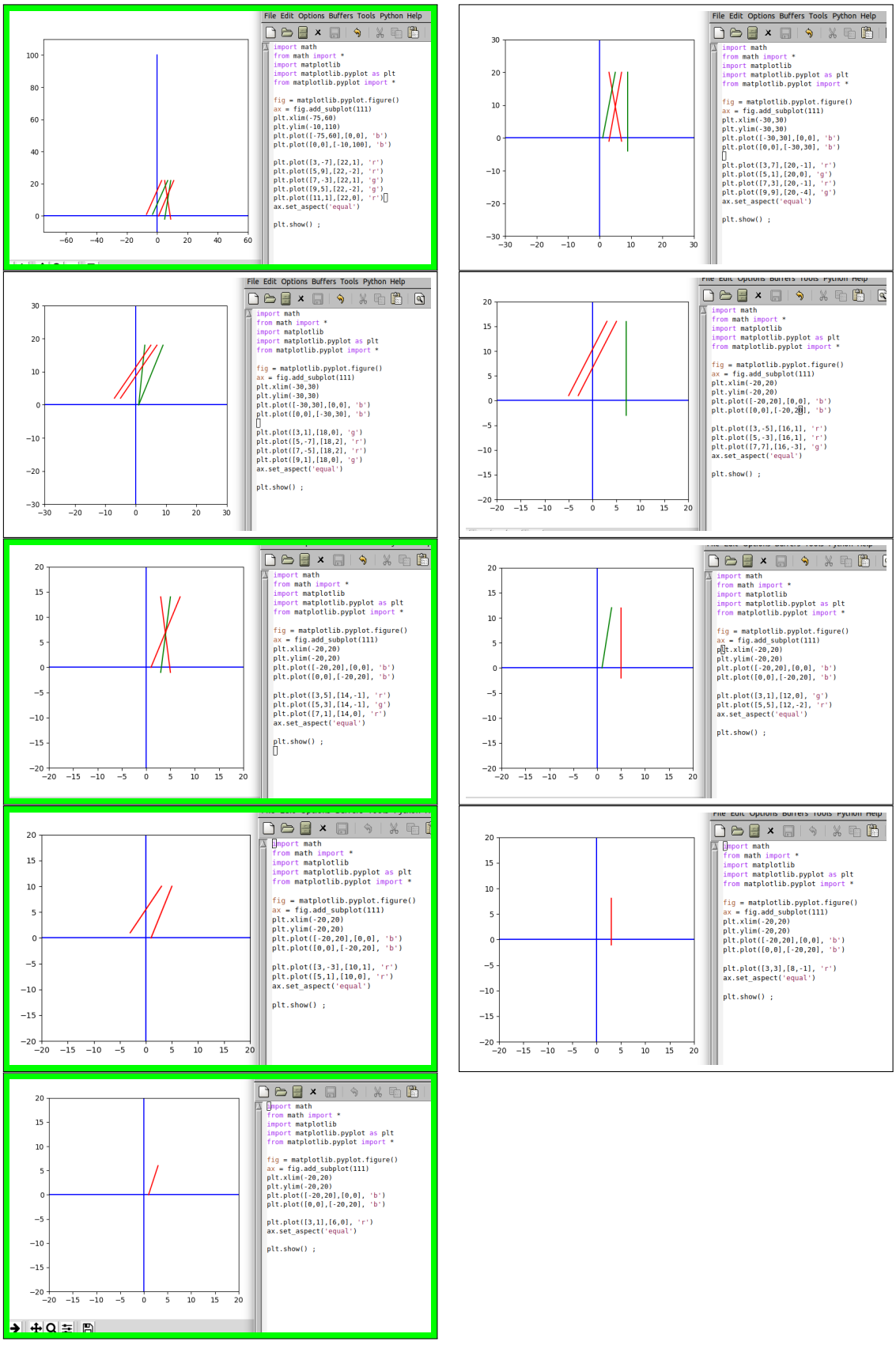

import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *
fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-10,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-10,100], 'b')
plt.plot([3,11],[32,-1], 'r')
plt.plot([5,13],[32,-2], 'g')
plt.plot([7,-9],[32,2], 'g')
plt.plot([9,-7],[32,2], 'g')
plt.plot([11,3],[32,-1], 'g')
plt.plot([13,5],[32,-2], 'r')
plt.plot([15,15],[32,-7], 'g')
ax.set_aspect('equal')
plt.show() ;


File Edit Options Buffers Tools Python He
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *
fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-10,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-10,100], 'b')
plt.plot([3,1],[30,0], 'g')
plt.plot([5,1],[30,0], 'g')
plt.plot([7,13],[30,-3], 'r')
plt.plot([9,-3],[30,1], 'g')
plt.plot([11,11],[30,-4], 'r')
plt.plot([13,7],[30,-3], 'r')
plt.plot([15,1],[30,0], 'g')
ax.set_aspect('equal')
plt.show() ;


File Edit Options Buffers Tools Python Help
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *
fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-10,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-10,100], 'b')
plt.plot([3,-9],[28,1], 'g')
plt.plot([5,-11],[28,2], 'r')
plt.plot([7,1],[28,0], 'g')
plt.plot([9,-3],[28,1], 'g')
plt.plot([11,-5],[28,2], 'r')
plt.plot([13,13],[28,-6], 'g')
ax.set_aspect('equal')
plt.show() ;


File Edit Options Buffers Tools Python Help
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *
fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-75,60)
plt.ylim(-10,110)
plt.plot([-75,60],[0,0], 'b')
plt.plot([0,0],[-10,100], 'b')
plt.plot([3,9],[26,-1], 'r')
plt.plot([5,-5],[26,1], 'g')
plt.plot([7,-11],[26,3], 'r')
plt.plot([9,3],[26,-1], 'g')
plt.plot([11,-7],[26,3], 'g')
plt.plot([13,1],[26,0], 'r')
ax.set_aspect('equal')
plt.show() ;


File Edit Options Buffers Tools Python Help
import math
from math import *
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import *
fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
plt.xlim(-30,30)
plt.ylim(-30,30)
plt.plot([-30,30],[0,0], 'b')
plt.plot([0,0],[-30,30], 'b')
plt.plot([3,1],[24,0], 'g')
plt.plot([5,5],[24,-1], 'r')
plt.plot([7,7],[24,-2], 'r')
plt.plot([9,3],[24,-1], 'g')
plt.plot([11,11],[24,-5], 'r')
ax.set_aspect('equal')
plt.show() ;



On observe pour les doubles de premiers (6, 10, 14, 22, 26, 34, 38, 46, 58, 62, 74, 82, 86, 94, leur graphique est entouré en vert) la décomposition de Goldbach triviale qui relie le point tout en haut tout à droite au point (1,0). À part le côté marrant de 72 avec toutes ses parallèles, on n'observe rien de probant, c'est un nouveau coup d'épée dans l'eau.

Calculs au sujet de la conjecture de Goldbach à l'aide du logiciel gb-tools (Denise Vella-Chemla, 31.8.2020)

Sont fournis ici des extraits des résultats de l'exécution d'un programme calculant le nombre de décompositions de Goldbach des nombres pairs inférieurs à 10^9 ; ce programme fait partie d'une suite logicielle *gb-tools* qui a été écrite par Daniel Diaz. Le nombre de décompositions de Goldbach d'un nombre pair est le nombre de façons différentes de l'écrire comme une somme de deux nombres premiers impairs. La suite logicielle *gb-tools* nous avait également permis en décembre 2010 d'étudier le positionnement de certains nombres de décompositions de Goldbach dans la fameuse "comète de Goldbach", ce positionnement dépendant des factorisations des nombres pairs considérés (cf [2]).

Est présentée ci-dessous une comparaison des résultats informatiques obtenus par programme avec les résultats mathématiques heuristiques proposés dans la littérature. Landau a proposé comme estimation de $G(n)$, le nombre de décompositions de Goldbach, la formule

$$\frac{n}{2\ (\ln n)^2}$$

(qu'on note $L(n)$ en troisième colonne dans les différentes parties du tableau) et Hardy et Littlewood ont proposé la formule

$$2\ C^{\text{te}} \frac{n}{(\ln n)^2} \prod_{\substack{p\ premier,\\ p|n,\\ p\geq 3}} \frac{p-1}{p-2}$$

(cf [1] p. 32, cette formule est notée $HL(n)$, en quatrième colonne dans les différentes parties du tableau). C^{te} est approximativement égale à 0.660175 [1].

n	G(n)	L(n)	HL(n)	2p	n	G(n)	L(n)	HL(n)	2p
92	4	2.24	5.94		992	13	10.41	28.46	
94	5	2.27	6.01	×	994	25	10.43	33.06	
96	7	2.30	12.17		996	37	10.44	55.18	
98	3	2.33	7.39		998	17	10.46	27.63	×
10^2	6	2.35	8.30		10^3	28	10.47	36.89	
102	8	2.38	12.59		1002	36	10.49	55.42	
104	5	2.41	6.37		1004	18	10.50	27.75	
106	6	2.43	6.44	×	1006	18	10.52	27.79	×
108	8	2.46	13.01		1008	42	10.53	66.78	
110	6	2.48	8.76		1010	25	10.55	37.15	

n	G(n)	L(n)	HL(n)	2p	n	G(n)	L(n)	HL(n)	2p
9992	102	58.90	155.55		99 992	638	377.19	1032.94	
9994	98	58.91	164.72		99 994	651	377.20	1068.68	
9996	255	58.92	398.32		99 996	1303	377.21	2173.27	
9998	99	58.93	155.62	×	99 998	605	377.21	996.10	×
10^4	127	58.94	207.52		10^5	810	377.22	1328.15	
10 002	197	58.95	311.33		100 002	1423	377.22	2390.71	
10 004	99	58.95	162.39		100 004	627	377.23	1043.58	
10 006	92	58.96	155.72	×	100 006	630	377.24	1030.51	
10 008	192	58.97	311.48		100 008	1209	377.24	1992.36	
10 010	191	58.98	302.09		100 010	831	377.25	1356.95	

1. voir http://denisevellachemla.eu/Hardy-Littlewood-p1.jpg page extraite de [1] dans laquelle est fournie la formule de calcul de la constante.

n	G(n)	L(n)	HL(n)	2p	n	G(n)	L(n)	HL(n)	2p
999 992	4858	2619.59	8300.89		9 999 992	29 047	19 246.07	50 822.10	
999 994	4235	2619.59	7246.82		9 999 994	29 790	19 246.08	52 061.67	
999 996	8194	2619.60	13 946.72		9 999 996	58 553	19 246.08	102 182.04	
999 998	4206	2619.60	7213.23		9 999 998	28 983	19 246.08	50 822.13	×
10^6	5402	2619.61	9223.28		10^7	38 807	19 246.09	67 762.86	
1 000 002	8200	2619.61	13 834.94		10 000 002	59 624	19 246.09	103 903.07	
1 000 004	4160	2619.62	7134.19		10 000 004	36 850	19 246.09	64 574.04	
1 000 006	4871	2619.62	8300.99		10 000 006	29 835	19 246.10	52 032.19	
1 000 008	9380	2619.62	16 006.53		10 000 008	58 229	19 246.10	101 644.34	
1 000 010	5951	2619.63	10 248.17		10 000 010	39 045	19 246.10	68 447.38	

n	G(n)	L(n)	HL(n)	2p	n	G(n)	L(n)	HL(n)	2p
99 999 992	218 826	147 352.86	389 106.94		999 999 992	1 706 569	1 164 269.68	3 080 611.25	
99 999 994	218 773	147 352.86	389 436.97		999 999 994	2 044 282	1 164 269.68	3 689 310.75	
99 999 996	437 175	147 352.87	778 837.19		999 999 996	3 407 072	1 164 269.68	6 148 851.00	
99 999 998	274 787	147 352.87	489 163.06		999 999 998	1 705 026	1 164 269.68	3 078 887.75	
10^8	291 400	147 352.87	518 809.31		10^9	2 274 205	1 164 269.68	4 099 234.25	
100 000 002	464 621	147 352.87	825 805.69		1 000 000 002	3 496 205	1 164 269.68	6 306 844.00	
100 000 004	247 582	147 352.87	440 288.19		1 000 000 004	1 747 858	1 164 269.68	3 153 257.25	
100 000 006	218 966	147 352.88	389 902.75		1 000 000 006	1 704 301	1 164 269.68	3 074 425.50	×
100 000 008	437 717	147 352.88	778 214.00		1 000 000 008	4 151 660	1 164 269.68	7 492 717.50	
100 000 010	323 687	147 352.88	576 454.88		1 000 000 010	2 422 662	1 164 269.68	4 372 516.50	

On propose notre propre formule basée sur la démonstration proposée en [3] (la démonstration a été rédigée formellement par Leila Schneps). On ne peut fournir d'explication pour la division par la racine du logarithme, pourtant les résultats fournis par programme, dans les tableaux ci-dessous, semblent de plus en plus proches des nombres de décomposants de Goldbach, et dans la dernière partie du tableau, pour n proche de 10^9 du moins, les valeurs minorent les nombres de décompositions de Goldbach.

$$D(n) = \frac{n}{\sqrt{\ln n}} \prod_{\substack{p\ premier,\\ p|n,\\ 3\leq p\leq\sqrt{n}}} \left(1-\frac{1}{p}\right) \prod_{\substack{p\ premier,\\ p\nmid n,\\ 3\leq p\leq\sqrt{n}}} \left(1-\frac{2}{p}\right)$$

Note : Le signe $\nmid$ sous le $\prod$ du deuxième produit signifie "ne divise pas". Il est plutôt insuffisamment barré.

n	G(n)	D(n)	n	G(n)	D(n)
92	4	6.18	992	13	24.25
94	5	6.30	994	25	28.19
96	7	12.83	996	37	47.07
98	3	7.84	998	17	23.58
10^2	6	8.87	10^3	28	31.49
102	8	13.55	1002	36	47.33
104	5	6.89	1004	18	23.71
106	6	7.01	1006	18	23.75
108	8	14.26	1008	42	57.11
110	6	9.66	1010	25	31.79

n	G(n)	D(n)	n	G(n)	D(n)
9992	102	126.09	99 992	638	752.15
9994	98	133.53	99 994	651	778.18
9996	255	322.92	99 996	1303	1582.51
9998	99	126.16	99 998	605	725.33
10^4	127	168.25	10^5	810	967.13
10 002	197	252.42	100 002	1423	1740.87
10 004	99	131.66	100 004	627	759.91
10 006	92	126.26	100 006	630	750.40
10 008	192	252.57	100 008	1209	1450.80
10 010	191	244.96	100 010	831	988.11

n	G(n)	D(n)	n	G(n)	D(n)
999 992	4 858	5 589.30	9 999 992	29 047	31 864.30
999 994	4 235	4 879.56	9 999 994	29 790	32 641.47
999 996	8 194	9 390.85	9 999 996	58 553	64 065.85
999 998	4 206	4 856.93	9 999 998	28 983	31 864.32
10^6	5 402	6 210.37	10^7	38 807	42 485.74
1 000 002	8 200	9 315.59	10 000 002	59 624	65 144.83
1 000 004	4 160	4 803.72	10 000 004	36 850	40 486.50
1 000 006	4 871	5 589.37	10 000 006	29 835	32 623.00
1 000 008	9 380	10 777.81	10 000 008	58 229	63 728.71
1 000 010	5 951	6 900.48	10 000 010	39 045	42 914.96

n	G(n)	D(n)	n	G(n)	D(n)
99 999 992	218 826	228 075.07	999 999 992	1 706 569	1 705 384.75
99 999 994	218 773	228 268.60	999 999 994	2 044 282	2 042 353.50
99 999 996	437 175	456 515.53	999 999 996	3 407 072	3 403 918.75
99 999 998	274 787	286 723.12	999 999 998	1 705 026	1 704 430.37
10^8	291 400	304 100.21	10^9	2 274 205	2 269 275.75
100 000 002	464 621	484 045.68	1 000 000 002	3 496 205	3 491 381.75
100 000 004	247 582	258 074.76	1 000 000 004	1 747 858	1 745 600.62
100 000 006	218 966	228 541.31	1 000 000 006	1 704 301	1 701 959.37
100 000 008	437 717	456 150.21	1 000 000 008	4 151 660	4 147 861.25
100 000 010	323 687	337 888.50	1 000 000 010	2 422 662	2 420 563.00

Bibliographie

[1] G. H. Hardy, J. E. Littlewood, Some problems of "partitio numerorum"; III : On the expression of a number as a sum of primes, Acta Math. **44**, 1923, 1-70.

[2] Denise Vella-Chemla, Quelques comètes : indicatrice d'Euler, somme des diviseurs, nombre de décompositions de Goldbach, http://denisevellachemla.eu/cometes1111.pdf.

[3] Denise Vella-Chemla, démonstration de la caractérisation des décomposants de Goldbach, http://denisevellachemla.eu/demo-caracterisation-DG.pdf.

Remarques sur le "Théorème de Goldbach".

de G. H. Hardy et J. E. Littlewood

New College, Oxford Trinity College, Cambridge

4.1. Notre méthode échoue quand $r = 2$[1]. Elle n'échoue pas *en principe*, car elle amène à un résultat précis qui semble être correct ; mais nous ne pouvons surmonter les difficultés de la preuve, même si nous supposons que $\theta = \frac{1}{2}$. La meilleure limite supérieure que nous pouvons déterminer pour l'erreur est trop large d'(environ) une puissance $n^{\frac{1}{4}}$[2].

La formule à laquelle notre méthode nous amène est contenue dans la

Conjecture A. *Tout nombre suffisamment grand est la somme de deux nombres premiers impairs. La formule asymptotique pour le nombre de décompositions est*

$$N_2(n) \sim 2\, C_2 \frac{n}{(\log\ n)^2} \prod_{\mathfrak{p}} \left(\frac{\mathfrak{p}-1}{\mathfrak{p}-2} \right) \tag{4.11}$$

où $\mathfrak{p}$ *est un nombre premier impair divisant* n *et*

$$C_2 = \prod_{\varpi=3}^{\infty} \left(1 - \frac{1}{(\varpi-1)^2} \right). \tag{4.12}$$

Nous ajoutons quelques mots concernant l'histoire de cette formule, et l'évidence empirique de sa vérité[3] .

La première formulation précise d'un résultat de ce genre semble être due à Sylvester[4], qui, dans un court résumé publié dans les *Proceedings of London Mathematical Society* en 1871, a suggéré que

$$N_2(n) \sim \frac{2n}{\log n} \prod \left(\frac{\varpi-2}{\varpi-1} \right), \tag{4.13}$$

Traduction d'un extrait de l'article *Some problems of "partitio numerorum" ; III : On the expression of a number as a sum of primes*, Acta Mathematica 44, 1923 (imprimé le 16 février 1922), 1-70.

[1]Cette note étant seulement la transcription d'un extrait de l'article de Hardy et Littlewood, il convient de préciser les notations utilisées par les auteurs, qui sont fournies dans les pages précédant la page 32 et non transcrites ici : ϖ est un nombre premier, $\mathfrak{p}$ est un nombre premier divisant n, $\Lambda(n) = \log \varpi$ si $n = \varpi^m$ et 0 sinon. $N_r(n)$ est le nombre de décompositions de n en une somme de r nombres premiers, en faisant attention à l'ordre dans lequel ils apparaissent et en autorisant des répétitions d'un même nombre premier. $v_r(n)$ désigne la somme $\sum_{\varpi_1+\varpi_2+\ldots+\varpi_r=n} \log\varpi_1\ \log\varpi_2 \ldots \log\varpi_r$ de telle façon que $\sum_{n=2}^{\infty} v_r(n)x^n = (f(x))^r$ en appelant $f(x)$ la fonction essentielle de l'article définie par $f(x) = \sum_{\varpi} \log\varpi x^{\varpi}$.

[2]Le paramètre θ est à trouver dans l'article complet en page 4, dans l'hypothèse R, qui stipule "il existe un nombre $\theta < \frac{3}{4}$ tel que $\beta \leq \theta$ pour tout ρ de toute $L(s)$.", ρ dénotant un zéro de $L(s)$ et β désignant la partie réelle de ρ. Tous les résultats de l'article dépendent de cette hypothèse R.

[3]Concernant l'histoire la plus ancienne du "théorème de Goldbach", voir L. E. Dickson, *History of the Theory of Numbers*, vol. I (Washington 1919), p. 421-425.

[4]J. J. Sylvester, "On the partition of an even number into two primes, *Proc. Lond. Math. Soc.* ser. I, vol. 4 (1871), p.4-6 (*Math. Papers*, vol. 2, p. 709-711). Voir aussi "On the Goldbach-Euler Theorem regarding prime numbers", Nature, vol. 55, (1896-7), p. 196-197, 269 (*Math. Papers*, vol. 4, p. 734-737).
Nous devons notre connaissance des notes de Sylvester sur le sujet à M. B. M. Wilson du Trinity College, Cambridge. Voir, en connexion avec tout ce qui suit Shah and Wilson, 1, et Hardy et Littlewood, 2.

avec

$$3 \leq \varpi < \sqrt{n}, \qquad \varpi \not| \, n.$$

Puisque

$$\prod_{\varpi<\sqrt{n}} \left(\frac{\varpi-2}{\varpi-1}\right) = \prod_{\varpi<\sqrt{n}} \left(1-\frac{1}{(\varpi-1)^2}\right) \prod_{\varpi<\sqrt{n}} \left(1-\frac{1}{\varpi}\right) \sim C_2 \prod_{\varpi<\sqrt{n}} \left(1-\frac{1}{\varpi}\right),$$

et[5]

$$\prod_{\varpi<\sqrt{n}} \left(1-\frac{1}{\varpi}\right) \sim \frac{2\, e^{-C}}{\log n}, \tag{4.14}$$

où C est la constante d'Euler, (4.13) est équivalente à

$$N_2(n) \sim 4\, e^{-C}\, C_2 \frac{n}{(\log\ n)^2} \prod_{\mathfrak{p}} \left(\frac{\mathfrak{p}-1}{\mathfrak{p}-2}\right), \tag{4.15}$$

et contredit (4.11), les deux formules différant d'un facteur $2e^{-C} = 1.123....$ Nous prouvons en 4.2 que (4.11) est la seule formule de cette sorte qui peut être correcte, de telle façon que la formule de Sylvester est erronée. Mais Sylvester a été le premier à identifier le facteur

$$\prod \left(\frac{\mathfrak{p}-1}{\mathfrak{p}-2}\right) \tag{4.16}$$

auquel les *irrégularités* de $N_2(n)$ sont dues. Il n'y a pas d'évidence claire de la manière dont il a été amené à ce résultat.

Une formule un peu différente a été suggérée par Stäckel[6] en 1896, i.e.

$$N_2(n) \sim \frac{n}{(\log\ n)^2} \prod \left(\frac{\mathfrak{p}}{\mathfrak{p}-1}\right).$$

Cette formule n'introduit pas le facteur (4.16), et ne donne pas de bonne approximation des faits ; elle a de toute façon été démontrée comme étant incorrecte par Landau[7] en 1900.

En 1915 est apparu un essai incomplet sur le théorème de Goldbach par Merlin[8]. Merlin ne donne pas une formule asymptotique complète mais il reconnaît (comme Sylvester avant lui) l'importance du facteur (4.16).

À peu près à la même époque, le problème a été attaqué par Brun[9]. La formule à laquelle l'argument de Brun conduit naturellement est

$$N_2(n) \sim 2H\ n \prod_{\mathfrak{p}} \left(\frac{\mathfrak{p}-1}{\mathfrak{p}-2}\right) \tag{4.17}$$

[5]Landau, p. 218.

[6]P. Stäckel, "Über Goldbach's empirisches Theorem : Jede grade Zahl kann als Summe von zwei Primzahlen dargestellt werden", *Göttinger Nachrichten*, 1896, p. 292-299.

[7]E. Landau, "Über die zahlentheoritsche Funktion $\varphi(n)$ und ihre Beziehung zum Goldbachschen Satz", *Göttinger Nachrichten*, 1900, p. 177-186.

[8]J. Merlin, "Un travail sur les nombres premiers", *Bulletin des sciences mathématiques*, vol. 39, 1915, p. 121-136.

[9]V. Brun, "Über das Goldbachsche Gesetz und die Anzahl der Primzahlpaare", *Archiv for Mathematik* (Christiania), vol. 34, part 2, 1915, no. 8, p. 1-15. La formule (4.18) n'est pas vraiment formulée par Brun : voir la discussion par Shah et Wilson, 1, et Hardy et Littlewood, 2. Voir aussi un second article du même auteur, "Sur les nombres premiers de la forme ap+b", ibid., part. 4, 1917„ n° 14, p. 1-9 ; et le postscriptum à ce mémoire.

où

$$H = \prod_{3 \le \varpi < \sqrt{n}} \left(1 - \frac{2}{\varpi}\right). \tag{4.171}$$

On montre facilement que cela est équivalent à

$$N_2(n) \sim 8e^{-2\gamma} C_2 \frac{n}{(\log\ n)^2} \prod_{\mathfrak{p}} \left(\frac{\mathfrak{p}-1}{\mathfrak{p}-2}\right), \tag{4.18}$$

et diffère de (4.11) par un facteur $4e^{-2C} = 1.263....$ L'argument de 4.2 montrera que cette formule, comme celle de SYLVESTER, est incorrecte.

Finalement, en 1916, STÄCKEL[10] est revenu sur ce sujet dans une série de mémoires publiés dans le *Sitzungsberichte der Heidelberger Akademie der Wissenschaften*, que nous n'avons pas pu consulter jusqu'à une date très récente. Des remarques plus approfondies concernant ces mémoires seront trouvées dans notre postscriptum final.

4.2. Procédons à la justification de notre assertion que les formules (4.15) et (4.18) ne peuvent pas être correctes.

Théorème F. *Supposons qu'il soit vrai que*[11]

$$N_2(n) \sim A \frac{n}{(\log n)^2} \prod_{\mathfrak{p}} \left(\frac{\mathfrak{p}-1}{\mathfrak{p}-2}\right) \tag{4.21}$$

si

$$n = 2^{\alpha} \mathfrak{p}^{a} \mathfrak{p}'^{a'} \ldots \qquad (\alpha > 0, a, a', \ldots > 0),$$

et

$$N_2(n) = o\left(\frac{n}{(\log n)^2}\right) \tag{4.22}$$

si n est impair. Alors

$$A = 2C_2 = \prod_{\varpi=3}^{\infty} \left(1 - \frac{1}{(\varpi-1)^2}\right). \tag{4.23}$$

Écrivons

$$\Omega(n) = A\, n \prod_{\mathfrak{p}} \left(\frac{\mathfrak{p}-1}{\mathfrak{p}-2}\right) \quad (n\ pair), \quad \Omega(n) = 0\ (n\ impair). \tag{4.24}$$

Alors, par (4.21) et par le théorème C, maintenant valide en vertu de (4.21),

$$v_2(n) = \sum_{\varpi + \varpi' = n} \log \varpi \log \varpi' \sim \Omega(n), \tag{4.25}$$

en comprenant que, lorsque n est impair, cette formule signifie

$$v_2(n) = o(n).$$

[10] P. STÄCKEL, "Die Darstellung der geraden Zahlen als Summen von zwei Primzahlen", 8 august 1916 ; "Die Lückenzahlen r-ter Stufe und die Darstellung der geraden Zahlen als Summen und Differenzen ungerader Primzahlen", I. Teil 27 décembre 1917, II Teil 19 janvier 1918, III Teil 19 juillet 1918.

[11] Durant tout 4.2, A est *la même* constante.

De plus, appelons

$$f(s) = \sum \frac{\Omega(n)}{n^s} = \sum \frac{\Omega(n)}{n^{1+u}},$$

cette série étant absolument convergente si $\Re(s) > 2, \Re(u) > 1$. Alors

$$\begin{aligned} f(s) &= A \sum_{n \equiv 0 \pmod 2} n^{-u} \prod_{\mathfrak{p}} \left(\frac{\mathfrak{p}-1}{\mathfrak{p}-2} \right) \\ &= A \sum_{a>0} 2^{-au} \mathfrak{p}^{-au} \mathfrak{p}'^{-a'u} \dots \frac{(\mathfrak{p}-1)(\mathfrak{p}'-1)\dots}{(\mathfrak{p}-2)(\mathfrak{p}'-2)\dots} \\ &= \frac{2^{-u}A}{1-2^{-u}} \prod_{\varpi=3}^{\infty} \left(1 + \frac{\varpi-1}{\varpi-2} \frac{\varpi^{-u}}{1-\varpi^{-u}} \right) = \frac{2^{-u}A}{1-2^{-u}} \xi(u), \end{aligned} \tag{4.26}$$

disons. Supposons maintenant que $u \to 1$, et posons

$$\eta(u) = \prod_{\varpi=3}^{\infty} \left(1 + \frac{\varpi^{-u}}{1-\varpi^{-u}} \right) = \prod_{\varpi=3}^{\infty} \left(\frac{1}{1-\varpi^{-u}} \right) = (1-2^{-u})\zeta(u).$$

Alors

$$\frac{\xi(u)}{\eta(u)} = \prod \left(\left(1 + \frac{\varpi-1}{\varpi-2} \frac{\varpi^{-u}}{1-\varpi^{-u}} \right) \Big/ \left(1 + \frac{\varpi^{-u}}{1-\varpi^{-u}} \right) \right)$$

$$\to \prod \left(\left(1 + \frac{1}{\varpi-2} \right) \Big/ \left(1 + \frac{1}{\varpi-1} \right) \right) = \prod \left(\frac{(\varpi-1)^2}{\varpi(\varpi-2)} \right)$$

$$= \prod \left(\frac{(\varpi-1)^2}{(\varpi-1)^2-1} \right) = \frac{1}{C_2}$$

Par conséquent,

$$f(s) \sim A\xi(u) \sim \frac{A}{C_2}\eta(u) \sim \frac{A}{2C_2}\zeta(u) \sim \frac{A}{2C_2(u-1)} = \frac{A}{2C_2(s-2)}. \tag{4.27}$$

D'un autre côté, lorsque $x \to 1$,

$$\sum v_2(n) x^n \sim \left(\sum \log \varpi x^{\varpi} \right)^2 \sim \frac{1}{(1-x)^2},$$

et ainsi [12]

$$v_2(1) + v_2(2) + \dots + v_2(n) \sim \frac{1}{2} n^2. \tag{4.28}$$

Il s'en déduit élémentairement que

$$g(s) = \sum \frac{v_2(n)}{n^s} \sim \sum \frac{1}{n^{s-1}} \sim \frac{1}{s-2}$$

lorsque $s \to 2$; et par conséquent[13] que (sous les hypothèses (4.21) et (4.22))

$$f(s) \sim \frac{1}{s-2}. \tag{4.29}$$

[12] Nous utilisons ici le Théorème 8 de notre article "Théorèmes Taubériens concernant les séries de puissances et les séries de Dirichlet dont les coefficients sont positifs", *Proc. London Math. Soc.*, sér. 2, vol. 13, p. 174-192. C'est la preuve la plus rapide, mais en aucun cas la plus élémentaire. La formule (4.28) est équivalente à la formule

$$\sum_{1}^{n} N_2(m) \sim \frac{n^2}{2(\log n)^2}$$

utilisée par Landau dans sa note citée p. 33.

[13] Pour les théorèmes généraux incluant ceux utilisés ici comme des cas très particuliers, voir K. Knopp, "Divergenscharactere gewisser Dirichlet'scher Reihen", *Acta Mathematica*, vol. 34, 1909, p. 165-204 (*e.g.* Satz III, p. 176).

En comparant (4.27) et (4.29), nous obtenons le théorème comme résultat.

4.3. Le fait que les formules de Sylvester et de Brun contiennent toutes deux un facteur erroné, et que ce facteur dans les deux cas soit une simple fonction du nombre e^{-C}, n'est pas si remarquable que cela pourrait le sembler.

En premier lieu, nous observons que toute formule dans la théorie des nombres premiers, *déduite de considérations de probabilités,* est susceptible d'être erronée de cette même manière. Considérons, par exemple, le problème *"Quelle est la probabilité qu'un grand nombre soit premier ?"*. Nous savons que la réponse à cette question est que cette probabilité est approximativement $\dfrac{1}{\log n}$.

Maintenant, la probabilité que n ne soit pas divisible par n'importe quel nombre premier inférieur à un nombre fixé x est asymptotiquement équivalente à

$$\prod_{\varpi<x}\left(1-\frac{1}{\varpi}\right);$$

et il serait naturel d'inférer[14] que la probabilité requise est asymptotiquement équivalente à

$$\prod_{\varpi<\sqrt{n}}\left(1-\frac{1}{\varpi}\right)$$

Mais[15]

$$\prod_{\varpi>\sqrt{n}}\left(1-\frac{1}{\varpi}\right)\sim\frac{2e^{-C}}{\log n};$$

et notre inférence est incorrecte, selon un facteur de $2e^{-C}$[16].

Il est vrai que l'argument de Brun n'est pas énoncé en termes de probabilités[17], mais il entraîne un passage heuristique à la limite exactement du même genre que l'argument que nous venons de citer. Brun a d'abord trouvé (par une utilisation ingénieuse du "crible d'Eratosthène") une formule asymptotique du nombre de représentations de n comme somme de deux nombres, *ni l'un ni l'autre divisible par n'importe quel nombre fixé de nombres premiers*. Cette formule est correcte et la preuve est valide. Il en est de même de la première étape de l'argument ci-dessus ; elle s'appuie sur une énumération de cas, et toute référence aux "probabilités"[18] est facilement éliminée. C'est dans le passage à la limite que l'erreur est introduite, et la nature de l'erreur est la même dans un cas et dans l'autre.

4.4. Shah et Wilson ont testé la conjecture A extensivement par comparaison avec les données empiriques collectées par Cantor, Aubry, Haussner, et Ripert. Nous réimprimons leur table de résultats ; mais quelques remarques préliminaires sont nécessaires. En premier lieu, il est essentiel, lors d'un test numérique, de travailler avec une formule pour $N_2(n)$, comme celle de (4.11), et non avec une formule pour $v_2(n)$, comme celle de (4.25). Dans notre analyse, d'un autre côté, c'est $v_2(n)$ qui se présente en premier, et la formule pour $N_2(n)$ est secondaire. Pour dériver la formule asymptotique pour $N_2(n)$, nous écrivons

$$v_2(n)=\sum_{\varpi+\varpi'=n}\log\varpi\log\varpi'\sim(\log n)^2N_2(n).$$

[14] On peut aussi bien remplacer $\varpi<\sqrt{n}$ par $\varpi<n$, auquel cas nous obtiendrions une probabilité moitié moins grande. Cette remarque est en elle-même suffisante pour montrer le caractère insatisfaisant de cet argument.

[15] Landau, p. 218.

[16] ndd : noter le changement de signe entre les produits dans l'article original.

[17] Nous n'avons pas de moyen direct de juger si l'argument de Sylvester est valide ou pas.

[18] *La probabilité* n'est pas une notion de mathématique pure, mais une notion de philosophie ou de physique.

Le facteur $(\log n)^2$ est certainement erroné selon un ordre de grandeur de $\log n$, et il est plus naturel [19] de remplacer $v_2(n)$ par

$$((\log n)^2 - 2\log n + \ldots)\, N_2(n).$$

Pour la formule *asymptotique*, naturellement, la substitution que nous adoptons est indifférente. Mais, dans un but de *vérification dans la limite des calculs*, elle n'est en aucun cas indifférente, car le terme en $\log n$ n'est en aucun cas d'une importance négligeable ; et l'on trouvera que cela fait une différence vitale dans la plausibilité des résultats. Gardant ces considérations à l'esprit, Shah et Wilson ont travaillé non avec la formule (4.11), mais avec la formule modifiée

$$N_2(n) \sim \rho(n) = 2\, C_2 \frac{n}{(\log n)^2 - 2\log n} \prod_{\mathfrak{p}} \left(\frac{\mathfrak{p}-1}{\mathfrak{p}-2} \right).$$

Le défaut de tenir compte de ce genre de choses a été responsable d'un certain nombre d'incompréhensions par le passé. Ainsi (comme cela est souligné par Shah et Wilson[20]), la formule erronée de Sylvester donne, pour les valeurs de n dans les limites de la Table I, des résultats décidément *meilleurs* que ceux obtenus par la formule (4.11) *inmodifiée*.

Il y a un autre point de moindre importance. La fonction qui se présente le plus naturellement dans notre analyse n'est pas

$$f(x) = \sum \log \varpi x^{\varpi}$$

mais

$$g(x) = \sum \Lambda(n) x^n = \sum_{\varpi, l} \log \varpi x^{\varpi^l}.$$

Les fonctions numériques correspondantes ne sont pas $v_2(n)$ et $N_2(n)$, mais

$$g_2(n) = \sum_{m+m'=n} \Lambda(m)\Lambda(m'), \quad Q_2(n) = \sum_{\varpi^l + \varpi^{l'} = n} 1$$

(de telle façon que $Q_2(n)$ est le nombre de décompositions de n *en deux nombres premiers ou en deux puissances de nombres premiers).* Ici à nouveau, $N_2(n)$ et $Q_2(n)$ sont asymptotiquement équivalentes ; la différence entre elles est en effet d'un ordre moindre que les erreurs que nous négligeons dans tous les cas ; mais on doit dire quelque chose pour prendre cette dernière comme base de comparaison, quand (ce qui est inévitable), les valeurs de n ne sont pas très grandes.

Dans la table, les décompositions en nombres premiers, et puissances de nombres premiers sont prises en compte séparément ; mais c'est le total qui est comparé à $\rho(n)$. La valeur de la constante $2C_2$ est 1.3203. On verra que la comparaison entre les valeurs calculées et les valeurs effectives est excellente.

[19] Comparer Shah et Wilson, *l. c.*, p. 238. On peut arriver à la même conclusion par différents chemins.
[20] *l. c.*, p. 242.

Table I.

n	$Q_2(n)$	$\rho(n)$	$Q_2(n)/\rho(n)$
$30 = 2.3.5$	$6 + 4 = 10$	22	0.45
$32 = 2^5$	$4 + 7 = 11$	8	1.38
$34 = 2.17$	$7 + 6 = 13$	9	1.44
$36 = 2^2.3^2$	$8 + 8 = 16$	17	0.94
$210 = 2.3.5.7$	$42 + 0 = 42$	49	0.85
$214 = 2.107$	$17 + 0 = 17$	16	1.07
$216 = 2^3.3^3$	$28 + 0 = 28$	32	0.88
$256 = 2^8$	$16 + 3 = 19$	17	1.10
$2\,048 = 2^{11}$	$50 + 17 = 67$	63	1.06
$2\,250 = 2.3^2.5^3$	$174 + 26 = 200$	179	1.11
$2\,304 = 2^8.3^2$	$134 + 8 = 142$	136	1.04
$2\,306 = 2.1153$	$67 + 20 = 87$	69	1.26
$2\,310 = 2.3.5.7.11$	$228 + 16 = 244$	244	1.00
$3\,888 = 2^4.3^5$	$186 + 24 = 210$	197	1.06
$3\,898 = 2.1949$	$99 + 6 = 105$	99	1.06
$3\,990 = 2.3.5.7.19$	$328 + 20 = 348$	342	1.02
$4\,096 = 2^{12}$	$104 + 5 = 109$	102	1.06
$4\,996 = 2^2.1249$	$124 + 16 = 140$	119	1.18
$4\,998 = 2.3.7^2.17$	$228 + 20 = 308$	305	1.01
$5\,000 = 2^3.5^4$	$150 + 26 = 176$	157	1.12
$8\,190 = 2.3^2.5.7.13$	$578 + 26 = 604$	597	1.01
$8\,192 = 2^{13}$	$150 + 32 = 182$	171	1.06
$8\,194 = 2.17.241$	$192 + 10 = 202$	219	0.92
$10\,008 = 2^3.3^2.139$	$388 + 30 = 418$	396	1.06
$10\,010 = 2.5.7.11.13$	$384 + 36 = 420$	384	1.09
$10\,014 = 2.3.1669$	$408 + 8 = 416$	396	1.05
$30\,030 = 2.3.5.7.11.13$	$1\,800 + 54 = 1\,854$	1 795	1.03
$36\,960 = 2^5.3.5.7.11$	$1\,956 + 38 = 1\,994$	1 937	1.03
$39\,270 = 2.3.5.7.11.17$	$2\,152 + 36 = 2\,188$	2 213	0.99
$41\,580 = 2^2.3^3.5.7.11$	$2\,140 + 44 = 2\,184$	2 125	1.03
$50\,026 = 2.25013$	$702 + 8 = 710$	692	1.03
$50\,144 = 2^5.1567$	$607 + 32 = 706$	694	1.02
$170\,166 = 2.3.79.359$	$3\,734 + 46 = 3\,780$	3 762	1.00
$170\,170 = 2.5.7.11.13.17$	$3\,784 + 8 = 3\,792$	3 841	0.99
$170\,172 = 2^2.3^2.29.163$	$3\,732 + 48 = 3\,780$	3 866	0.98

Positionner les nombres sur le disque-unité par leurs restes de divisions euclidiennes (Denise Vella-Chemla, 28.8.2020)

On voudrait fournir ici des résultats surprenants qui nous ont été inspirés par un travail tout l'été autour de la preuve par Alain Connes du théorème de Morley ([1]).

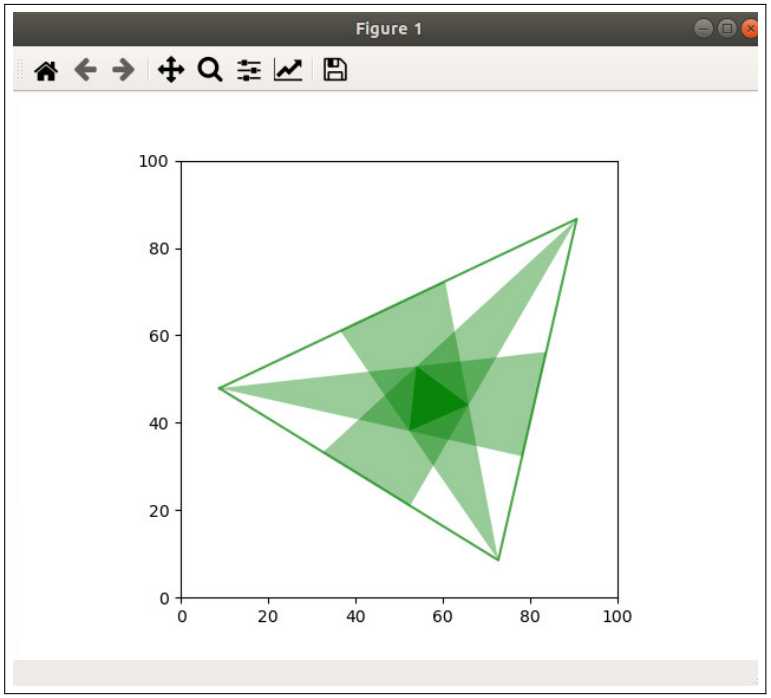
En effet, on souhaitait programmer la preuve pour essayer de vérifier que le théorème n'est pas applicable pour les triangles sphériques (voir notamment la conférence [2]).

Programme de vérification et visualisation du Théorème de Morley dans le plan

```
from math import *
from matplotlib import *
from matplotlib.pyplot import *

def vecteur(point1, point2):
  return [y - x for x, y in zip(point1, point2)]

def add(vecteur1, vecteur2):
  return [x + y for x, y in zip(vecteur1, vecteur2)]

def norme(vecteur):
  return sqrt(prodscal(vecteur, vecteur))

def prodscal(vecteur1, vecteur2):
  return sum([x*y for x, y in zip(vecteur1, vecteur2)])

def determ(vecteur1, vecteur2):
 return vecteur1[0]*vecteur2[1]-vecteur1[1]*vecteur2[0]

def angle(vecteur1, vecteur2):
  cosinus=prodscal(vecteur1, vecteur2)/(norme(vecteur1)*norme(vecteur2))
  sinus=determ(vecteur1, vecteur2)/(norme(vecteur1)*norme(vecteur2))
  return atan2(sinus,cosinus)

def rotation(u, theta):
  return [u[0]*cos(theta)-u[1]*sin(theta),u[0]*sin(theta)+u[1]*cos(theta)]

def intersecte(x, y, z, t):
  a1 = (y[1]-x[1])/(y[0]-x[0])
  b1 = x[1]-a1*x[0]
  a2 = (t[1]-z[1])/(t[0]-z[0])
  b2 = z[1]-a2*z[0]
  return [(b2-b1)/(a1-a2),((b2-b1)/(a1-a2))*a1+b1]

a=[8.83, 47.89]
b=[72.74, 8.55]
c=[90.72, 86.63]
ab = vecteur(a,b) ; ac = vecteur(a,c)
ba = vecteur(b,a) ; bc = vecteur(b,c)
ca = vecteur(c,a) ; cb = vecteur(c,b)
anglea=angle(ab,ac)  ; angleb=angle(bc,ba)  ; anglec=angle(ca,cb)
aprime = add(a, rotation(ab,anglea/3.0))  ; aseconde = add(a, rotation(ab,anglea*2.0/3.0))
bprime = add(b, rotation(bc,angleb/3.0)) ; bseconde = add(b, rotation(bc,angleb*2.0/3.0))
cprime = add(c, rotation(ca,anglec/3.0)) ; cseconde = add(c, rotation(ca,anglec*2.0/3.0))
p=intersecte(a,aseconde,b,c) ; q=intersecte(a,aprime,b,c)
r=intersecte(c,cseconde,a,b) ; s=intersecte(c,cprime,a,b)
t=intersecte(b,bseconde,c,a) ; u=intersecte(b,bprime,c,a)
n=intersecte(a,aseconde,c,cprime)
o=intersecte(c,cseconde,b,bprime)
x=intersecte(b,bseconde,a,aprime)
print('Normes des cotes %3.15f '% norme(vecteur(n,o)))
print('Normes des cotes %3.15f '% norme(vecteur(o,x)))
print('Normes des cotes %3.15f '% norme(vecteur(x,n)))

fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
matplotlib.pyplot.plot([a[0],b[0],c[0],a[0]],[a[1],b[1],c[1],a[1]], 'g', alpha=0.7)
matplotlib.pyplot.fill([a[0],p[0],q[0]],[a[1],p[1],q[1]], 'g', 2, alpha=0.4)
matplotlib.pyplot.fill([c[0],r[0],s[0]],[c[1],r[1],s[1]], 'g', 2, alpha=0.4)
matplotlib.pyplot.fill([b[0],t[0],u[0]],[b[1],t[1],u[1]], 'g', 2, alpha=0.4)
matplotlib.pyplot.fill([n[0],o[0],x[0]],[n[1],o[1],x[1]], 'g', 2, alpha=0.8)
matplotlib.pyplot.xlim(0,100)
matplotlib.pyplot.ylim(0,100)
ax.set\_aspect('equal')
matplotlib.pyplot.show()
```

Le programme ci-dessus produit la visualisation ci-après, et imprime comme longueur des normes des côtés du triangle de Morley (le triangle équilatéral de sommets les intersections des trissectrices adjacentes du triangle quelconque "externe") la valeur 14.863746806168091 pour deux côtés et la valeur 14.863746806168082 pour

le troisième côté : la différence entre les valeurs est négligeable, les côtés sont de longueur égale, le théorème le prouve, même si l'ordinateur ne le constate qu'à un ε près (d'ailleurs, l'ordinateur détecte toujours l'égalité de flottants (les réels en langage informatique) à un ε près).



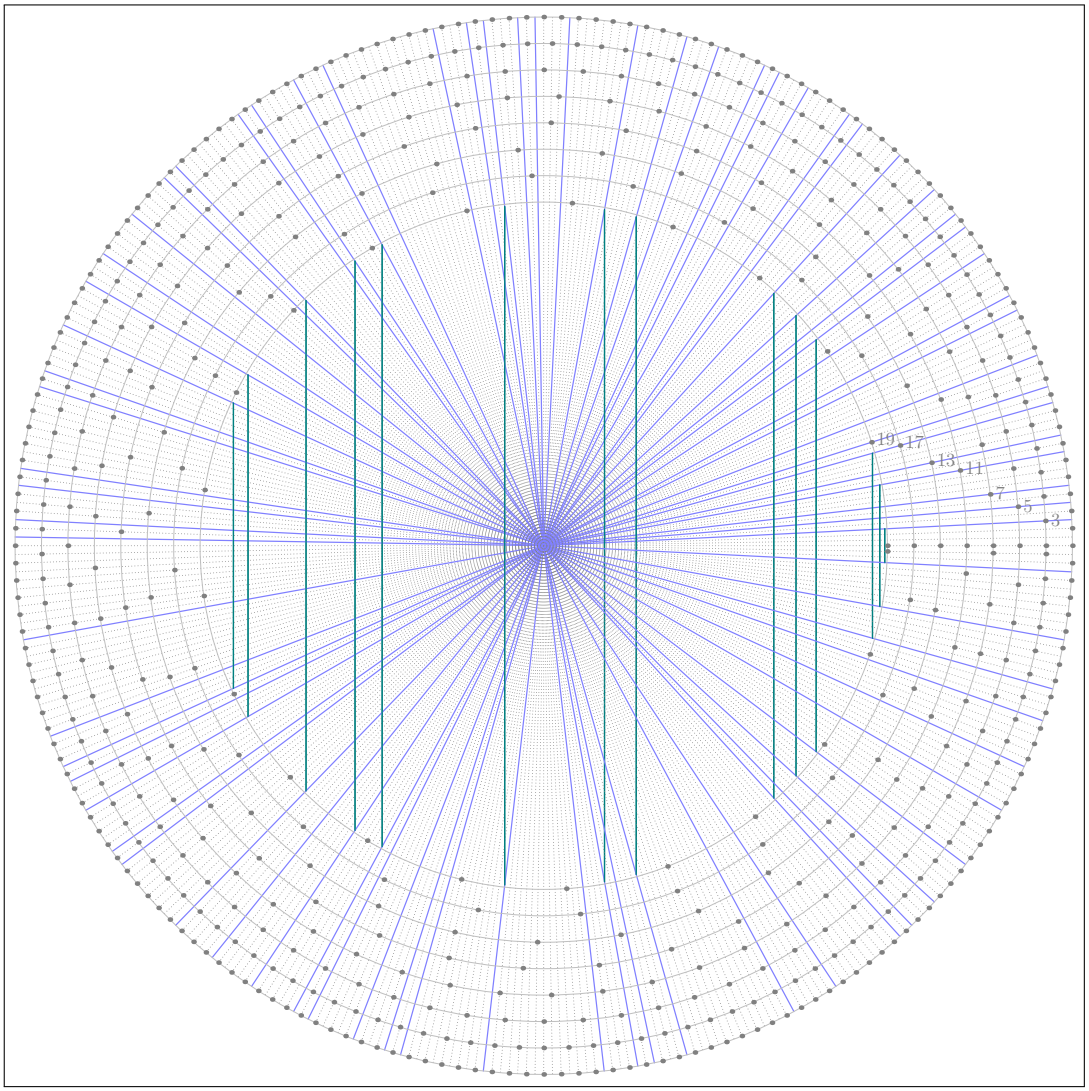
On avait également programmé des rotations dans le cercle unité en langage Asymptote, puis en langage python, pour visualiser les décomposants de Goldbach sur le cercle par les programmes suivants :

Programme asymptote de la visualisation de Goldbach des décomposants de 400 :

```
unitsize(1cm);
real xmax = 12;

pen prem = lightblue;
pen couleurdot = gray;
pen comp = dotted+gray;
pen lieprem = deepcyan;

int n = 400;
int m = floor(sqrt(n));

int[] premier = {2};

bool[] est_premier;
for (int k = 0; k < n; ++k) {
  est_premier.push(false);
}

bool is_prime(int p) {
  bool flag = true;
  for (int k = 0; k < premier.length; ++k) {
    if (p % premier[k] == 0) {
      flag = false;
      break;
    }
  }
  return flag;
}

for (int p = 3; p <= n; p = p+2) {
  if (is_prime(p)) {
    premier.push(p);
    est_premier[p] = true;
  }
}
```

```
path cercle = scale(xmax)*unitcircle;
path cercle2 = scale(7.75)*unitcircle;
for (int k = 0; k < n; ++k) {
  real t = length(cercle)*k/n;
  real t2 = length(cercle)*(n-k)/n;
  pair P = point(cercle, t);
  pair P2 = point(cercle2, t);
  pair P3 = point(cercle2, t2);
  if (est_premier[k]) {
    draw((0,0)--P, prem+0.8bp);
    if (est_premier[n-k])
      draw(P2--P3, lieprem+0.8bp);
  } else {
    draw((0,0)--P, comp);
  }
}

for (int k = 0; premier[k] < m; ++k) {
  path cercle = scale(xmax*(1-k/m))*unitcircle;
  draw(cercle, mediumgray);
  for (int d = 0; d < n; d = d+premier[k]) {
    real t = length(cercle)*d/n;
    pair P = point(cercle, t);
    if (d <= m) {if (est_premier[d]) dot(format("%d",d), P, dir(P), couleurdot);
                 else dot(P,couleurdot);}
    else dot(P, couleurdot);
  }
}
```

qui produit la visualisation ci-dessous ($400 = 3 + 397 = 11 + 389 = 17 + 383 = 41 + 359 = 47 + 353 = 53 + 347 = 83 + 317 = 89 + 311 = 107 + 293 = 131 + 269 = 137 + 263 = 149 + 251 = 167 + 233 = 173 + 227$).



Le même programme en python / matplotlib fournit une visualisation moins fine.

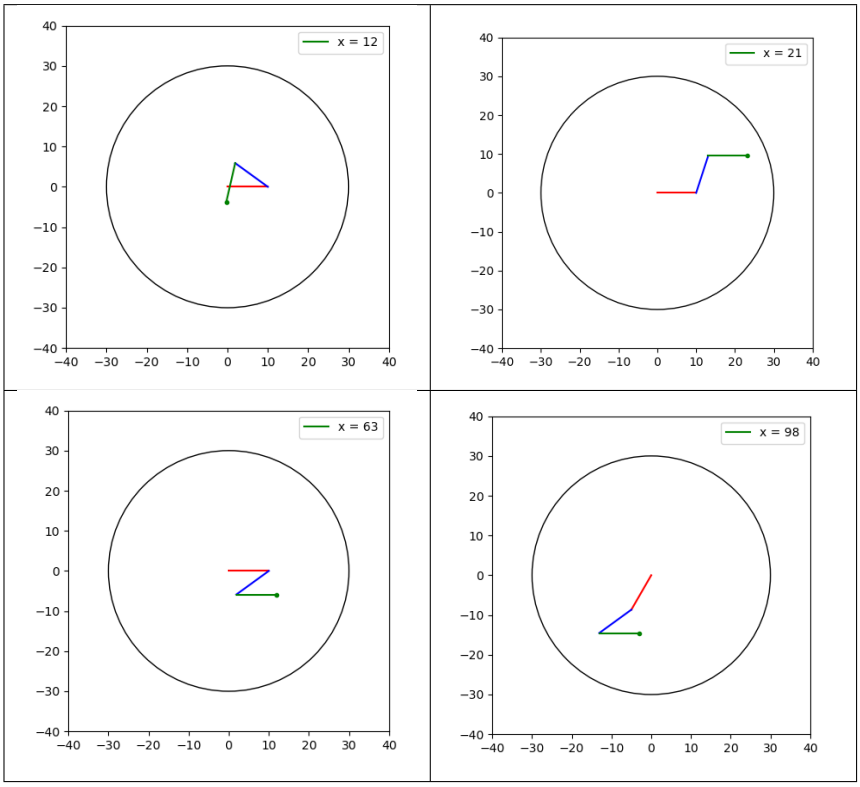
```
from math import *
from matplotlib import *
import matplotlib.pyplot as plt

def prime(atester):
  pastrouve = True
  k = 2
  if (atester == 0): return False
  if (atester == 1): return False
  if (atester == 2): return True
  if (atester == 3): return True
  if (atester == 5): return True
  if (atester == 7): return True
  while (pastrouve):
    if ((k * k) > atester):
      return True
    else:
      if ((atester % k) == 0):
        return False
      else: k=k+1

fig = plt.gcf()
ax = fig.gca()
xmax = 50
n = 98
m = floor(sqrt(n))
ax = plt.gca() ; ax.cla() ; plt.axis('scaled')
num = 0
for p in range(1,m):
  if (prime(p)):
    rayon = xmax*(1-num/m)
    cercle = plt.Circle((0,0), rayon, fill=False)
    num = num+1
    ax.add_artist(cercle)
num = 0
for p in range(1,m):
  if (prime(p)):
    for d in range(n):
      if ((d % p) == 0):
        t = 2*pi*d/n
        rayon = xmax*(1-num/m)
        ax.plot(rayon*cos(t), rayon*sin(t), 'bo', markersize=3)
        ax.text(rayon*cos(t), rayon*sin(t), str(d), color='blue',
                fontsize=9, va='bottom', ha='right', alpha=0.85)
    num = num+1
cercle = xmax
xmin = 33
cercle2 = 33
for q in range(n):
  t = 2*pi*q/n
  t2 = 2*pi*(n-q)/n
  xP, yP = xmax*cos(t), xmax*sin(t)
  xP2, yP2 = xmin*cos(t), xmin*sin(t)
  xP3, yP3 = xmin*cos(t2), xmin*sin(t2)
  xs = [0,xP]
  ys = [0,yP]
  if (prime(q)):
    plt.plot(xs, ys, color='fuchsia')
    xs = [xP2,xP3]
    ys = [yP2,yP3]
    if (prime(n-q)):
      plt.plot(xs, ys, color='chartreuse')
  else:
    plt.plot(xs, ys, color='gray', ls=':')
ax.set_xlim(((-1)*xmax-10, xmax+10))
ax.set_ylim(((-1)*xmax-10, xmax+10))
plt.show()
```

De fil en aiguille, on a eu l'idée d'un programme, qui utiliserait notre modélisation par les restes pour la conjecture de Goldbach (le Snurp∞, cf. [3]) ainsi que des rotations.

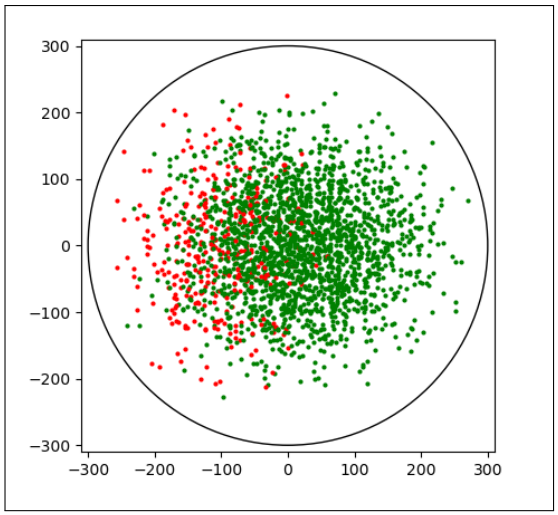
Présentons l'idée sur quelques exemples. On se place sur le disque unité. On représente chaque nombre comme un petit chemin, depuis le centre du cercle unité, jusqu'à un point du disque à déterminer. Le chemin est une ligne brisée dont les segments sont obtenus comme des rotations d'angles correspondant aux restes du nombre dans les différents corps premiers $\mathbb{Z}/p_i\mathbb{Z}$.

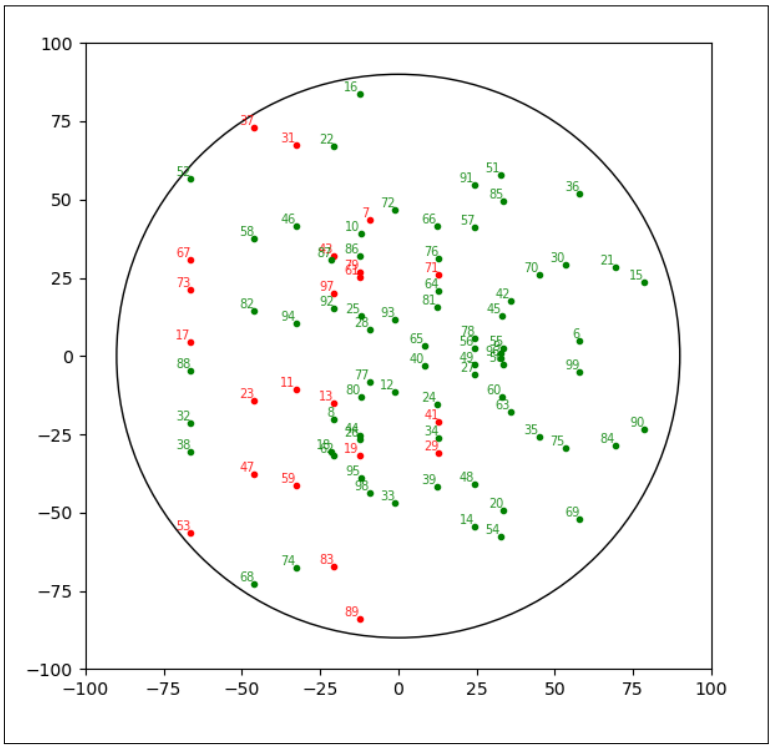
Ci-dessous, sont fournis les petits chemins pour les nombres 12, 21, 63 et 98, en ne gérant que 3 restes modulaires, le reste modulo 3 partant du centre et en rouge, suivi du reste modulo 5 en bleu, suivi du reste modulo

7 en vert. Le dernier point atteint est visualisé par un petit cercle plein vert. Par exemple, la “manivelle” pour 21 représente le fait que 21 est congru à 0 modulo 3, à 1 modulo 5 (angle orienté vers la première racine $5^{\text{ème}}$, à $e^{\frac{2i\pi}{5}}$) de l'unité, et congru à 0 modulo 7 (on voit également $12 \equiv 0 \pmod 3; 12 \equiv 2 \pmod 5; 12 \equiv 5 \pmod 7$ ou $21 \equiv 0 \pmod 3; 21 \equiv 1 \pmod 5; 21 \equiv 0 \pmod 7$ ou enfin $98 \equiv 2 \pmod 3; 98 \equiv 3 \pmod 5; 98 \equiv 0 \pmod 7$).



On a dans un premier temps utilisé nos petits chemins dans les corps premiers pour les nombres premiers considérés “dans l'ordre”, de 2, 3, 5,... à 13, du plus petit au plus grand, et on était un peu découragée car les “clusters” de points ne semblaient pas être aisément circonscriptibles, comme on le voit sur les 2 visualisations ci-dessous.

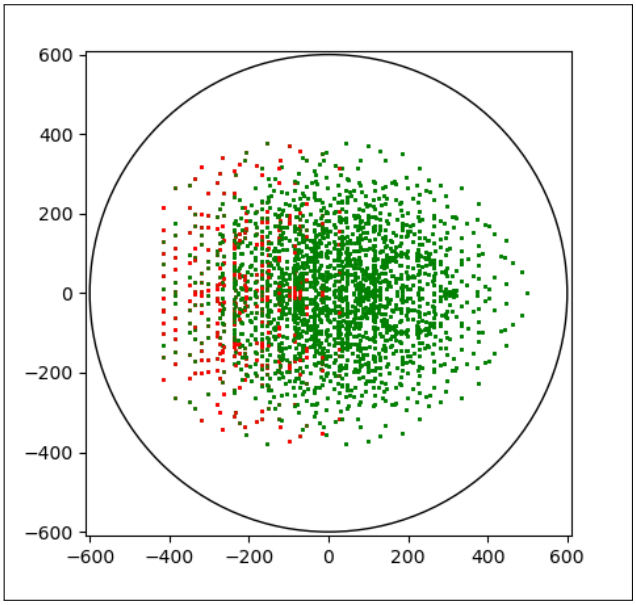




On utilise un programme sur le modèle de celui ci-dessous :

```
import math
from math import *
import matplotlib.pyplot as plt

def prime(atester):
  pastrouve = True ; k = 2 ;
  if (atester in [0,1]): return False ;
  if (atester in [2,3,5,7]): return True ;
  while (pastrouve):
    if ((k * k) > atester): return True
    else:
      if ((atester % k) == 0): return False
      else: k=k+1

n = 30 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [2,3,5]
nbprime = len(Ens) + 1 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)

#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= n:
  x0, y0 = 0, 0
  for element in Ens:
    t = 2*pi*(x % element)/element
    x0 = x0 + xmax*math.cos(t)
    y0 = y0 + xmax*math.sin(t)
  #print(str(x0)+'   '+str(y0))
  if (prime(x)):
    plt.plot(x0, y0, 'r', marker='o', markersize=1)
    ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
  else:
    plt.plot(x0, y0, 'g', marker='o', markersize=1)
    ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
  if (x != 1):
    ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.25)
  xprec = x0
  yprec = y0
  ax.set_aspect('equal')
  x = x+1
plt.show()
```

Voici le résultat de ce programme pour les entiers inférieurs à 10000.



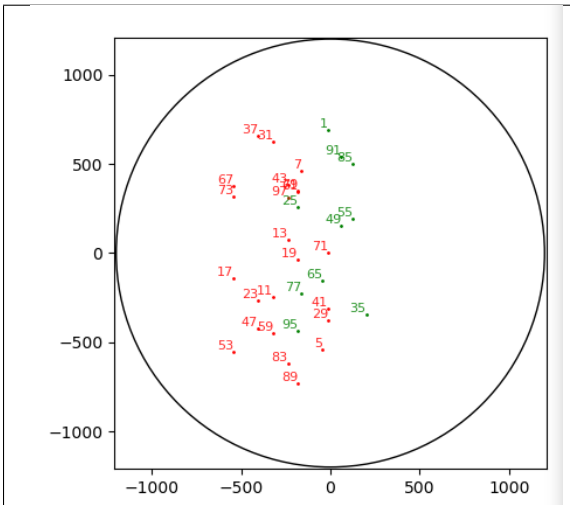
Le positionnement symétrique des points (pas forcément de la même couleur) par rapport à l'axe des abscisses se justifie par le fait qu'il y a probabilistiquement autant de points de reste x dans une division par p qu'il y en a de reste $p - x$ dans une division par p.

Des expérimentations informatiques avec des programmes similaires s'avèrent surprenantes, même si très difficiles à comprendre a priori. On fournit une ébauche d'explication mais voyons les résultats.

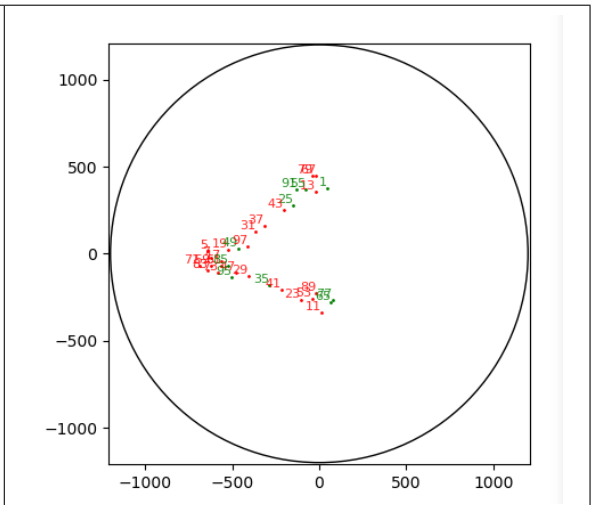
On réalise qu'on obtient des résultats qui semblent intéressants, ou du moins des alignements (même si on n'arrive pas à mettre les nombres premiers et les nombres composés sur des droites distinctes), en "isolant" les restes dans 4 corps premiers, dont les corps $\mathbb{Z}/3\mathbb{Z}$ et $\mathbb{Z}/2\mathbb{Z}$.

On a aussi l'impression que les visualisations présentent de meilleures propriétés si les deux autres corps premiers sont des corps de nombres premiers jumeaux. Enthousiasmée par les visualisations, on a même tenté 6 corps premiers au lieu de 4 (les nombres premiers 2 et 3, et 2 paires de nombres premiers jumeaux).

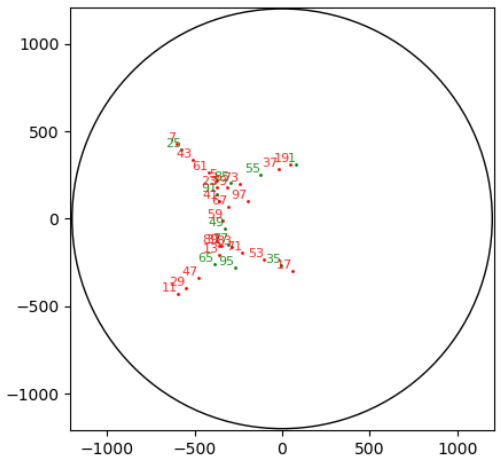
Les alignements sont provoqués par le fait d'avoir des restes simultanément complémentaires dans les différents corps premiers. Par exemple, 67 et 73 sont alignés selon les restes 7 et 5 parce qu'on a à la fois $67 \bmod 7 + 73 \bmod 7 = 7$ et $67 \bmod 5 + 73 \bmod 5 = 5$. Les cercles contiennent parfois des nombres de même reste dans des divisions euclidiennes par un certain nombre.



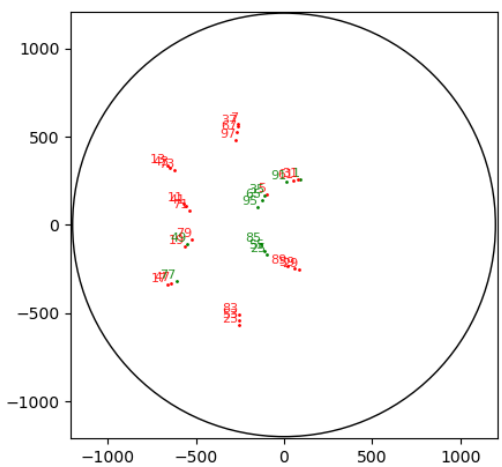
Premiers 7, 5, 3, 2.



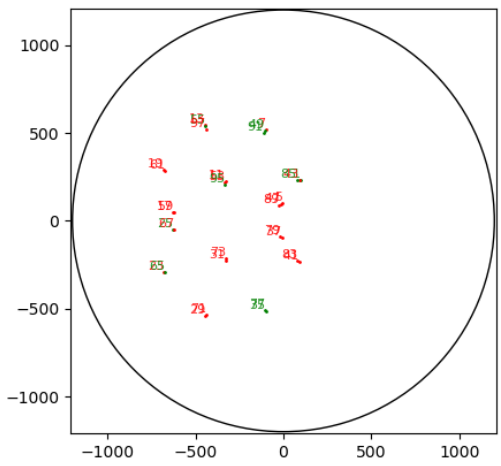
Premiers 13, 11, 3, 2.



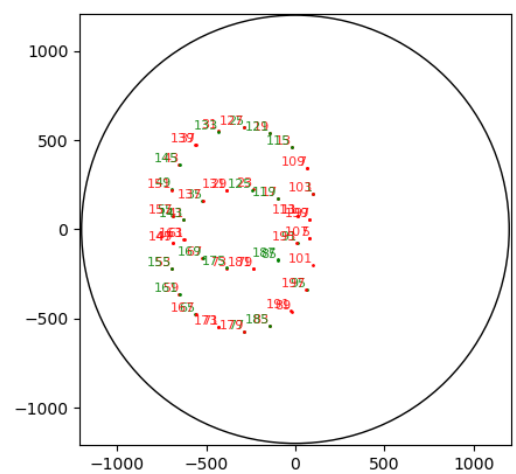
Premiers 19, 17, 3, 2.



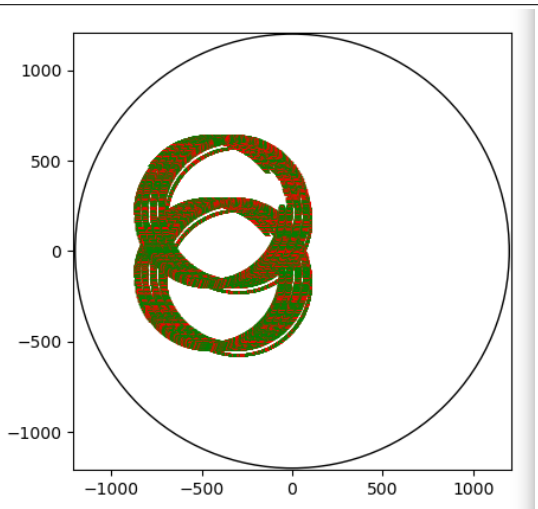
Premiers 31, 29, 3, 2.



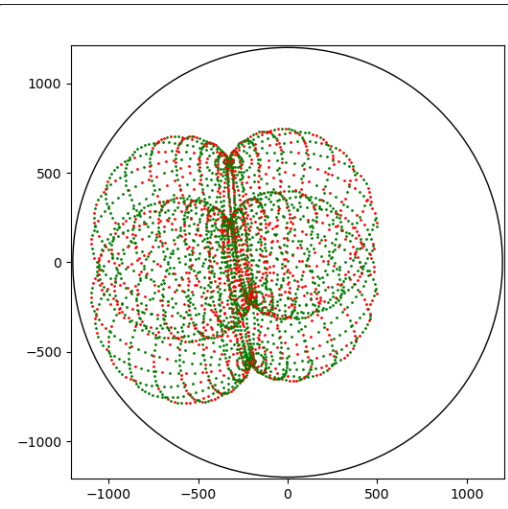
Premiers 43, 41, 3, 2.



Premiers 103, 101, 3, 2.



Premiers 9931, 9929, 3, 2.

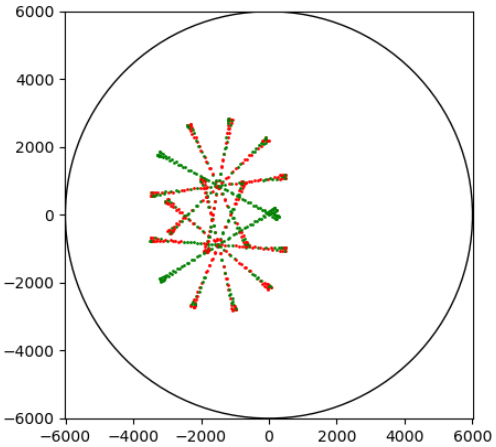


Premiers 9931, 9929, 103, 101, 3, 2.

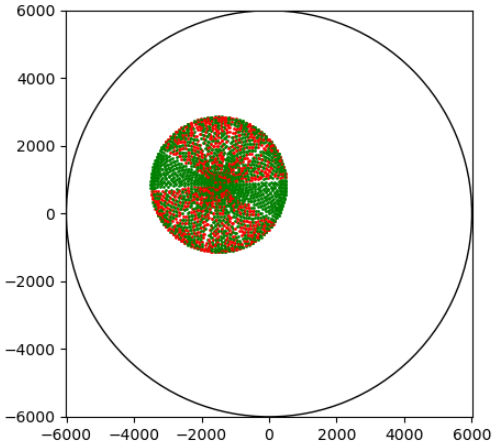
□ □

Dans la visualisation 13-11-3-2 (sorte de bec d'oiseau), les deux branches correspondent à nouveau aux $6x+1$, $6x-1$ (6 divise 12 entre les jumeaux 11 et 13). Dans la visualisation suivante 19-17-3-2, il y a un embranchement supplémentaire sur chacune des branches haute et basse. Le haut et le bas sont bien séparés comme d'habitude en $6x+1$ et $6x-1$, puis par exemple en haut, ce sont les $9x-1$ et les $9x+1$ qui sont dans les branches droite et gauche (car 9 divise 18 entre les jumeaux). Pour la visualisation 31-29, ce sont comme attendu des séparations selon deux pentagones, 5 divisant 30 entre les jumeaux 29 et 31. La visualisation suivante 43-41 séparant en heptagones. Pour ces deux séparations pentagonale et heptagonale, on constate avec intérêt que certains sommets ne semblent contenir que des premiers, ça reste à vérifier. Pour le 103-101, comme attendu, 102=2.3.17, on a deux séparations en deux cercles à 17 sommets, l'un en haut l'autre en bas. L'avant-dernière visualisation montre aussi les deux cercles et la dernière visualisation, qui fait penser à un schéma d'embryogenèse, reste difficile à interpréter avec ses 2 couples de jumeaux (il faut dire que 9930=2.3.5.331).

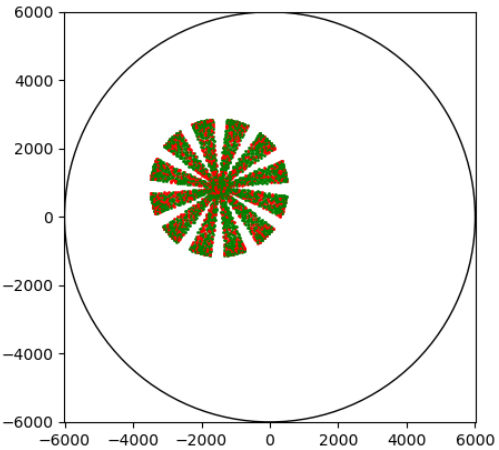
Ci-après, quelques visualisations supplémentaires qu'on pourrait appeler "fleurs d'enfants".



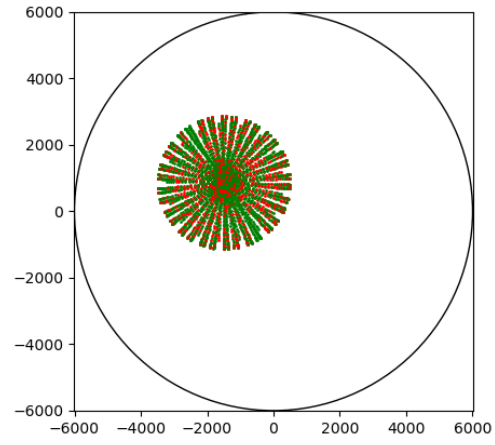
61, 59, haut et bas jusqu'à 2020



61, 59, $6x+1$ seuls, jusqu'à 20 020

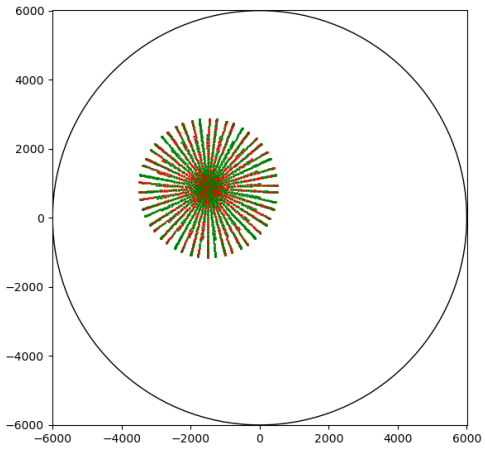


73, 71

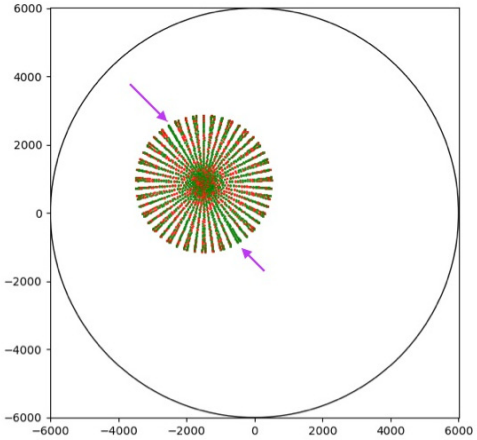


103, 101

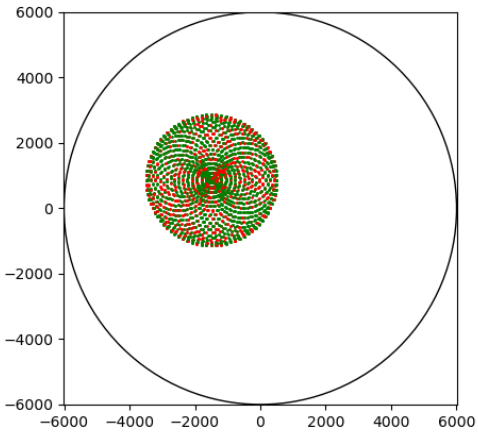
Dès le second dessin, on "oublie informatiquement" les $6x-1$ qui présente une structure symétrique de celle présentée par les $6x+1$, de l'autre côté (sous) l'axe des abscisses.



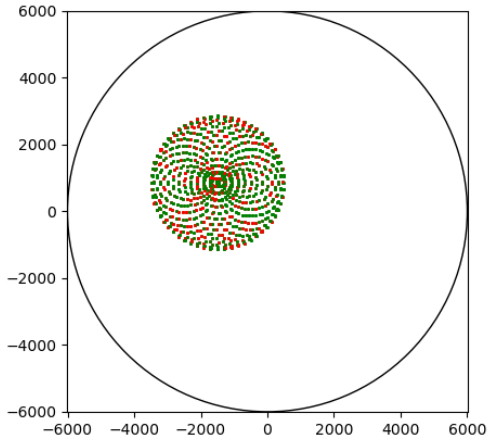
199, 197



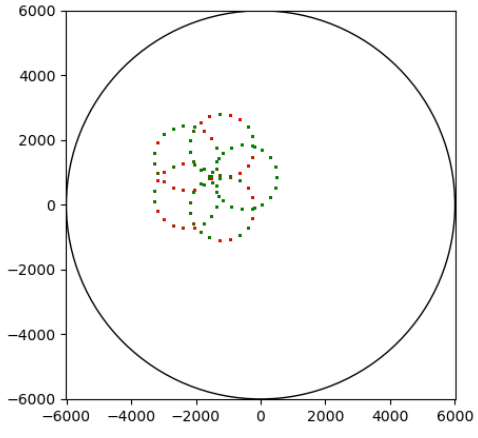
139, 137



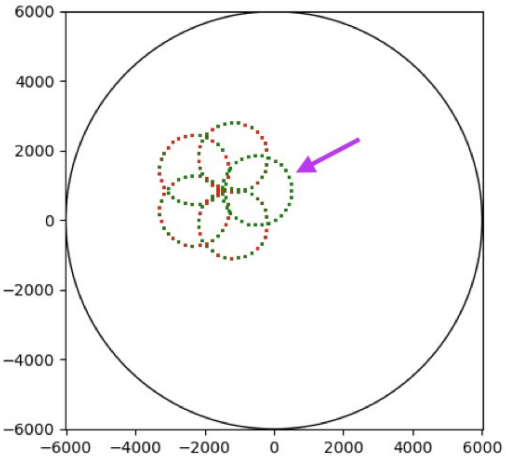
43, 41



31, 29



19 et 5, pas jumeaux



31 et 5, pas jumeaux

Les structures conditionnées par la factorisation de celui qu'on avait appelé un "père de premiers jumeaux" (nombre pair entre eux deux) apparaissent clairement dans la forme et le nombre des "pétales" (par exemple,

une succession rapprochée de deux pétales puis d'un, deux pétales longs, un pétale court, par exemple). Les pétales intégralement verts ou les cercles ne contenant que des points verts correspondent aux nombres divisibles par tel ou tel premier (on les a désignés par des flèches roses dans les $6^{\text{ème}}$ et dernière visualisations ci-dessus). Les "fleurs" (numérotés 1, 3, 4, 5, 6 sur 10 dessins) qui montrent des pétales droites depuis le centre correspondent à des pères de jumeaux divisibles par 4. Les autres dessins, montrant des cercles successifs ressemblant au coeur d'une fleur de tournesol, correspondent à un couple de jumeaux dont le père est un $4k + 2$ (n'est pas divisible par 4).

Bibliographie

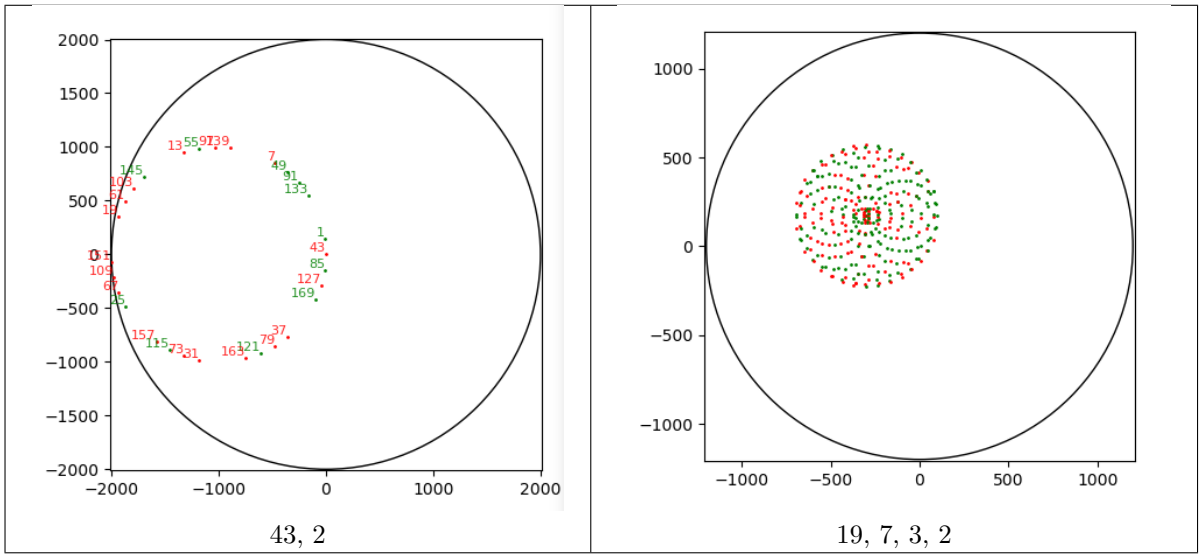
[1] Alain Connes, "A new proof of Morley's theorem", *Publications Mathématiques de l'IHÉS*, **S88** : 43-46, 1998.

[2] Transcription d'une vidéo d'Alain Connes au Collège de France, visionnable ici :https://www.college-de-france.fr/site/colloque-2018/symposium-2018-10-18-10h00.htm, http://denisevellachemla.eu/transc-AC-langage.pdf.

[3] Denise Vella-Chemla, Snurpf, exemple, 2019,http://denisevellachemla.eu/snurpf-exemple.pdf, démonstration de la caractérisation :http://denisevellachemla.eu/demo-caracterisation-DG.pdf.

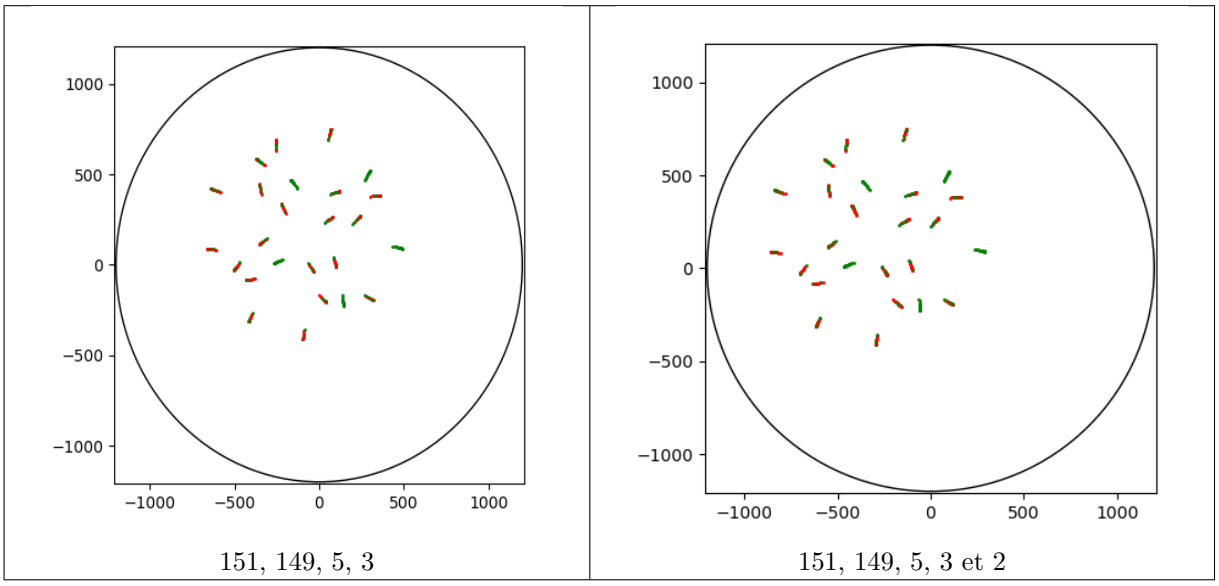
D'autres visualisations issues de rotations sur le disque unité (Denise Vella-Chemla, 31.8.2020)

On fournit d'autres visualisations faisant suite à la note "aligner les premiers" consultable ici : http://denisevellachemla.eu/aligner-les-premiers.pdf.



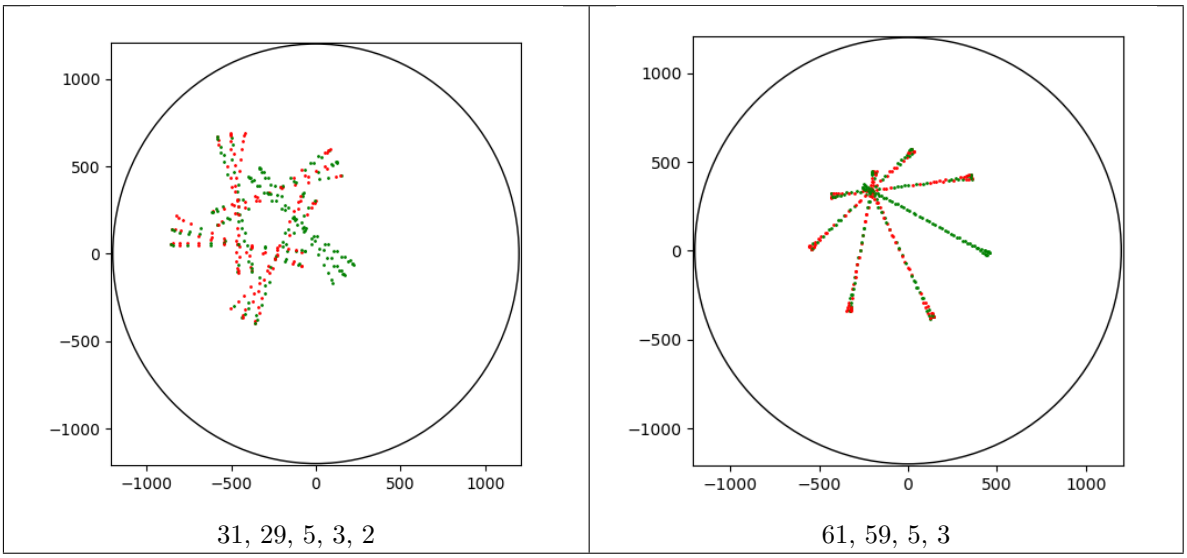
43, 2 | 19, 7, 3, 2

Sur le dessin de gauche n'utilisant que des rotations dans $\mathbb{Z}/43\mathbb{Z}$ et dans $\mathbb{Z}/2\mathbb{Z}$, on peut voir que deux points symétriques l'un de l'autre par rapport à l'axe des abscisses (l'un en haut, l'autre en bas), ont systématiquement pour somme le produit 86=43.2 (ou l'un de ces multiples lorsque les nombres envisagés augmentent).



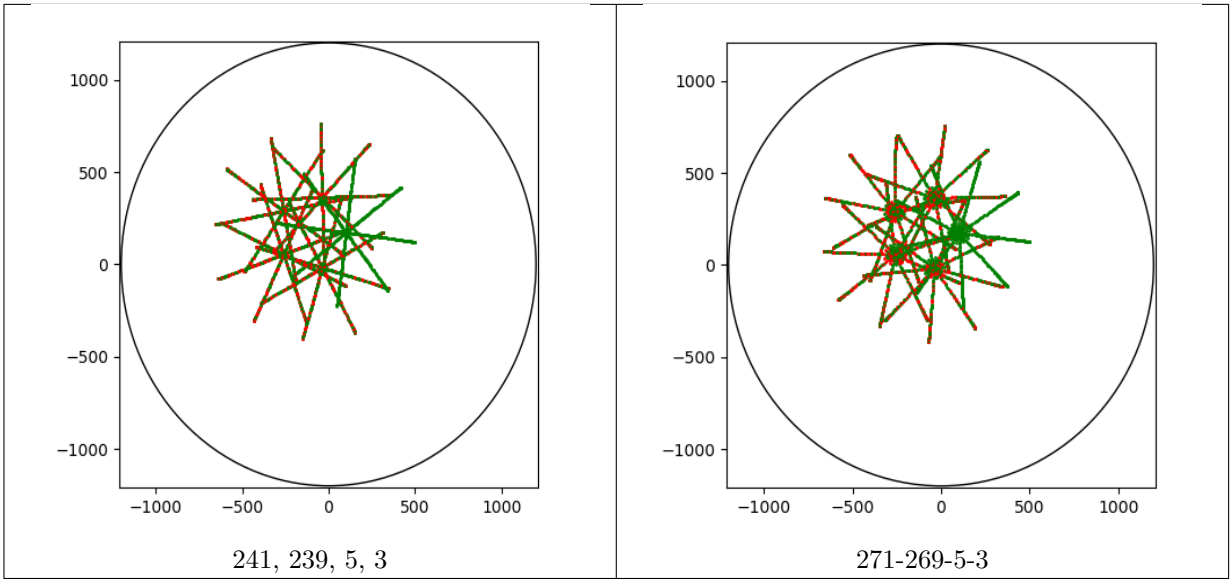
151, 149, 5, 3 | 151, 149, 5, 3 et 2

Ces deux dessins montrent bien une structure décomposée en 4 (compris entre 3 et 5). Le fait de considérer également le nombre premier 2 ou pas ne fait que translater le dessin global.

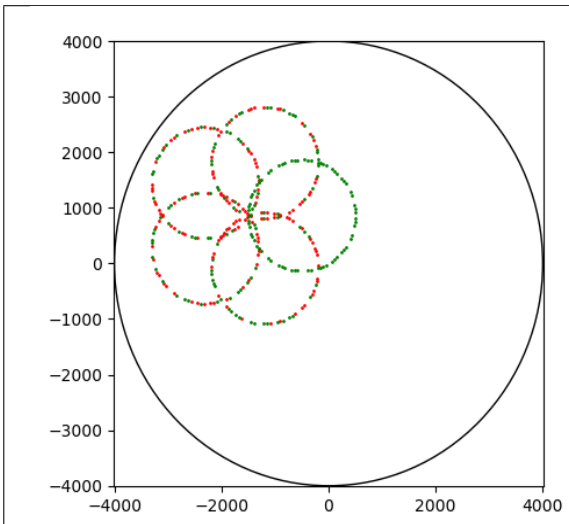


31, 29, 5, 3, 2 | 61, 59, 5, 3

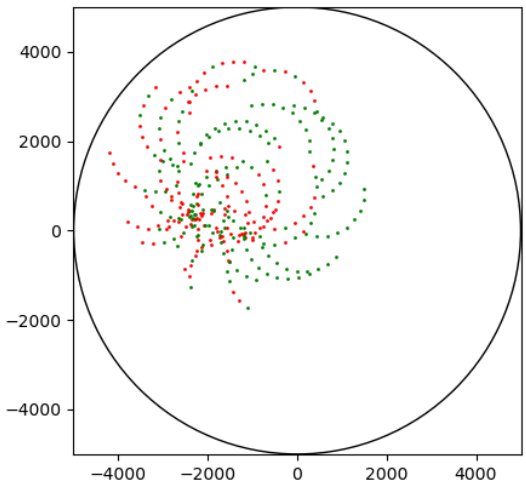
Sur les quatre dessins (2 ci-dessus et 2 ci-après), on distingue clairement les parties toutes vertes (sans aucun nombre premier). De façon générale, la divisibilité correspondant au fait d'avoir un reste nul fait que les concentrations de nombres composés sont dans la partie du dessin la plus à droite.



241, 239, 5, 3 | 271-269-5-3



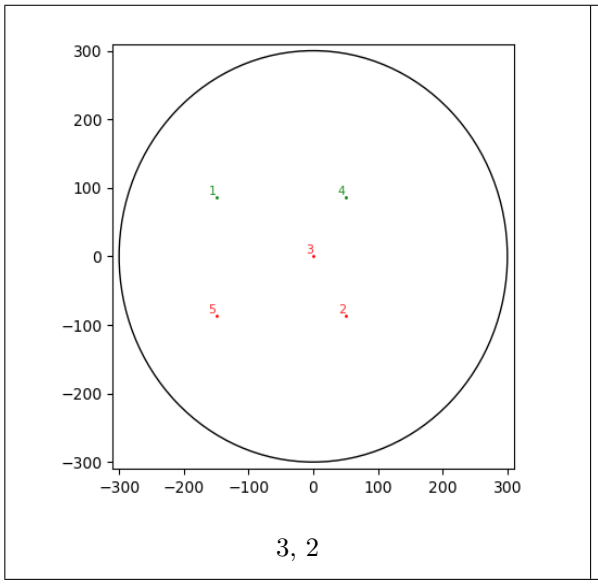
103, 5, 3, 2



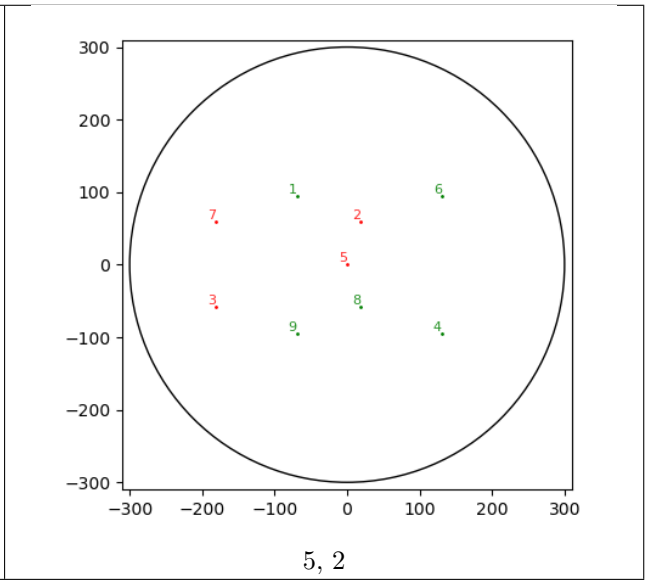
103, 7, 5, 3, 2

Le fait de ne considérer, sur le dessin de gauche, que les nombres premiers 5, 3 et 2, fait apparaître très clairement une structure à 5 cercles ; le dessin de droite montre qu'il suffit de considérer également, en plus des nombres premiers 2, 3 et 5, le nombre premier 7 (en considérant à peine les rotations dans les corps de 5 nombres premiers seulement en tout) pour que toute la structure disparaisse et que le dessin se complexifie complètement.

Voyons maintenant ce que donne les rotations composées d'un seul premier (compris entre $p = 3$ et $p = 23$ [1]) et de 2 :

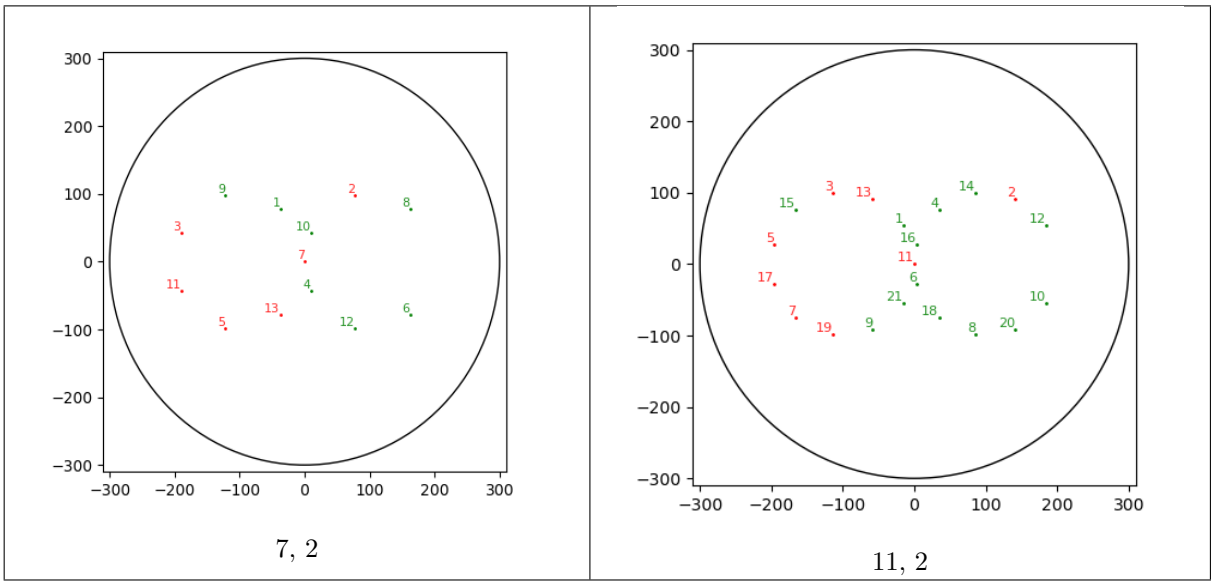


3, 2

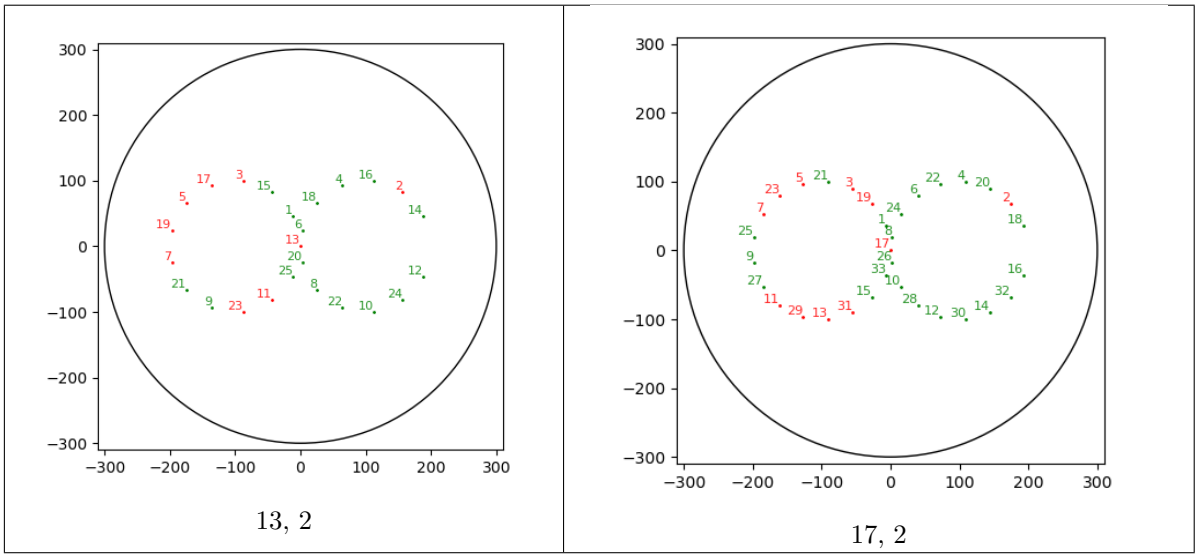


5, 2

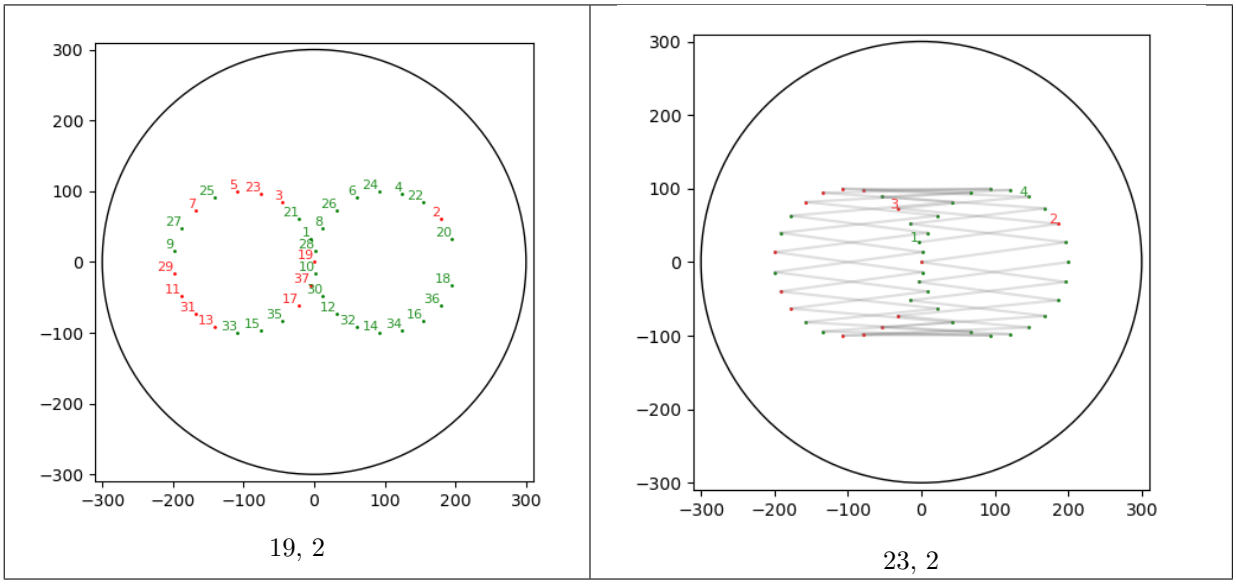
1. parce que l'infini, c'est trop loin.



7, 2 11, 2



13, 2 17, 2



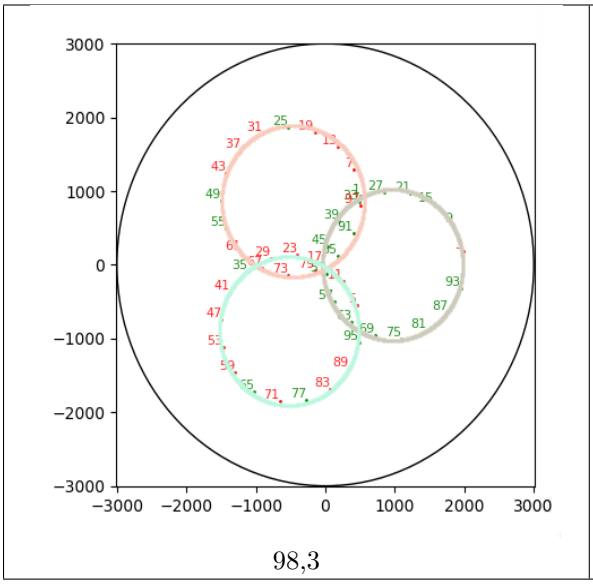
19, 2 23, 2

Les rotations séparent les pairs (cercle de droite) des impairs (cercle de gauche) et symétrisent autour de l'axe des abscisses les nombres complémentaire à $2p$ puisqu'ils ont même reste modulo 2 et ont des restes opposés modulo p. Ce qui est sympathique et étonnant, c'est qu'il semblerait qu'on puisse effectuer les additions selon d'autres directions que la direction verticale et obtenir des multiples. Prenons l'avant dernière sorte de lemniscate pour $p = 19$ en bas à gauche. Dans le cercle de gauche, choisissons d'additionner 11 et

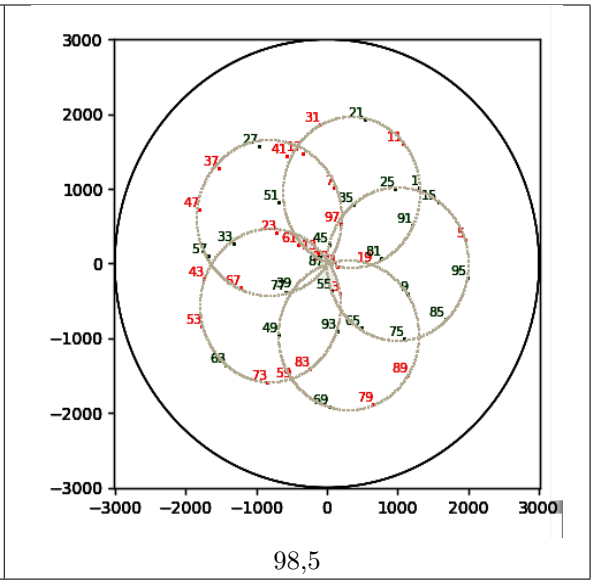
31 au sud-ouest, les doublons successifs sont alors 29+13=9+33=27+15=7+35, etc., tous égaux à 42. Le fait d'avoir représenté les nombres sur des cercles plutôt que par exemple, comme Gauss enfant, pour additionner les 50 premiers nombres, i.e. en quinconce, nous fait appréhender des connaissances nouvelles associées à la forme choisie de la représentation.

Revenons au cas qui nous guide dans nos recherches autour de la conjecture de Goldbach : le nombre pair 98 et ses 3 décomposants de Goldbach 19, 31 et 37, tous ayant un reste différent de celui de 98 lorsqu'on les divise par les nombres premiers impairs inférieurs à la racine carré de 98 que sont 3, 5 et 7.

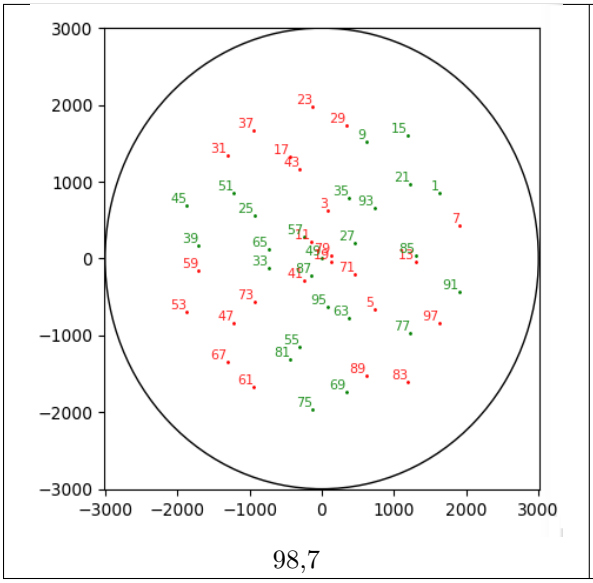
Utiliser les rotations selon $n = 98$ et l'un seulement des 3 nombres premiers 3, 5 ou 7 fournit des visualisations très compréhensibles. On a matérialisé sur le premier dessin les cercles contenant les $6x + 1$, $6x + 3$ et $6x + 5$ ou bien sur le deuxième dessin, les $10x + k$ mais on n'a pas noté les cercles correspondant aux $14x + k'$ sur le troisième dessin. Dès qu'on passe à deux nombres premiers plutôt qu'un, les visualisations sont beaucoup plus embrouillées. On montre également sur les autres dessins les "chemins" qui font passer d'un nombre au suivant, ce qui fait bien appréhender la dynamique associée aux rotations.



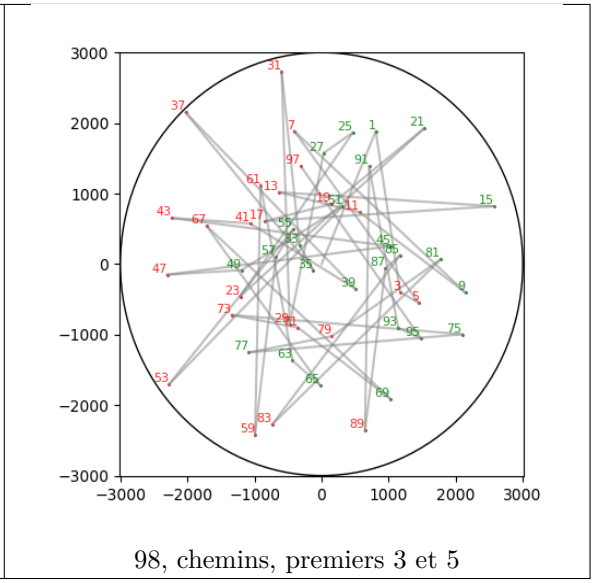
98,3



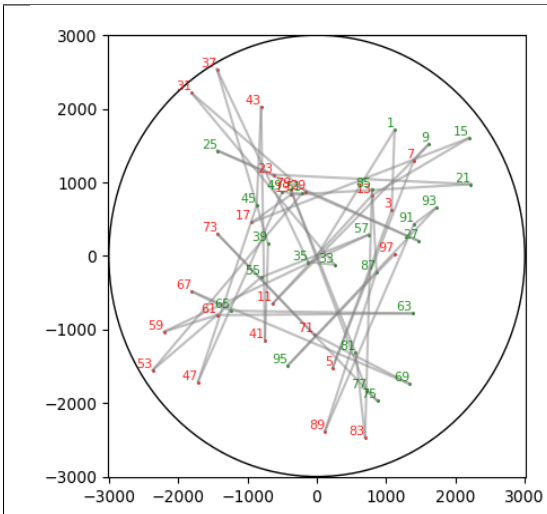
98,5



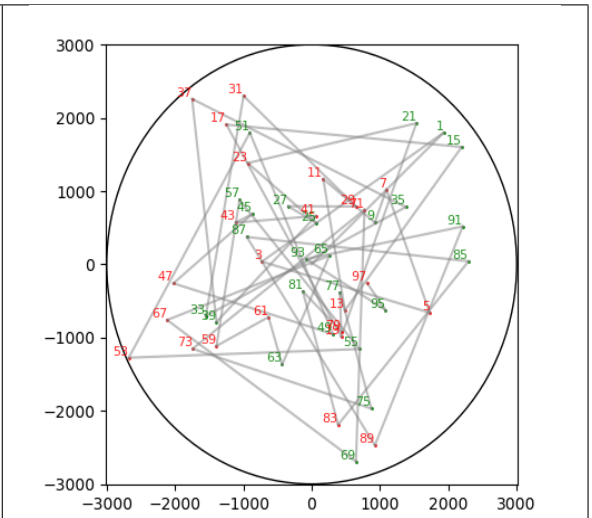
98,7



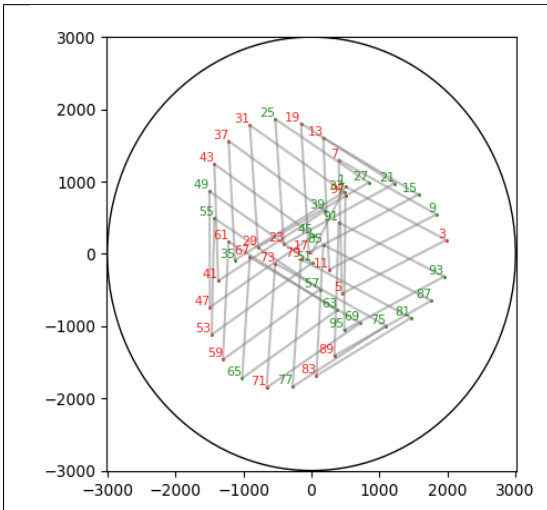
98, chemins, premiers 3 et 5



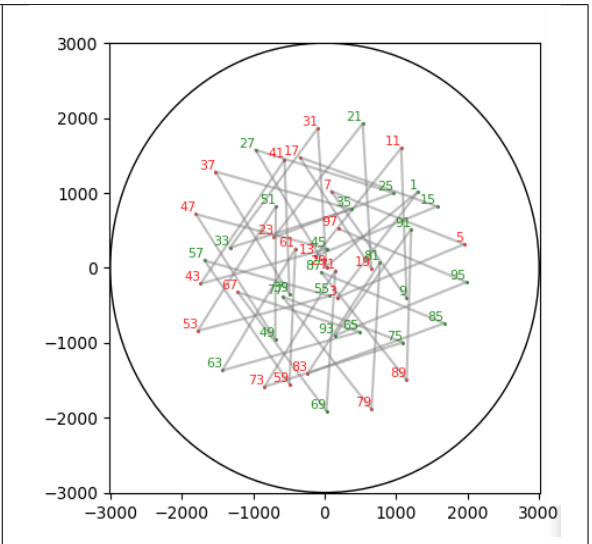
98, chemins, premiers 3 et 7



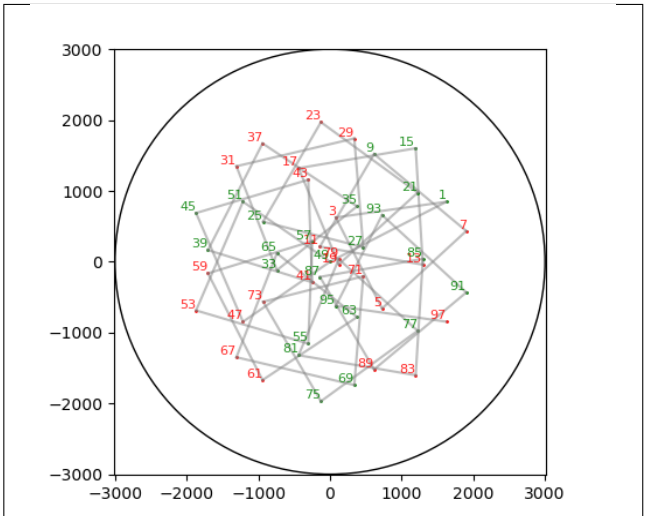
98, chemins, premiers 5 et 7



98, chemins, seul premier 3



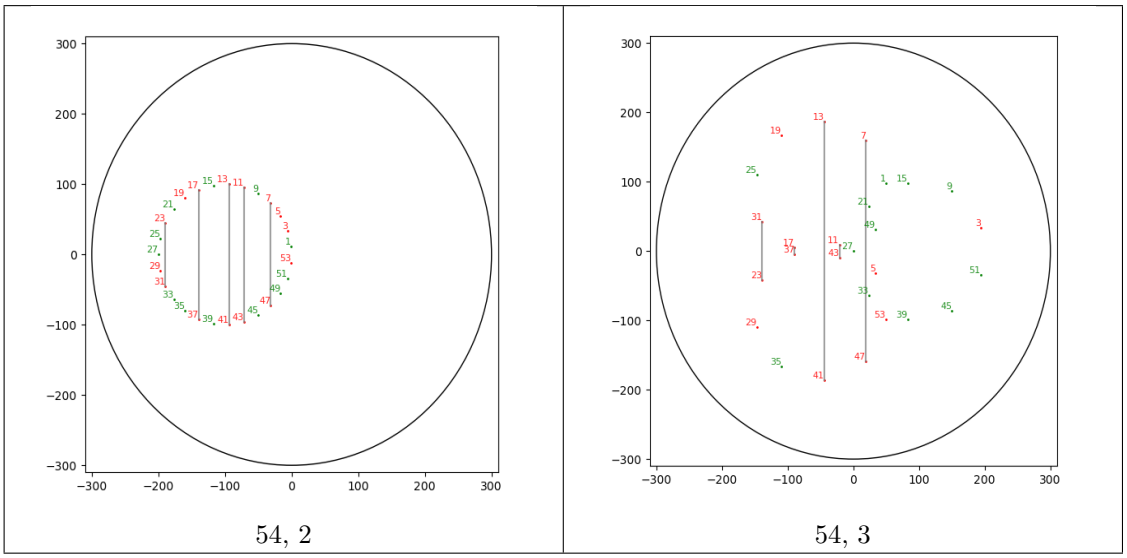
98, chemins, seul premier 5



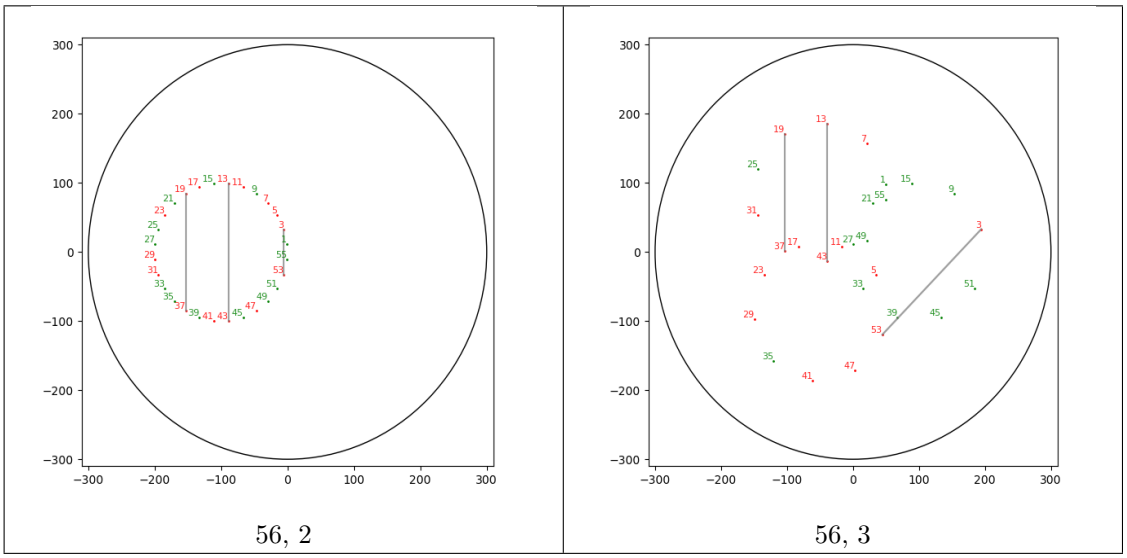
98, chemins, seul premier 7

Mercredi 2 septembre 2020

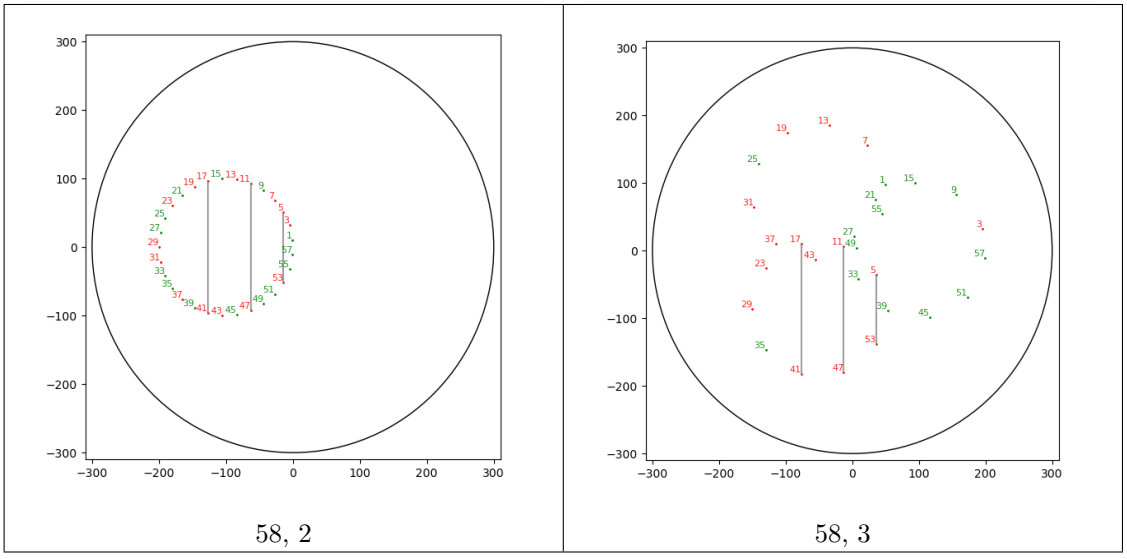
On peut observer la manière dont le passage de la combinaison avec une rotation modulo 2 (qui ne fait que séparer les pairs des impairs) à une rotation de $e^{\frac{2i\pi}{3}}$ (équivalente à l'addition dans $\mathbb{Z}/3\mathbb{Z}$) fait "bouger" les décompositions de Goldbach, matérialisées par des traits gris entre décomposants.



54, 2 | 54, 3

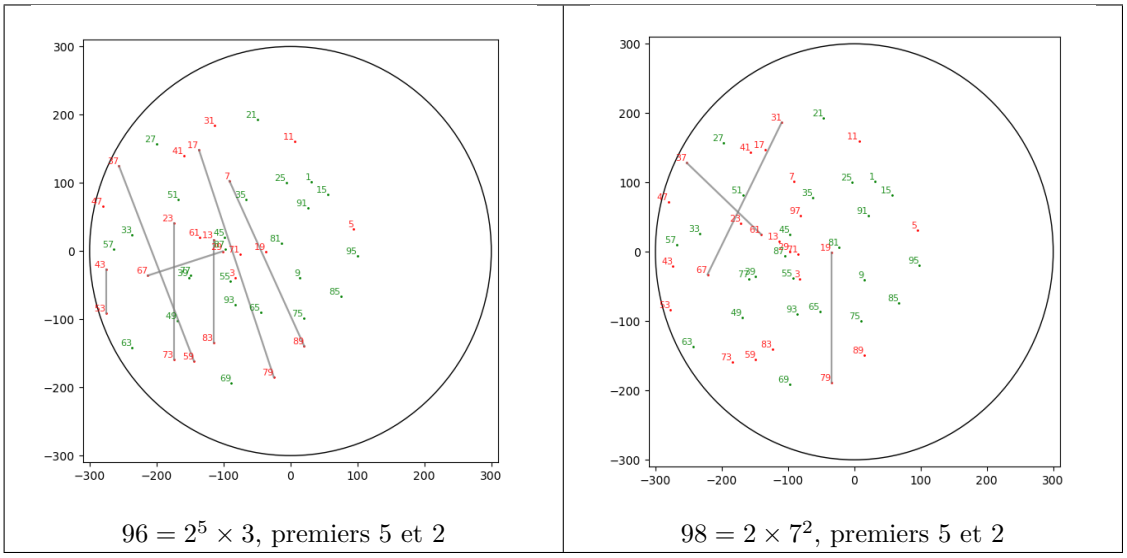


56, 2 | 56, 3



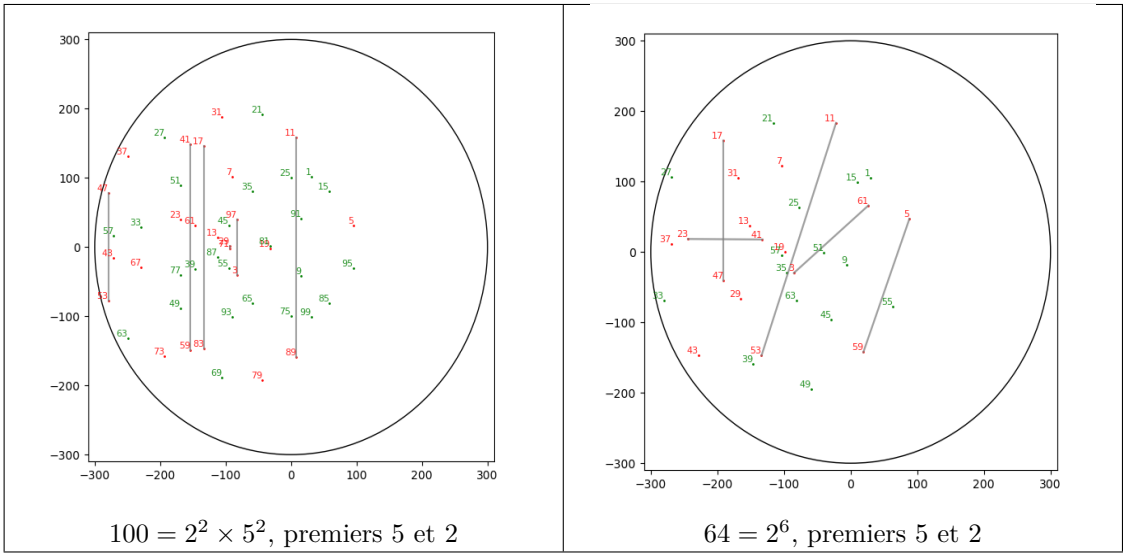
58, 2 | 58, 3

Cela fait comprendre l'effet qu'aura le fait de ne garder que des diviseurs de n, les décompositions de Goldbach seront alors toutes "parallèles".



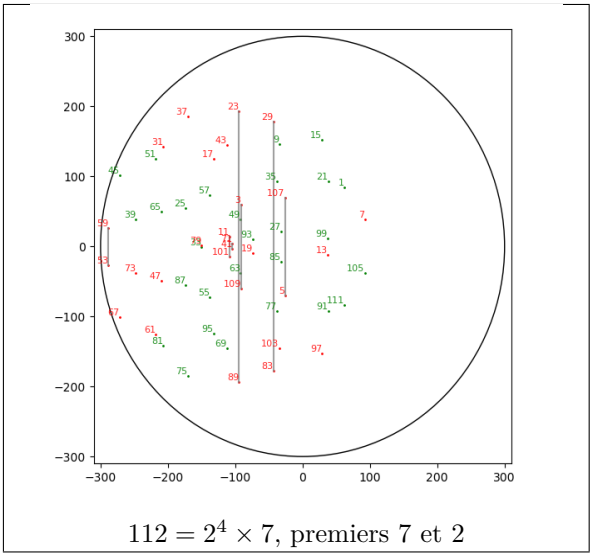
$96 = 2^5 \times 3$, premiers 5 et 2

$98 = 2 \times 7^2$, premiers 5 et 2



$100 = 2^2 \times 5^2$, premiers 5 et 2

$64 = 2^6$, premiers 5 et 2



$112 = 2^4 \times 7$, premiers 7 et 2

Sommes de complexes sur disque unité (Denise Vella-Chemla, 9.9.2020)

Dans le but de comprendre l'espace des nombres premiers, mais surtout dans le but de comprendre la raison pour laquelle tout nombre pair (supérieur à 4) est la somme de deux nombres premiers, on continue de programmer un certain nombre de visualisations sur le disque-unité [1].

On choisit un ensemble de nombres entiers $E = \{n_1, n_2, \ldots, n_k\}$ et on représente par des points sur un disque "normalisé", pour chaque nombre entier x inférieur à une certaine valeur, sa somme de complexes associée :

$$f(x) = \sum_{k=1}^{k=|E|} e^{\frac{2i\pi(x \bmod n_k)}{n_k}}$$

L'ensemble E étant fini, du fait de phénomènes de périodicité constatables et explicables, des nombres périodiquement espacés ont même image par f. Dit autrement, la trajectoire, si on considère qu'un doigt imaginaire passe du point représentant un entier au point représentant l'entier suivant, se referme, et est parcourue à nouveau périodiquement. Comme on choisit informatiquement de représenter cette trajectoire avec ce qu'on appelle des couleurs transparentes, on voit la trajectoire devenir plus foncée plus elle est parcourue de fois.

On représente les nombres premiers en rouge, les nombres composés en vert. Les figures 1 et 2 ci-dessous montrent les images des nombres jusqu'à 123 à gauche, en "agrégeant les restes" selon les nombres premiers jumeaux 11 et 13. La visualisation pour les nombres jusqu'à 1000 (à droite) est identique mais plus foncée, la trajectoire a été parcourue plusieurs fois. Par exemple, 1 et 144 ayant même image dans $\mathbb{Z}/11\mathbb{Z}$ et dans $\mathbb{Z}/13\mathbb{Z}$, leur image est la même et la rosace se referme. La "rosace" a été parcourue de nombreuses fois en passant par les mêmes points à droite. On voit qu'elle frôle le point origine $(0, 0)$ au centre.

Sur la figure 3, on a souhaité montrer la forme d'une demi-trajectoire, il y a passage par l'origine symbolisée par une étoile. La figure 4 est destiné à faire comprendre la symétrie haut-bas du dessin : les restes ayant été pris selon les deux nombres premiers jumeaux 5 et 7, on voit que les images des nombres dont la somme est 35=5.7 sont symétriques par rapport à l'axe des abscisses.

Sur la figure 5, comme attendu, dans la mesure où on a constaté que les restes selon deux nombres premiers jumeaux permettent d'obtenir une symétrie simple, on a l'idée de prendre des nombres premiers dont la différence est égale à 8, ici 11 et 19, pour obtenir une rosace à 8 symétries.

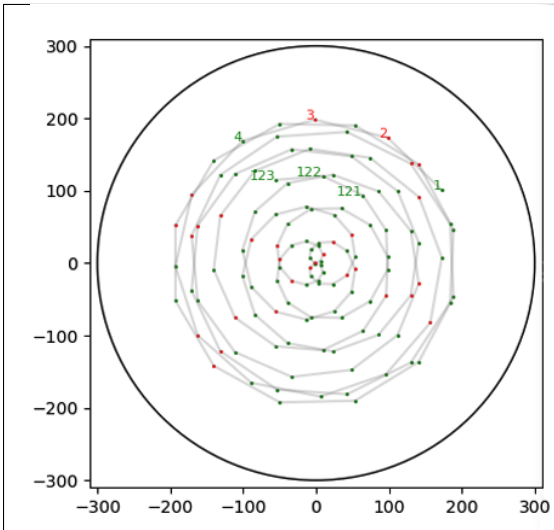


Fig. 1 : E={11,13} jusqu'à 123.

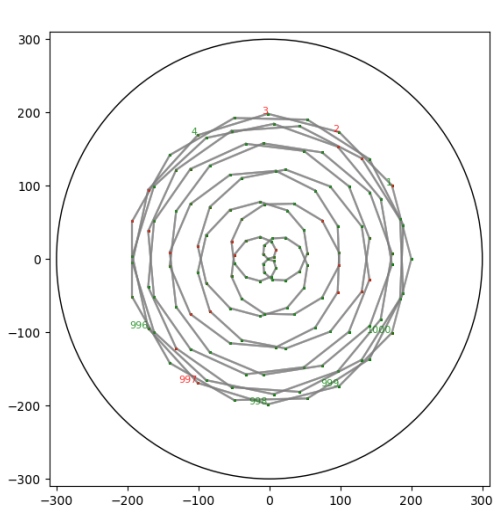


Fig. 2 : E={11,13} jusqu'à 1000.

1. On a laissé un "écart" entre le bord des visualisations et le cercle censé représenter le disque-unité pour des raisons pratiques, on peut toujours normaliser, ici on a pris comme rayon du cercle deux fois et demi le nombre de nombres premiers utilisés pour les calculs de complexes.

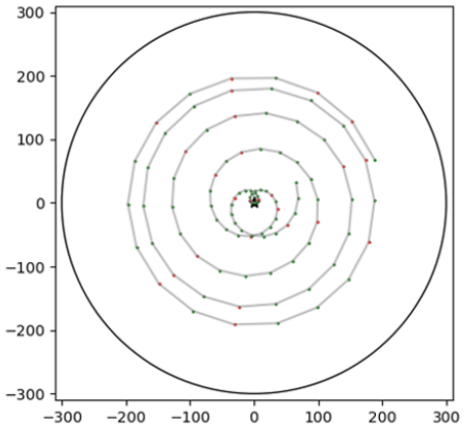


Fig. 3 : E={17,19} jusqu'à 100.

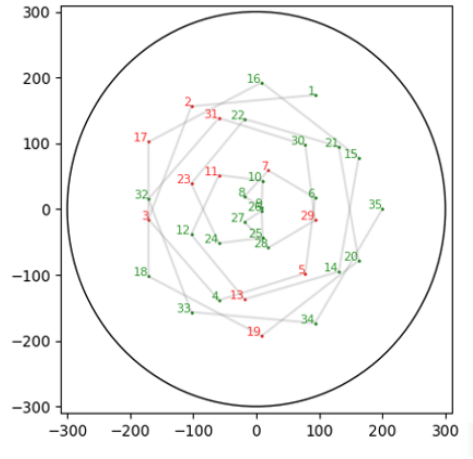


Fig. 4 : E={5,7} jusqu'à 35.

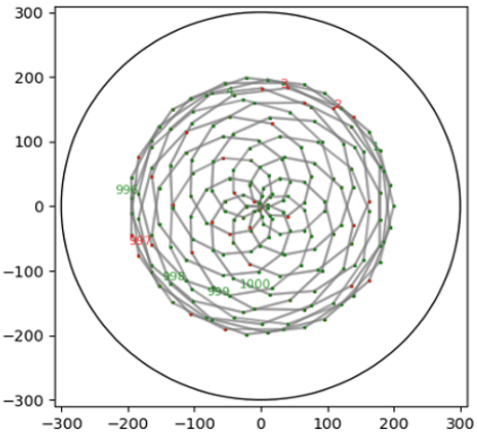


Fig. 5 : E={11,19} périodicité 8.

Sur les figures 6 et 7, on visualise selon $15 = 3.5$ et $21 = 3.7$ et la périodicité est de 105. On a utilisé deux couleurs différentes pour la moitié initiale de la rosace (en gris) et la seconde moitié (en orange). Ces courbes ressemblent à des cardioides.

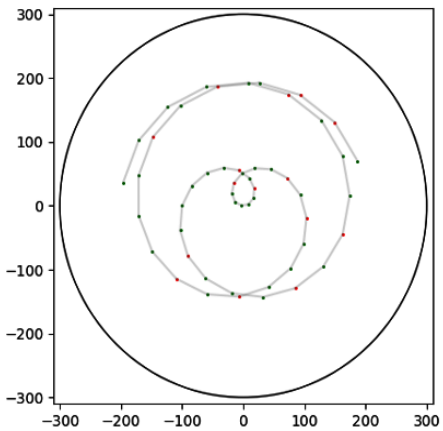


Fig. 6 : E={15,21}

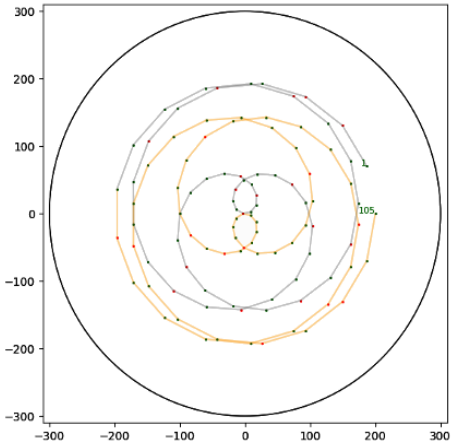


Fig. 7 : E={15,21}, voir la symétrie

Les figures 8 et 9, restes calculés selon 3 et $27 = 3^3$ ou bien selon 5 et $125 = 5^3$ montrent bien sur celle de gauche un “triangle de triangles emboîtés”, c'est-à-dire que les formes sont suffisamment triangulaires pour que nous les visualisions bien mais tout de même insuffisamment triangulaires pour qu'il y ait globalement un déplacement ne ramenant à l'origine qu'en 27 coups. La figure de droite montre de la même façon des pentagones dont les sommets sont sur les 5 cercles.

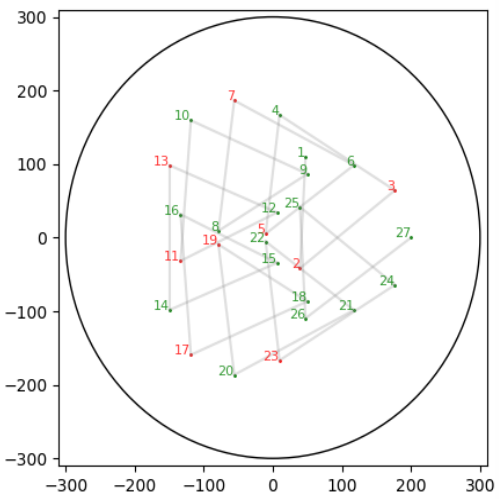


Fig. 8 : E={3,27} jusqu'à 27.

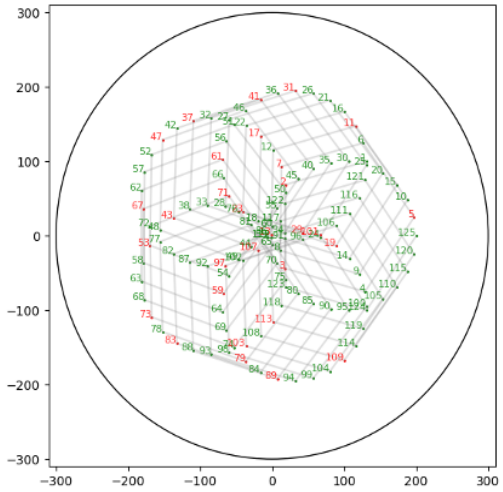


Fig. 9 : E={5,125} jusqu'à 125.

La figure 10, restes calculés selon les nombres 4, 3 et 5, montre le retour sur 1 au bout de 60 avec des sortes de “triangles-carrés-pentagones”. Les sommes de deux nombres de part et d'autre de l'axe des abscisses (l'un en haut et son symétrique en bas) valent 60=4.3.5 ou l'un de ses multiples (par exemple 61+59=120).

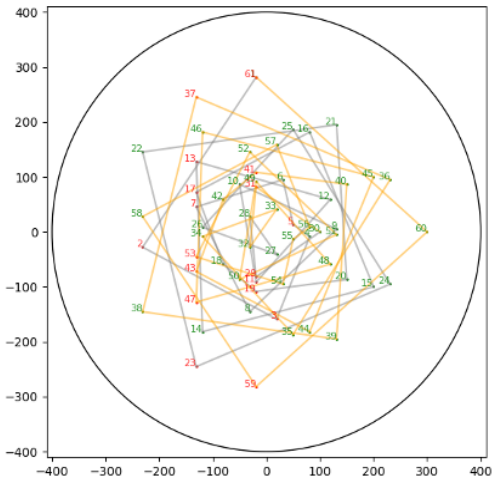


Fig. 10 : E={4,3,5} jusqu'à 61 qui revient sur 1.

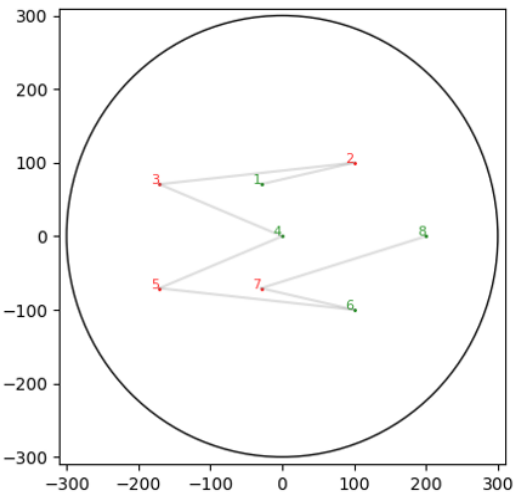


Fig. 11 : E={2,8} jusqu'à 8.

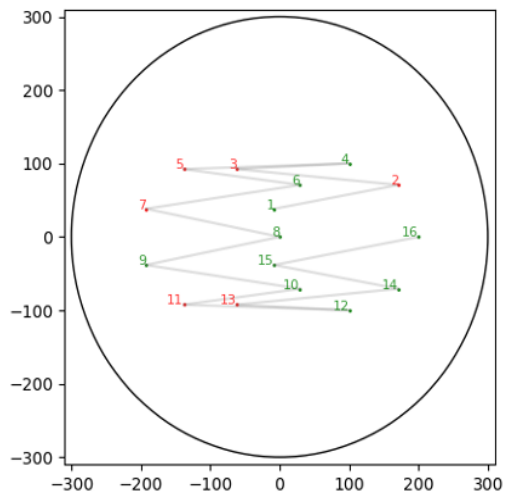


Fig. 12 : E={2,16} jusqu'à 16.

Sur les figures 11 et 12, avec des restes selon des nombres puissances de 2 (2 et 8, ou 2 et 16), comme sur un écheveau, on passe de nombre en nombre de gauche à droite et inversement, et le nombre de "replis" est de plus en plus élevé plus la puissance de 2 choisie augmente.

Enfin, la Figure 13 montre qu'on n'obtient pas le même résultat en calculant les restes selon $E = \{4, 6\}$ ou selon $E = \{2, 2, 3, 3\}$ (24 a 3 décompositions de Goldbach, 5+19, 7+17, 11.13). Cependant, l'addition étant commutative dans le corps des complexes, les résultats obtenus sont identiques indépendamment de l'ordre dans lequel sont utilisés les restes modulaires.

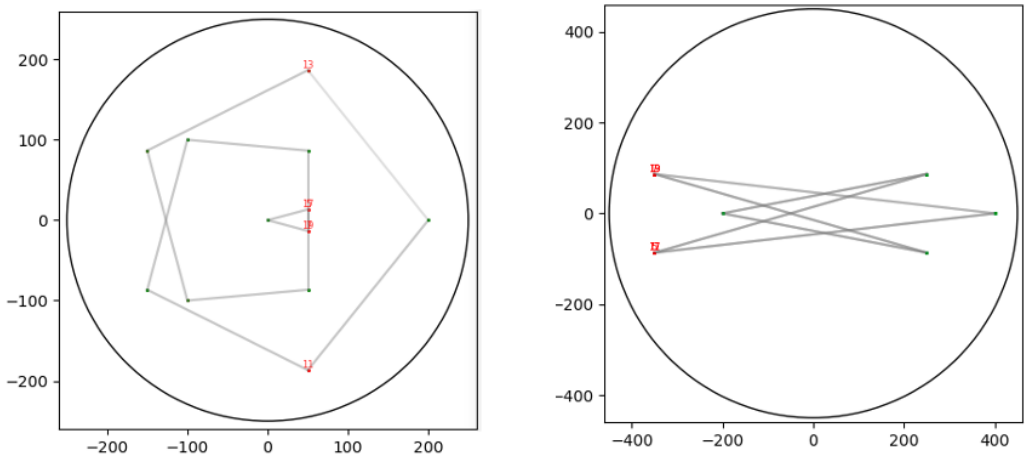


Fig. 13 : Pour $n = 24$, selon que l'on prend E={4,6} à gauche et E={2,2,3,3} à droite, le résultat est différent.

Revenons maintenant à notre cas emblématique du nombre 98, qui a 3 décompositions de Goldbach, 19+79, 31+67 et 37+61. Voyons si les décompositions se placent de façon particulière sur les "spirales". On voit sur le dessin de droite que l'une des 3 décompositions est très éloignée sur la gauche.

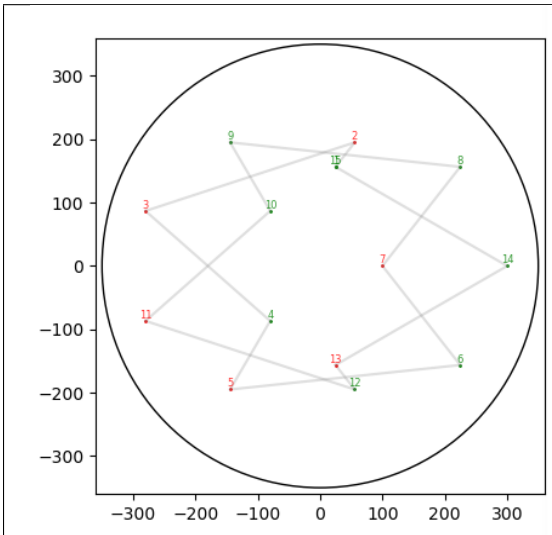


Fig. 14 : E={2,7,7} pour 98 jusqu'à 14.

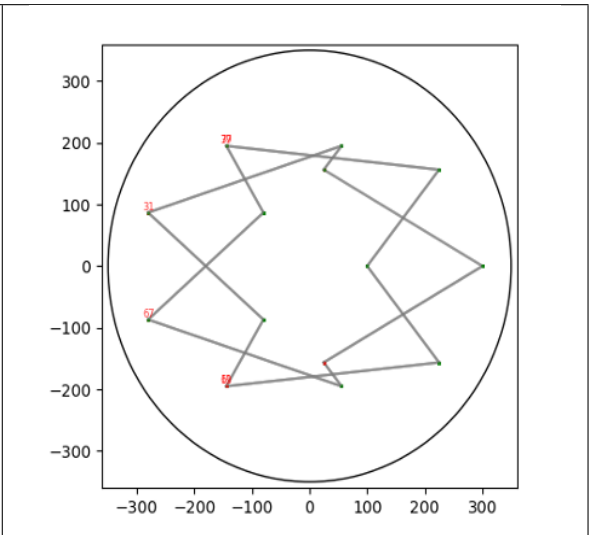


Fig. 15 : E={2,7,7} pour 98 plus "loin".

On rappelle les visualisations associées en utilisant les seuls restes selon les nombres premiers 2 ou 3, avec ou sans chemin de passage entre les nombres successifs.

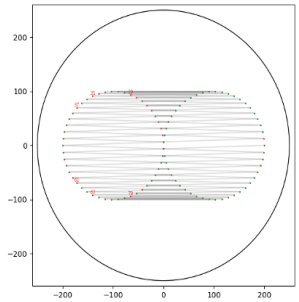


Fig. 16 : E={2, 98} décompositions sur deux cercles avec chemins entre nombres successifs.

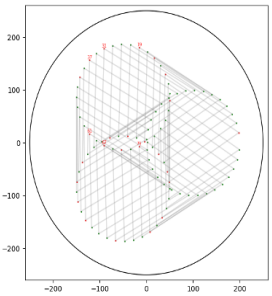


Fig. 17 : E={3, 98} décompositions sur trois cercles avec chemins entre nombres successifs.

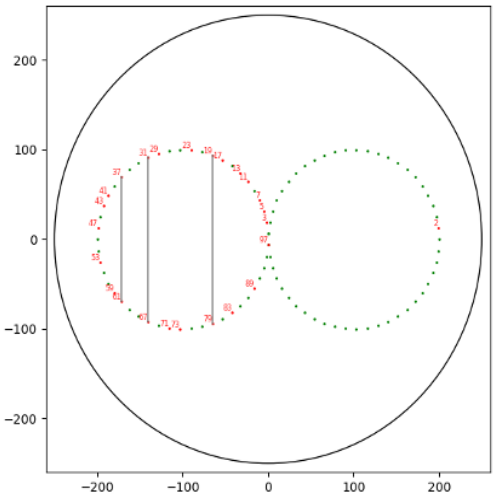


Fig. 18 : E={2, 98} décompositions sur deux cercles.

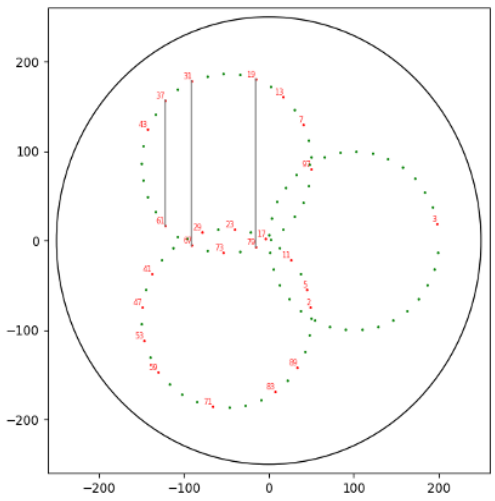
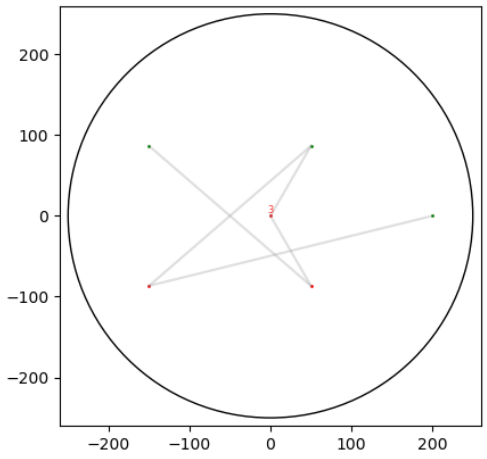
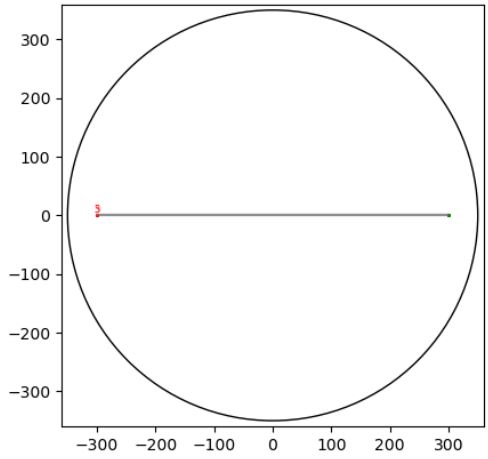


Fig. 19 : E={3, 98} décompositions sur deux cercles.

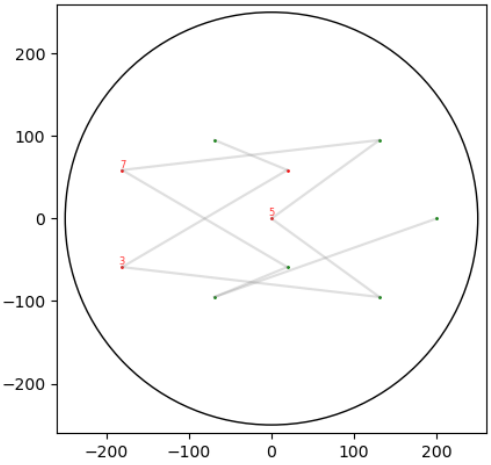
Décomposants de Goldbach : visualisations par sommes de complexes sur disque unité (Denise Vella-Chemla, 9.9.2020)



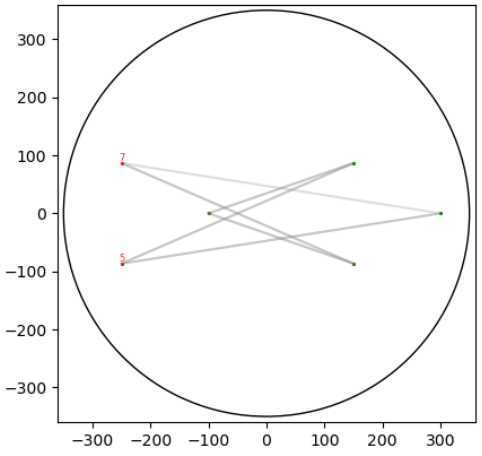
n = 6 (2p) ; E = {2,3}



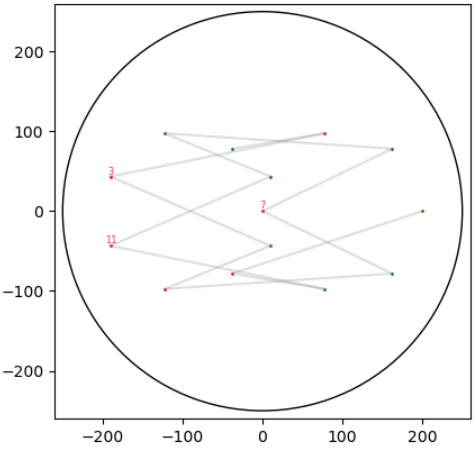
n = 8 (2^k) ; E = {2,2,2}



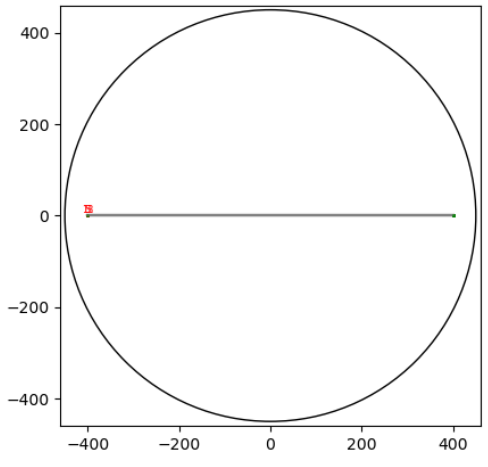
n = 10 (2p) ; E = {2,5}



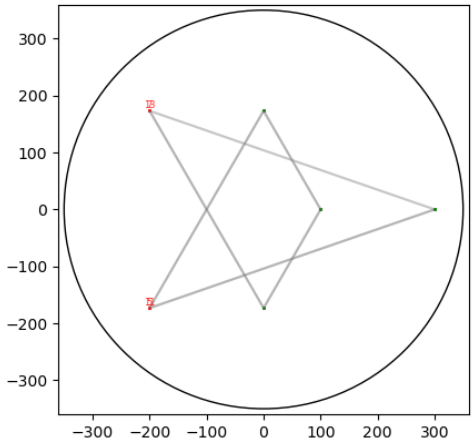
n = 12 (2^kp) ; E = {2,2,3}



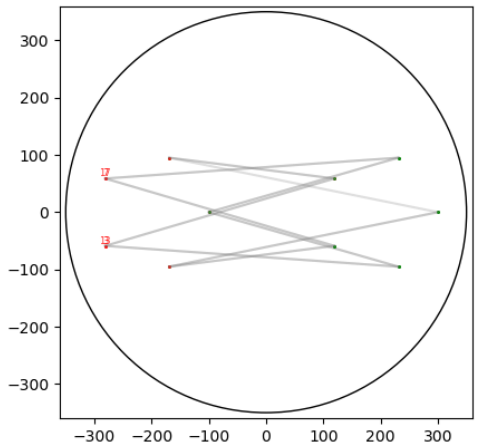
n = 14 (2p) ; E = {2,7}



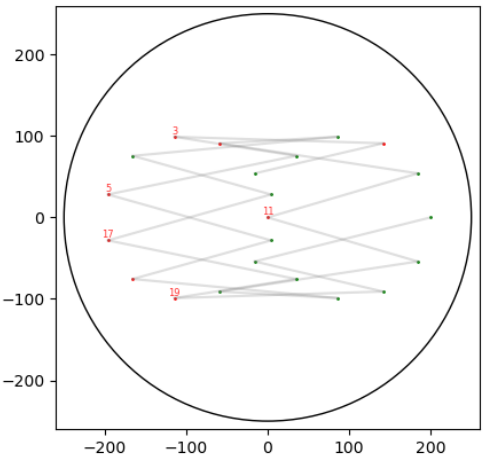
n = 16 (2^k) ; E = {2,2,2,2}



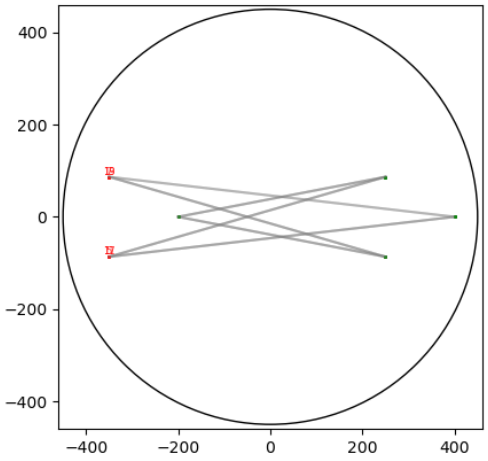
n = 18 ; E = {2,3,3}



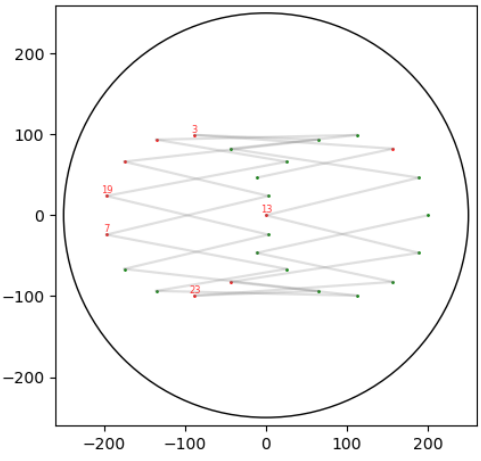
n = 20 $(2^k p)$; E = {2,2,5}



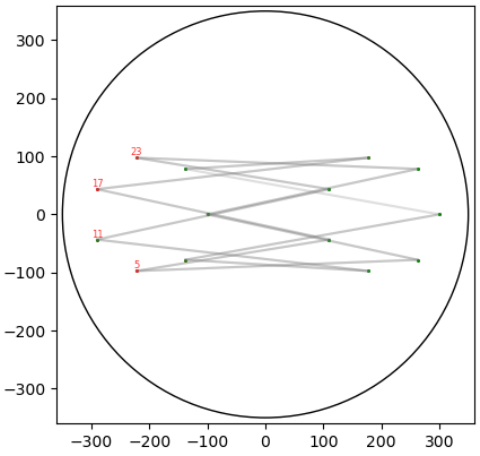
n = 22 $(2p)$; E = {2,11}



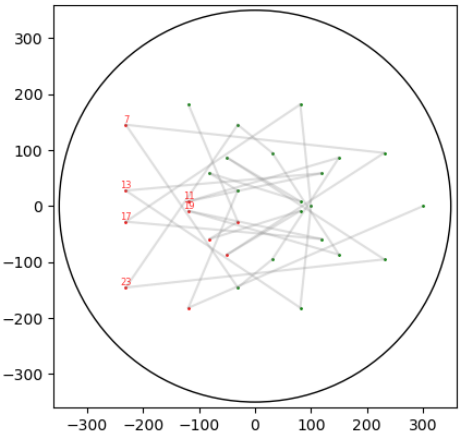
n = 24 $(2^k p)$; E = {2,2,2,3}



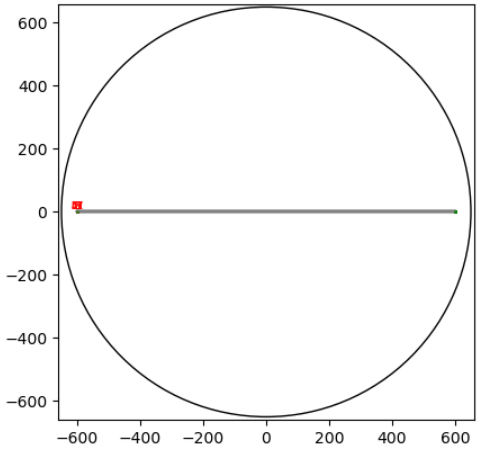
n = 26 $(2p)$; E = {2,13}



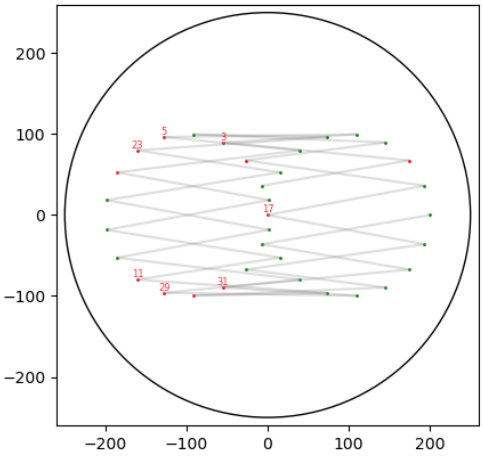
n = 28 $(2^k p)$; E = {2,2,7}



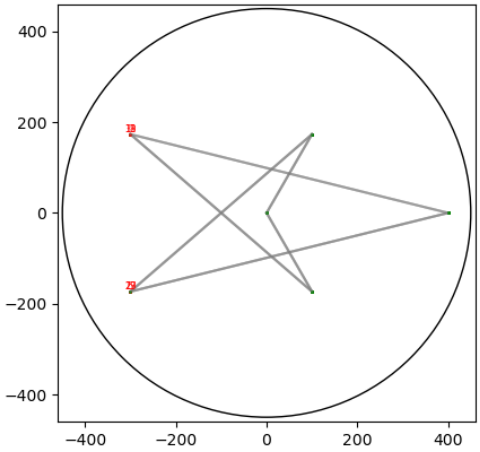
n = 30; E = {2,3,5}



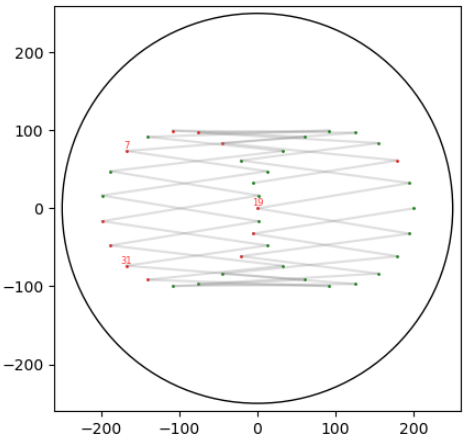
n = 32 (2^k) ; E = {2,2,2,2,2}



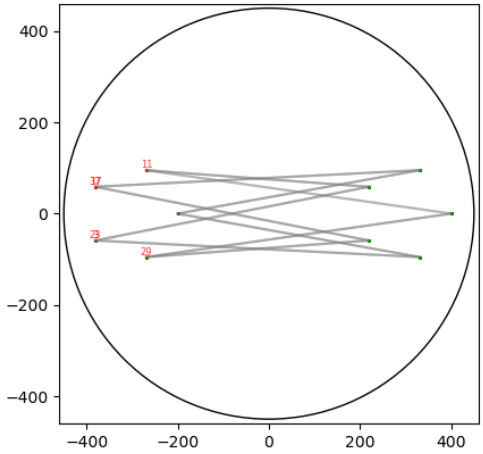
n = 34 (2p); E = {2,17}



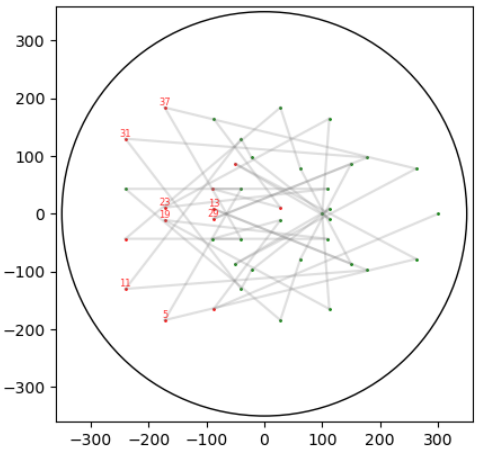
n = 36; E = {2,2,3,3}



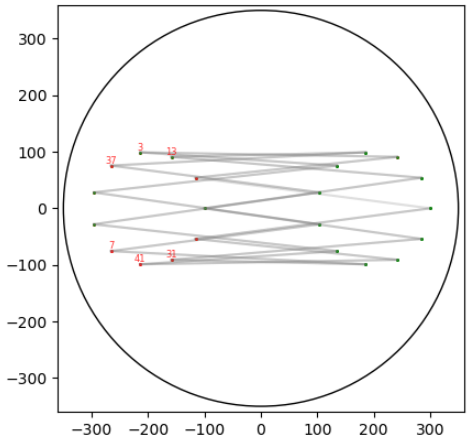
n = 38 (2p); E = {2,19}



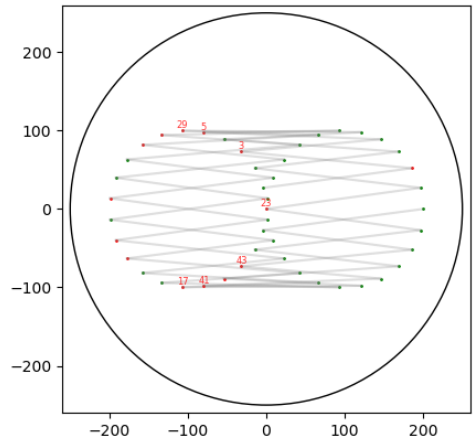
n = 40 (2^kp); E = {2,2,2,5}



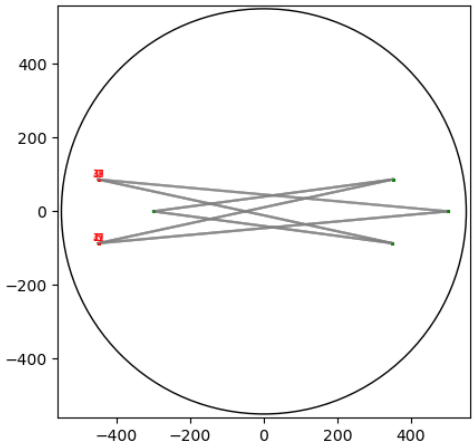
n = 42 ; E = {2,3,7}



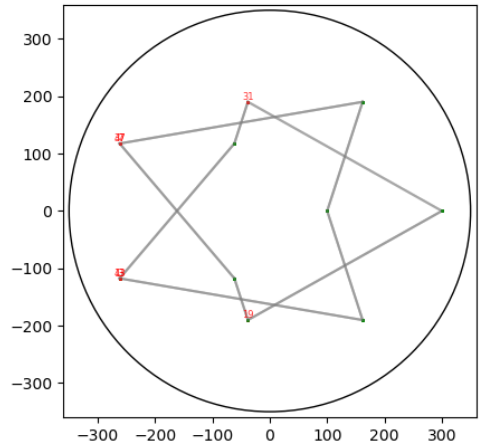
n = 44 (2^kp) ; E = {2,2,11}



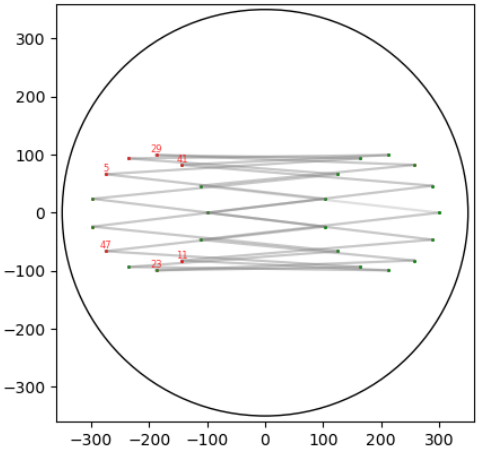
n = 46 (2p) ; E = {2,23}



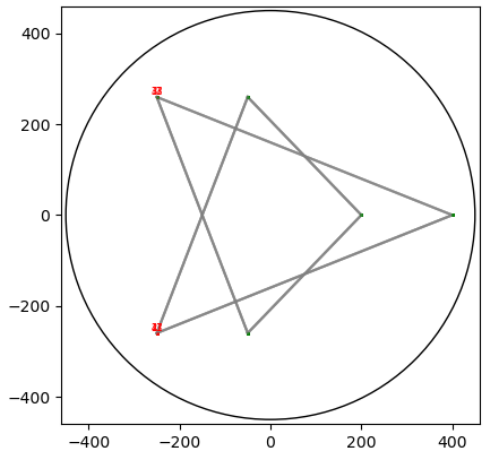
n = 48 (2^kp) ; E = {2,2,2,2,3}



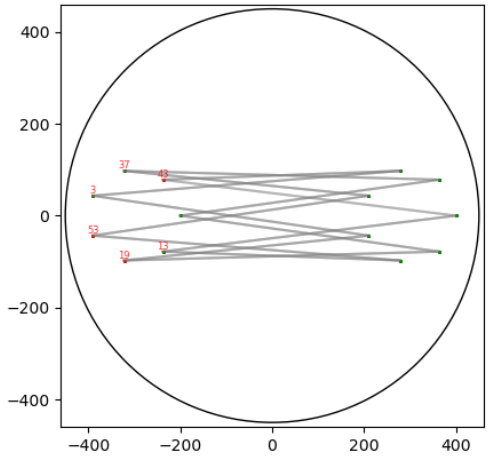
n = 50 ; E = {2,5,5}



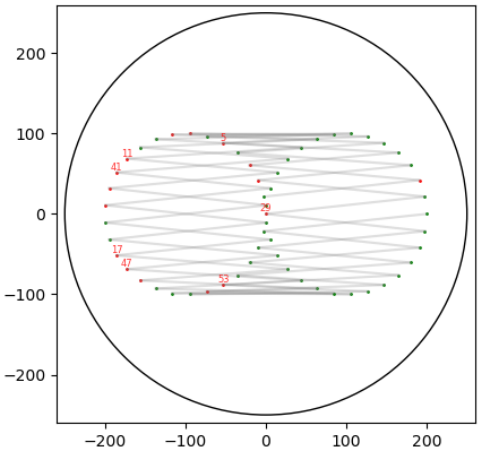
n = 52 (2^kp) ; E = {2,2,13}



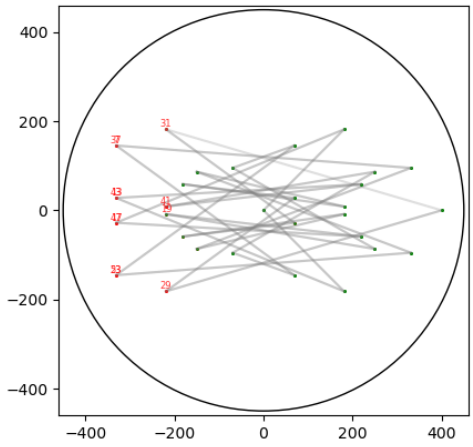
n = 54 ; E = {2,3,3,3}



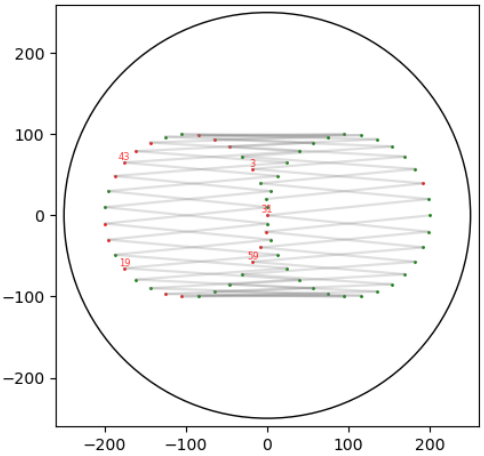
n = 56 (2^kp) ; E = {2,2,2,7}



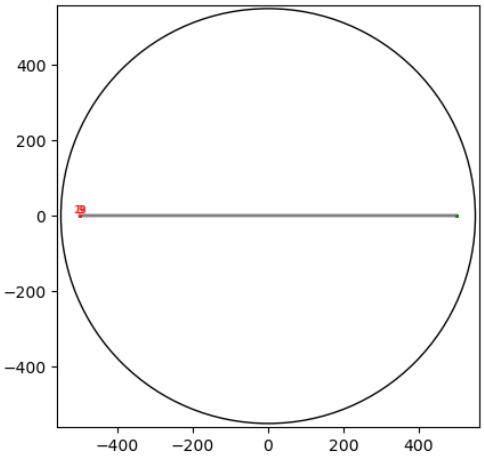
n = 58 (2p) ; E = {2,29}



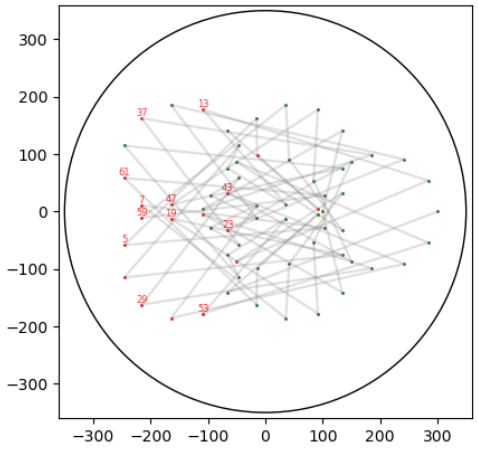
n = 60 ; E = {2,2,3,5}



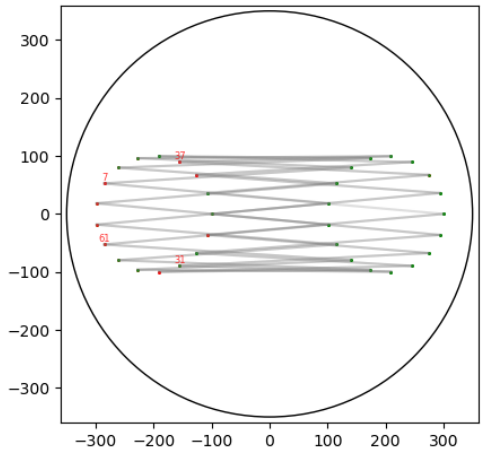
n = 62 (2p) ; E = {2,31}



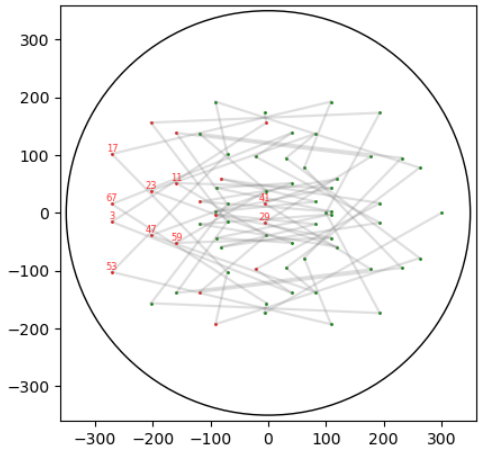
n = 64 (2^k) ; E = {2,2,2,2,2,2}



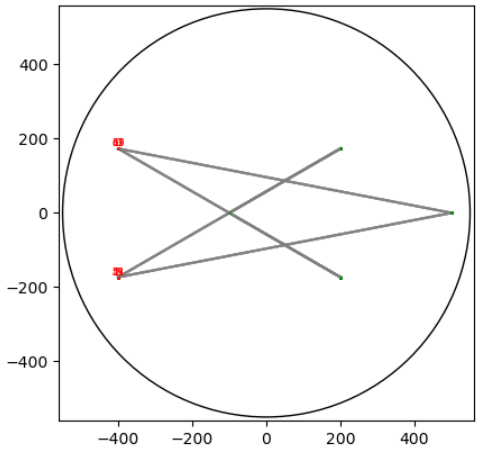
n = 66; E = {2,3,11}



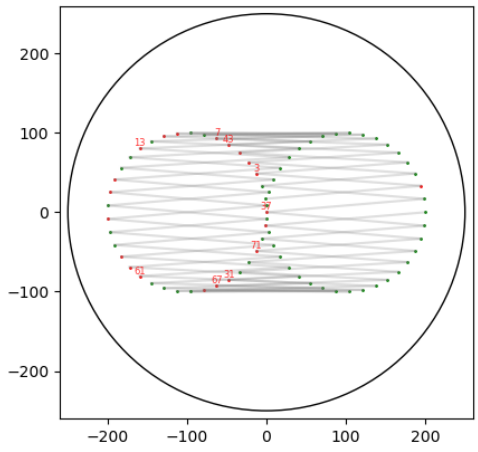
n = 68 (2^kp); E = {2,2,17}



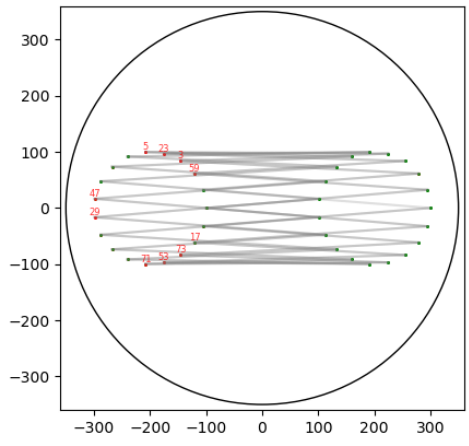
n = 70; E = {2,5,7}



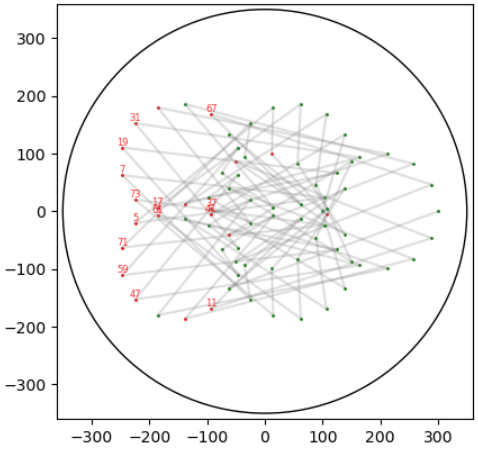
n = 72; E = {2,2,2,3,3}



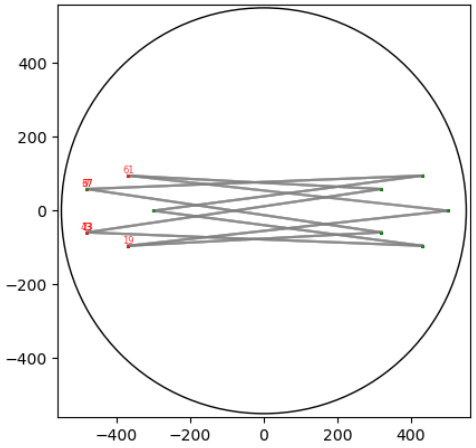
n = 74 (2p); E = {2,37}



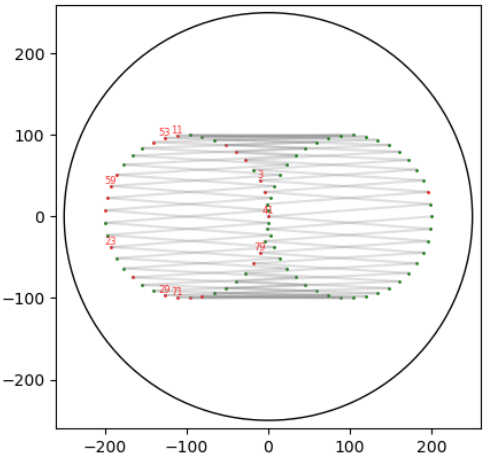
n = 76 (2^kp); E = {2,2,19}



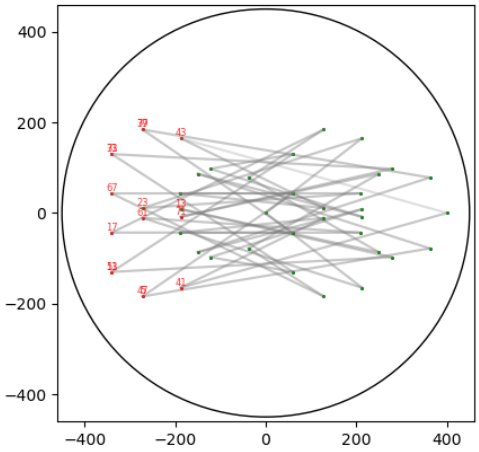
n = 78 ; E = {2,3,13}



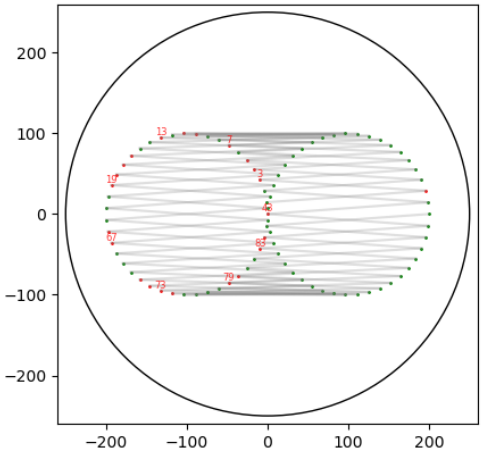
n = 80 (2^kp) ; E = {2,2,2,2,5}



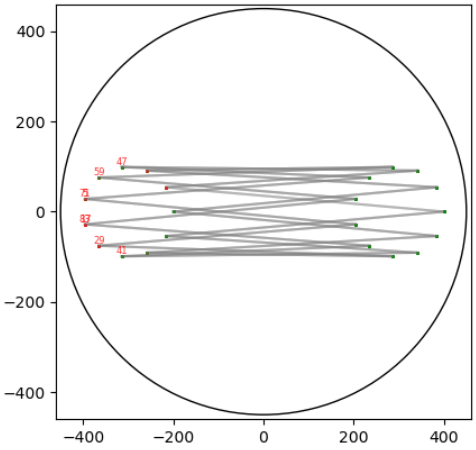
n = 82 (2p) ; E = {2,41}



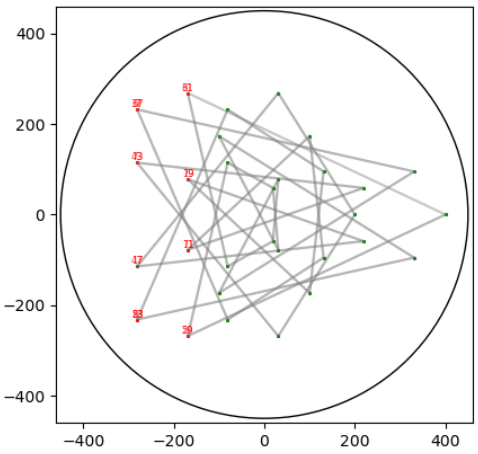
n = 84 ; E = {2,2,3,7}



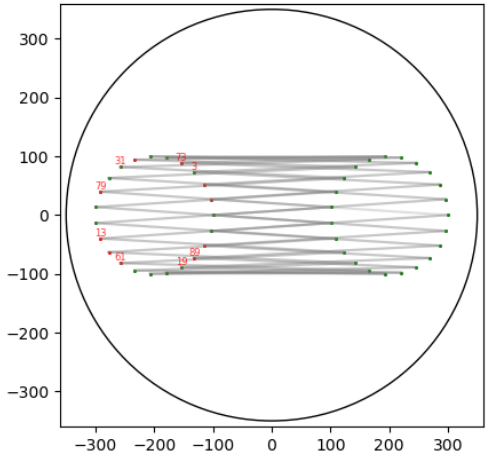
n = 86 (2p) ; E = {2,43}



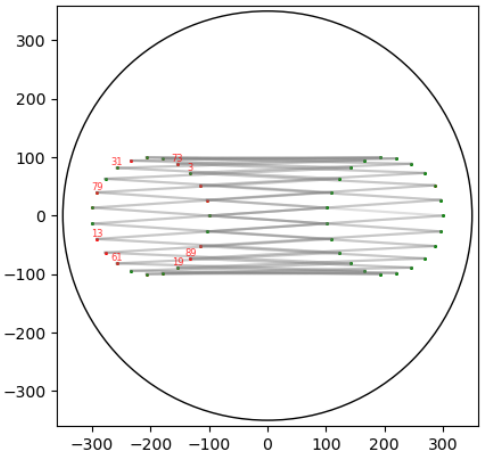
n = 88 (2^kp) ; E = {2,2,2,11}



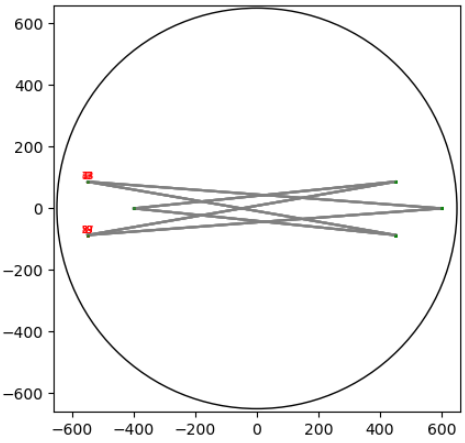
n = 90; E = {2,3,3,5}



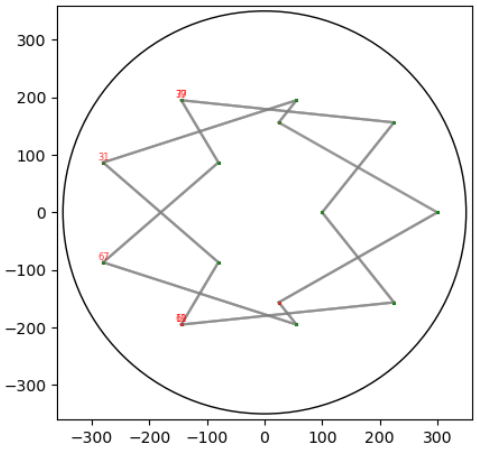
n = 92 (2^kp); E = {2,2,23}



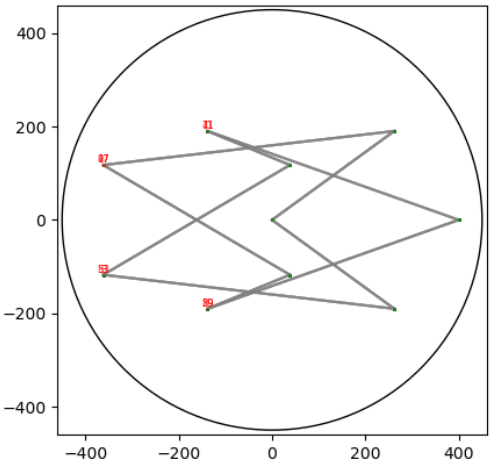
n = 94 (2p); E = {2,47}



n = 96; E = {2,2,2,2,2,3}



n = 98; E = {2,7,7}



n = 100; E = {2,2,5,5}

Délimiter (Denise Vella-Chemla, 12.9.2020)

On a compris un certain nombre de choses autour de la conjecture de Goldbach [1]. Cependant on n'a pas réussi à démontrer qu'il existe toujours un décomposant car on n'a pas pu "délimiter la recherche".

Dernièrement on a choisi de représenter les nombres par des points sur un disque "normalisé".

On se focalise sur le nombre pair n dont on cherche des décomposants de Goldbach. n a une factorisation en produit de nombres premiers de la forme

$$F(n) = \prod_{p_k \leq \sqrt{n}} 2^{\alpha_1} 3^{\alpha_2} \dots p_k^{\alpha_k}.$$

On considère alors E $= \{p'_1, p'_2, \dots\}$ l'ensemble des nombres premiers qui apparaissent (ont une puissance non nulle) dans la décomposition $F(n)$. On représente géométriquement pour chaque nombre entier x inférieur à une certaine valeur, sa somme de complexes associée :

$$g(x) = \sum_{k=1}^{k=|E|} e^{\frac{2i\pi(x \bmod p'_k)}{p'_k}}$$

Il aurait semblé, au premier abord, qu'il y ait toujours un décomposant de Goldbach de n que l'on appellera p, et qui ait une image $g(p)$ qui a son cosinus qui est minimal, i.e. qui se retrouve le plus à gauche qu'il est possible sur les visualisations ci-dessous, pour les nombres de 12 à 100. Du fait de la périodicité des restes, si cela s'était avéré toujours vrai, du moins sur les cas étudiés, cela nous aurait permis de nous focaliser sur quelques suites arithmétiques de nombres seulement, et peut-être de pouvoir démontrer l'impossibilité de l'absence d'un décomposant de Goldbach sur les seules suites $ax + b$ en question.

Cependant, cette hypothèse n'est pas valide pour les nombres 44, 52, 68 et 92. Bizarrement ce sont tous des $4p$ ($4 \times 11, 4 \times 13$, 4×17 et 4×23). En effet, les suites arithmétiques les "plus à gauche" pour 44 sont les $11x + 5$ et $11x + 6$ alors que ses décomposants sont 3, 7 et 13, d'aucune de ces 3 formes. Pour 52, ce sont les $13x + 6$ et les $13x + 7$ qui sont les plus à gauche alors que ses décomposants 5, 11 et 23 ne sont pas de ces formes et pour 68, ce sont les $17x + 8$ et les $17x + 9$ qui sont à l'extrême-gauche et 7 et 31, les décomposants de Goldbach de 68 ne sont pas de ces formes.

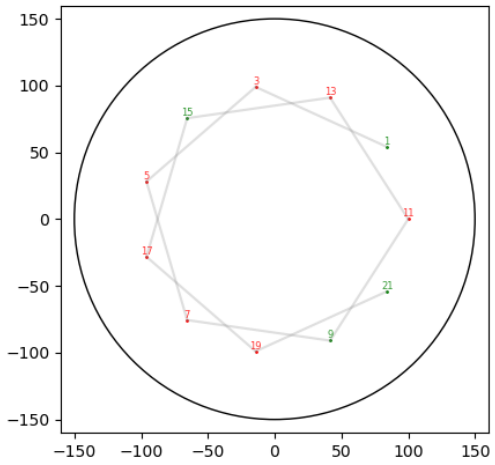
Concernant 54, même si aucun de ses décomposants n'apparaît parce qu'on n'a pas écrit les nombres "assez loin", l'un de ses décomposants (par exemple 11) est bien de la forme $(3x + 2)$ des nombres dont l'image est la plus à gauche qu'il est possible.

Remarque 1 : pour alléger les représentations, on ne visualise que les images des nombres impairs. Comme vu dans une note précédente [2], le fait de considérer également les nombres pairs ne fait qu'alourdir les dessins qui deviennent des "écheveaux" dans lesquels on passe systématiquement de gauche (les impairs) à droite (les pairs), ceci découlant de l'addition systématique de 1 (pour les pairs, correspondant à $e^{2i\pi(x \bmod 2)/2} = 1$) ou de -1 (pour les impairs, correspondant au $e^{2i\pi(x \bmod 2)/2} = e^{i\pi} = -1$).

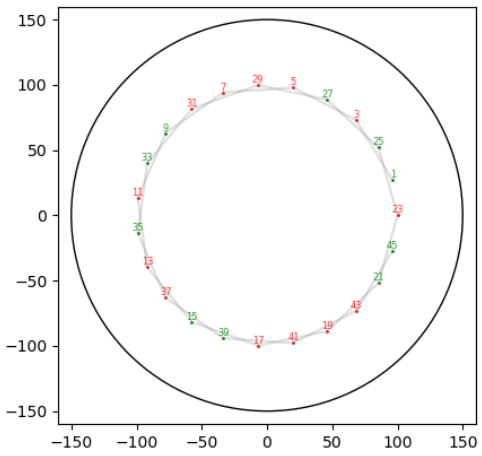
Remarque 2 : on oublie les doubles de nombres premiers qui vérifient trivialement la conjecture, on ne montre ci-dessous que les visualisations pour $n = 22$ (resp. $n = 46$) qui est le double du nombre premier 11 (resp. le double du nombre premier 23, ce sont les deux premières visualisations) et toutes les visualisations pour des doubles de nombres premiers seront du même style, une sorte de chaînette doublée entrelacée (les quadruples de nombres premiers ou les $2^k p$, comme 44, 52, 56 semblent avoir des visualisations associées similaires). Pour les $2p$ d'ailleurs, mais non pas pour les puissances de 2 supérieures strictement à 2 (les $4p$, les $8p$, etc., produits d'une puissance de 2 par un nombre premier simple), le décomposant trivial est tout à droite du dessin.

Remarque 3 : toujours pour des raisons de lisibilité, on n'a pas refermé les boucles et on a fait apparaître seulement les valeurs des nombres du "premier tour", i.e. les nombres positifs inférieurs à $2 \times \prod_{k \in E} k$ (dans la note précédente, l'écriture de certains nombres ayant lieu au même endroit, on ne parvenait plus à les lire).

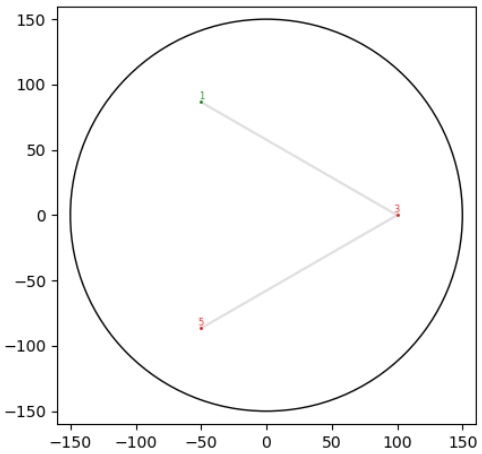
1. cf. http://denisevellachemla.eu/demo-caracterisation-DG.pdf.
2. http://denisevellachemla.eu/cgdv-sommes-de-complexes.pdf.



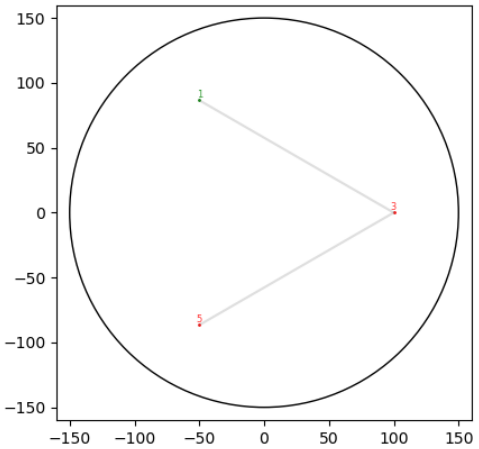
n = 22 (2p) ; E = {11}



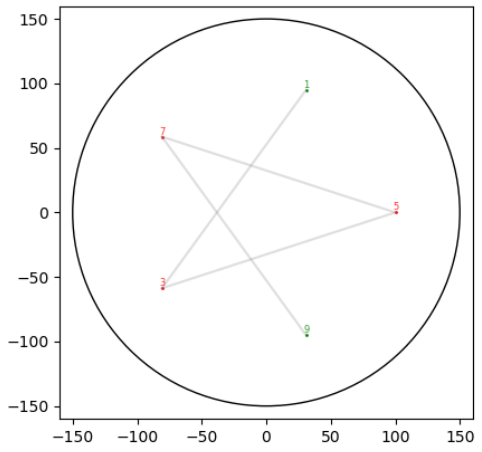
n = 46 (2p) ; E = {23}



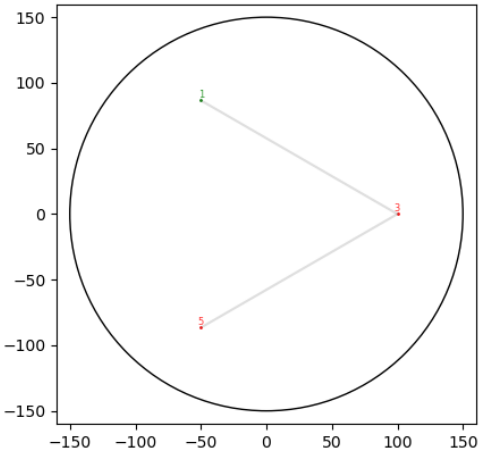
n = 12 ; E = {3}



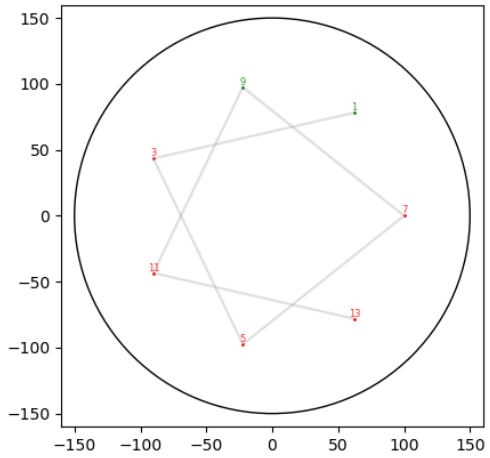
n = 18 ; E = {3}



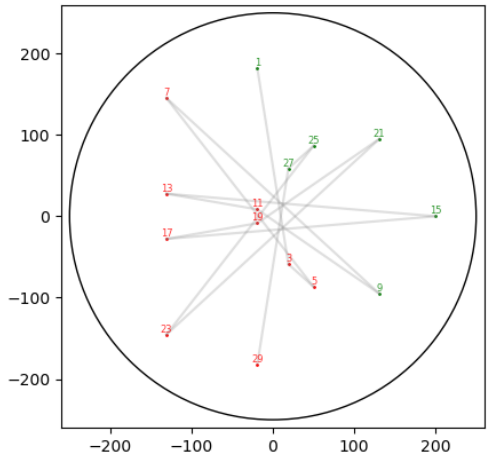
n = 20 ; E = {5}



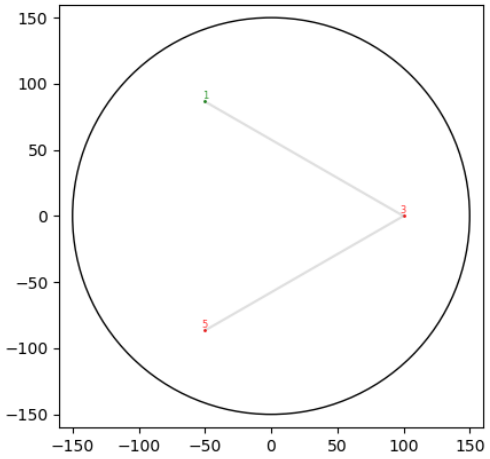
n = 24 ; E = {3}



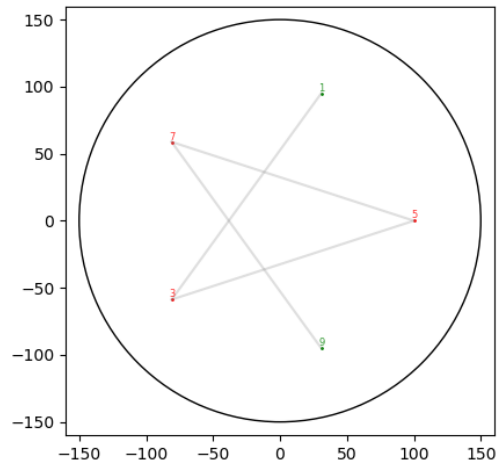
n = 28 ; E = {7}



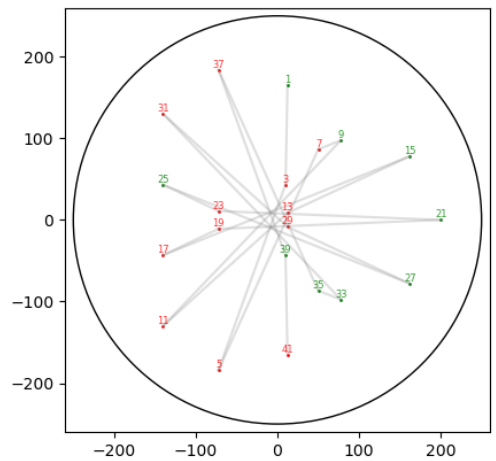
n = 30 ; E = {3,5}



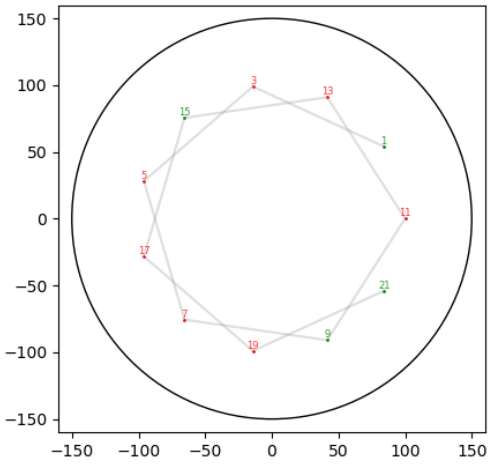
n = 36 ; E = {3}



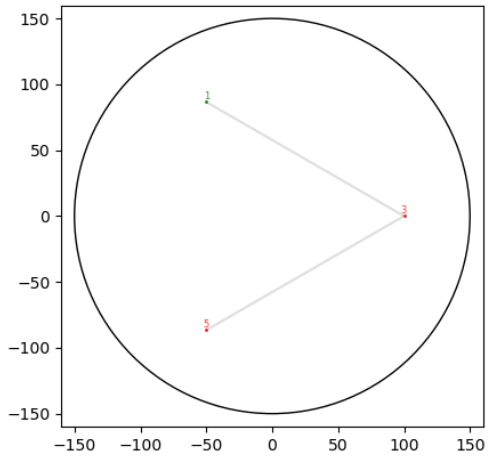
n = 40 ; E = {5}



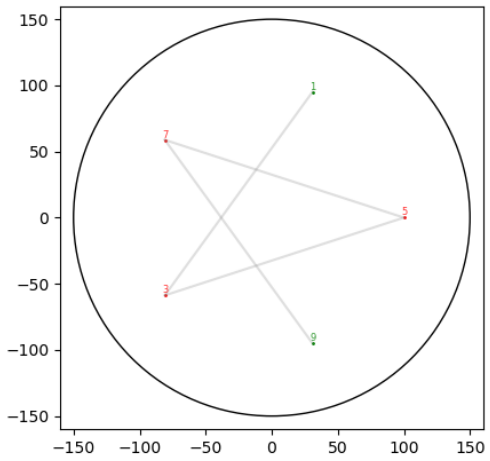
n = 42 ; E = {3,7}



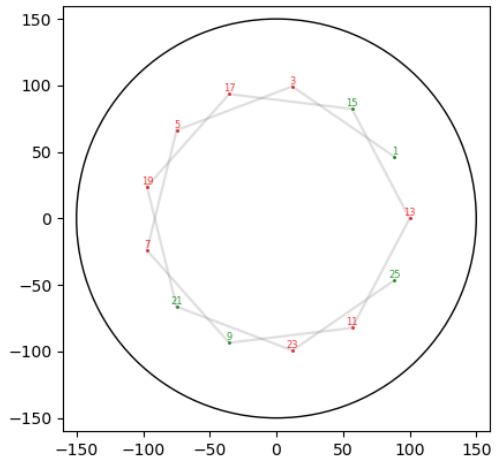
n = 44 ; E = {11}



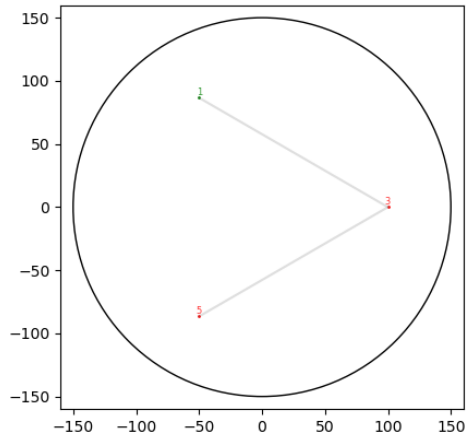
n = 48 ; E = {3}



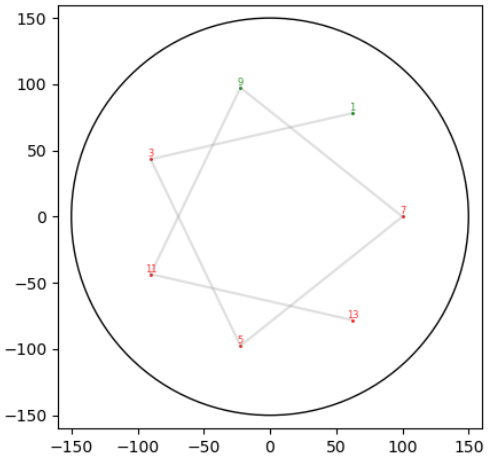
n = 50 ; E = {5}



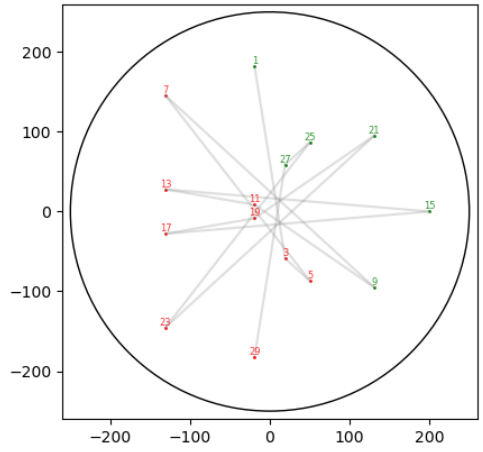
n = 52 ; E = {13}



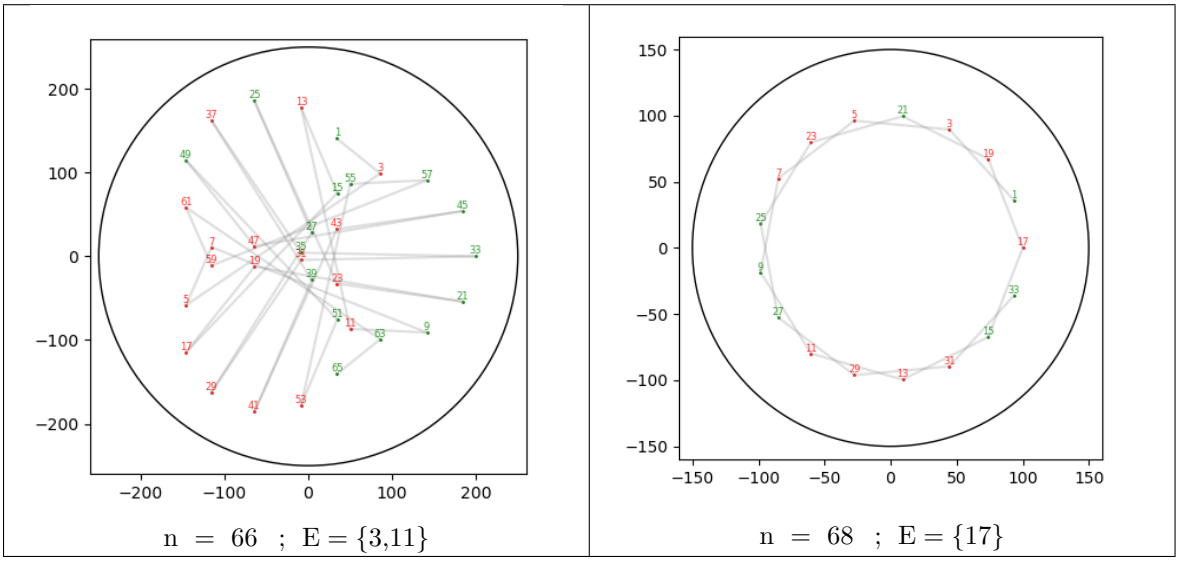
n = 54 ; E = {3}



n = 56 ; E = {7}

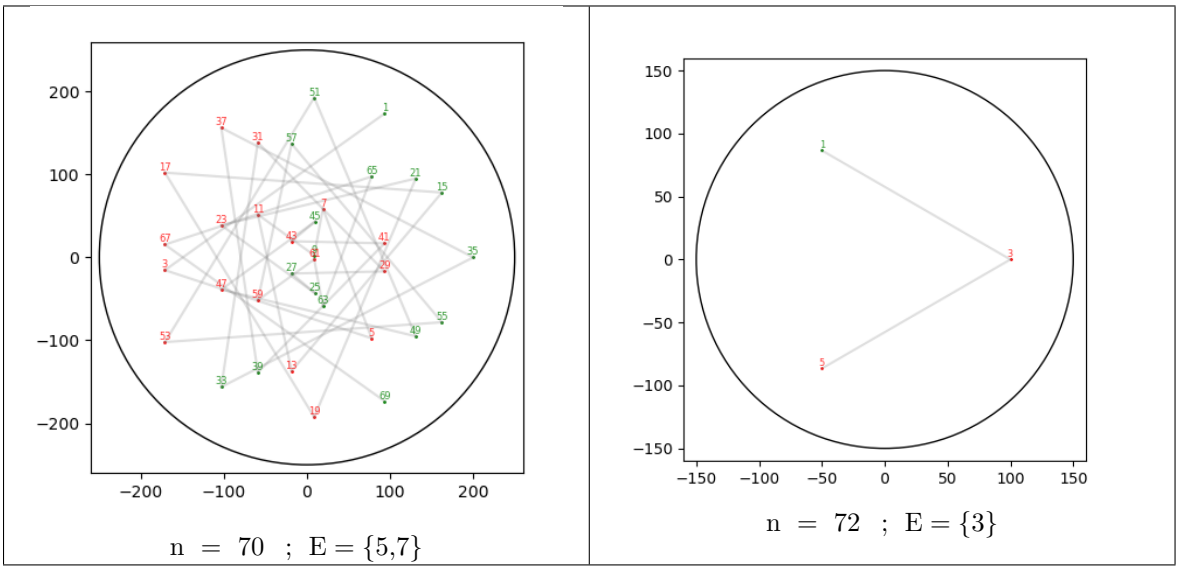


n = 60 ; E = {3,5}



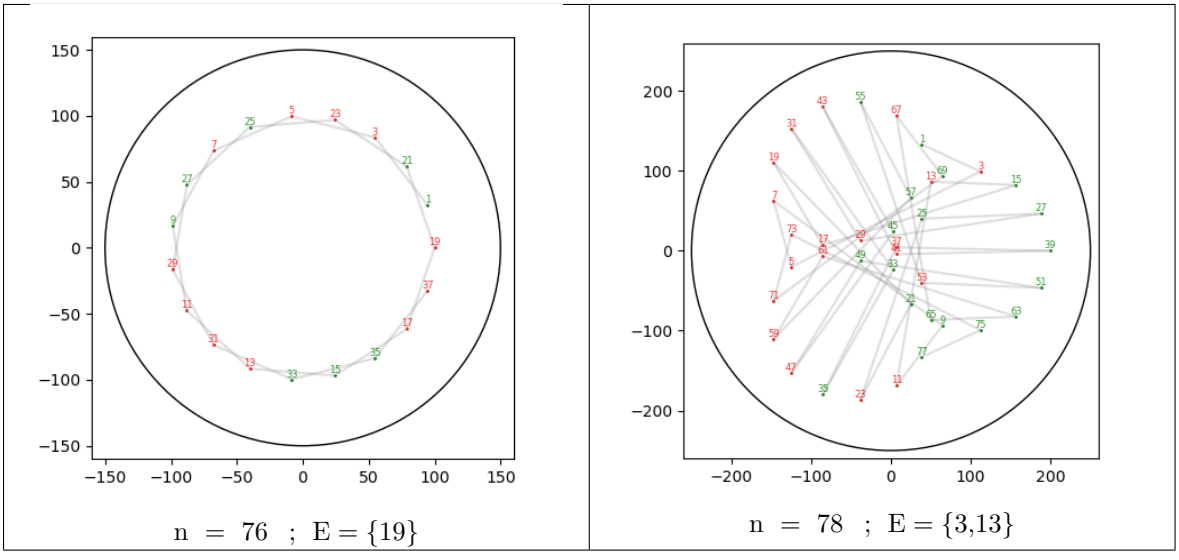
n = 66 ; E = {3,11}

n = 68 ; E = {17}



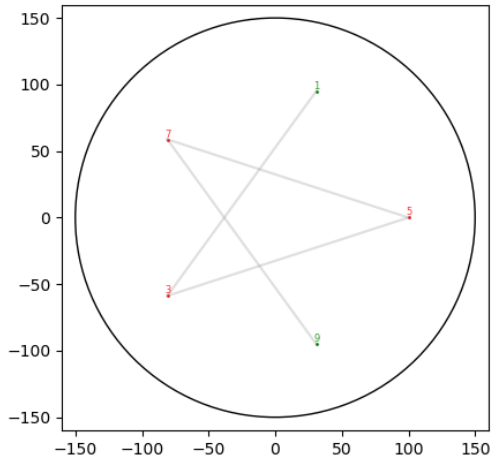
n = 70 ; E = {5,7}

n = 72 ; E = {3}

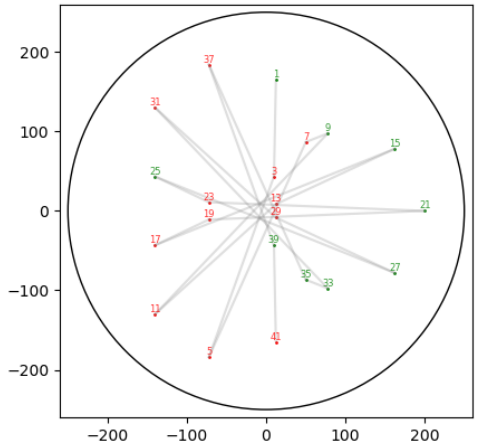


n = 76 ; E = {19}

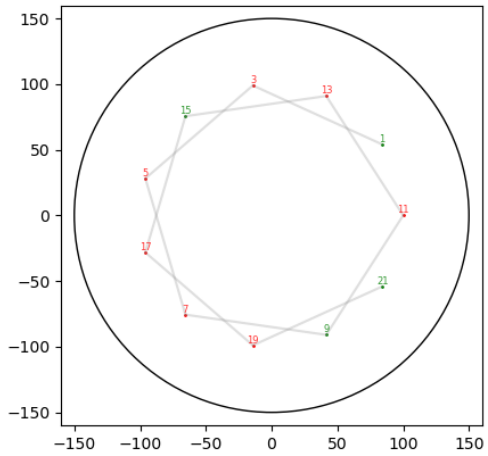
n = 78 ; E = {3,13}



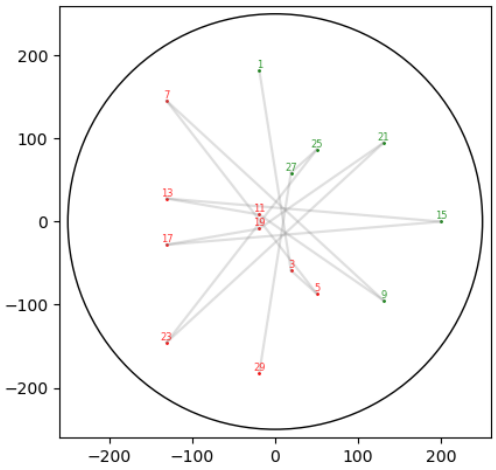
n = 80 ; E = {5}



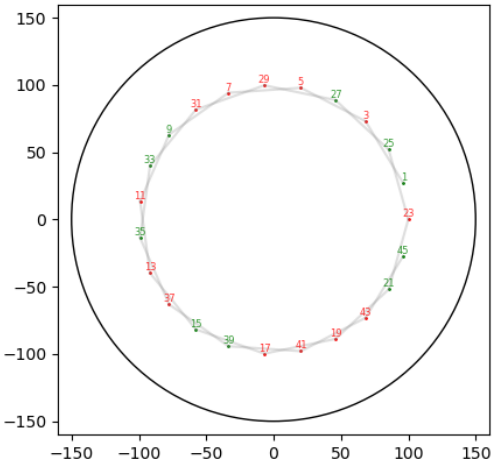
n = 84 ; E = {3,7}



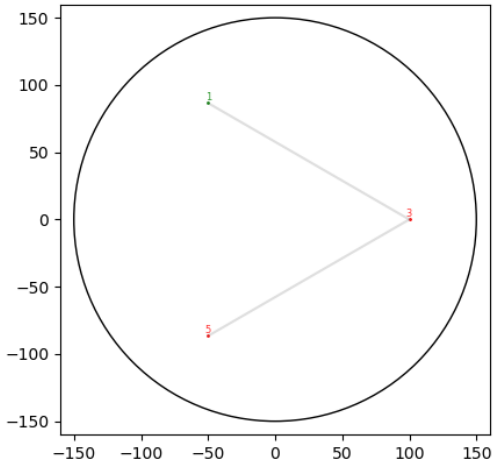
n = 88 ; E = {11}



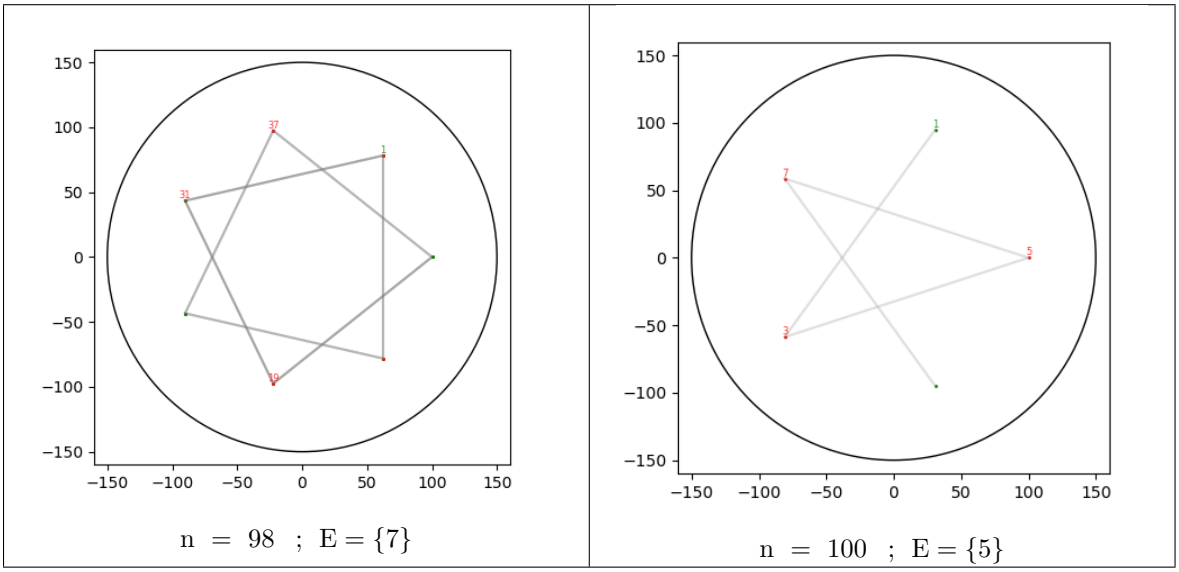
n = 90 ; E = {3,5}



n = 92 ; E = {23}



n = 96 ; E = {3}



Pour les nombres posant problème de la forme $4p$ que sont 44, 52 et 68, on "réussit" cependant à placer l'un de leur décomposant à l'extrême gauche du graphique en prenant $E = \{5\}$. On obtient alors comme visualisation la même étoile à 5 branches déjà obtenue pour 20, 40, 50, 80 et 100. Il faudrait pouvoir démontrer que pour les nombres de forme $4p$, les suites arithmétiques $10x + 3$ ou $10x + 7$ permettraient systématiquement de trouver un décomposant, en poussant plus loin les expérimentations dans un premier temps.

Si on essaye de généraliser[3] les résultats pour les nombres dont la factorisation contient seulement deux nombres premiers (2 et un autre nombre premier p), on a remarqué sur les cas étudiés que :

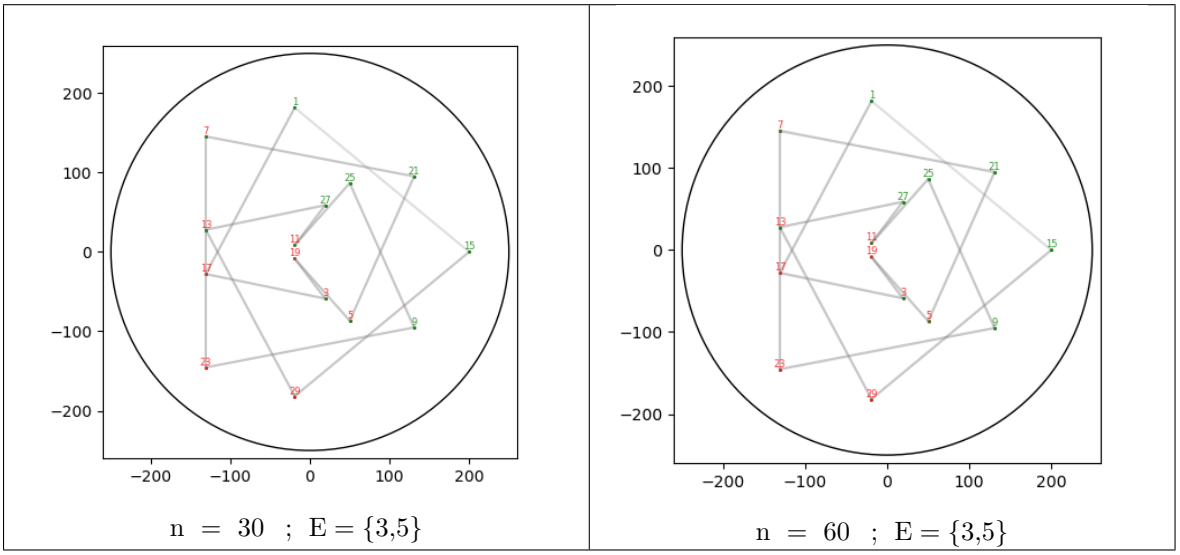
- les nombres factorisables comme $2^a 3^b$ ont 5 ou bien un autre nombre $3x + 2$ parmi leurs décomposants de Goldbach (12, 18, 24, 36, 48, 54, 72, 96, dessin ressemblant au signe $>$) ;
- les nombres factorisables comme $2^a 5^b$ ont 3 et/ou 7 ou bien un autre $5x + 2$ et/ou $5x + 3$ parmi leurs décomposants de Goldbach (20, 40, 50, 80, 100, étoile de Noël) ;
- les nombres de la forme $2^a 7^b$ ont 3 et/ou 11 ou bien un autre $7x+3$ et/ou $7x+4$ parmi leurs décomposants de Goldbach (28, 56, 98) ;
- les nombres de la forme $2^a 11^b$ ont 5 et/ou 17 ou bien un autre $11x + 5$ et/ou $11x + 6$ parmi leurs décomposants de Goldbach ;
- les nombres de la forme $2^a 13^b$ ont 7 et/ou 19 ou bien un autre $13x + 6$ et/ou $13x + 7$ parmi leurs décomposants de Goldbach ;
- les nombres de la forme $2^a 19^b$ ont 29 ou bien un autre $19x+9$ et/ou $19x+10$ parmi leurs décomposants de Goldbach ;
- les nombres de la forme $2^a 23^b$ ont 11 ou bien un autre $23x+11$ et/ou $23x+12$ parmi leurs décomposants de Goldbach ;

ce qu'on aimerait pouvoir résumer par :

- les nombres de la forme $2^a p^b$ ont un $px + \dfrac{p \pm 1}{2}$ parmi leurs décomposants de Goldbach.

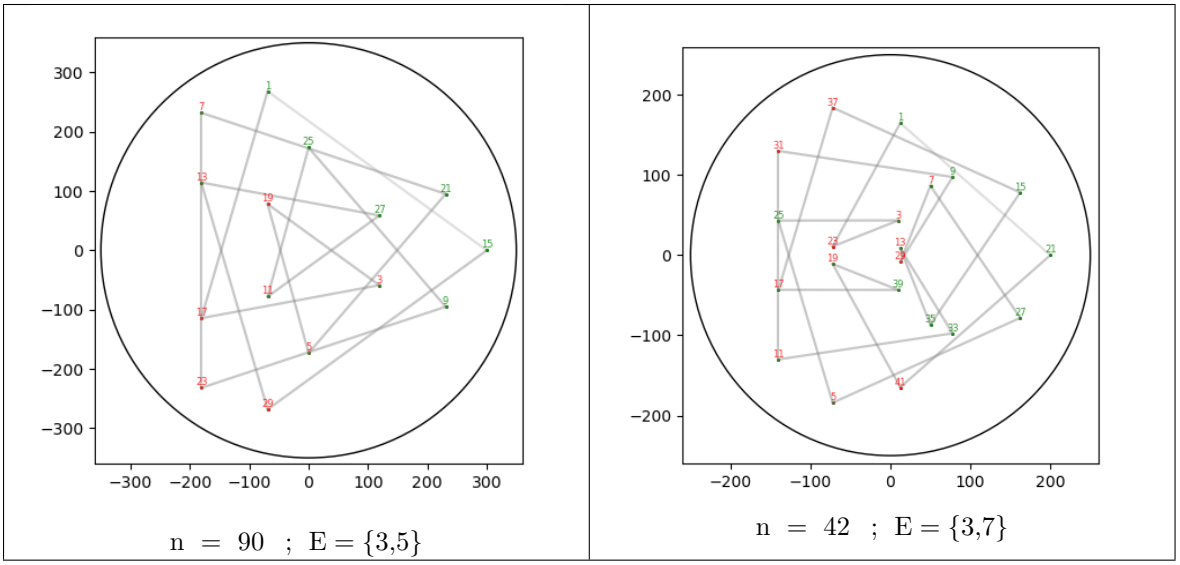
Pour mémoire, le fait d'avoir éliminé les pairs des représentations fait un peu perdre de vue les structures, même si cela allège les visualisations. On reprend ci-dessous les visualisations complètes, pour les nombres 30, 42, 60, 66, 78, 90 qui ont au moins 3 nombres premiers différents dans leur factorisation ainsi que 210, 2310, des doubles de primorielles ou 770. Il faut garder à l'esprit que c'est parfois au "second tour" que les nombres impairs "plus grands", i.e. ceux des moitiés hautes des intervalles, sont "attrapés".

3. Ce qui ne prouve rien.



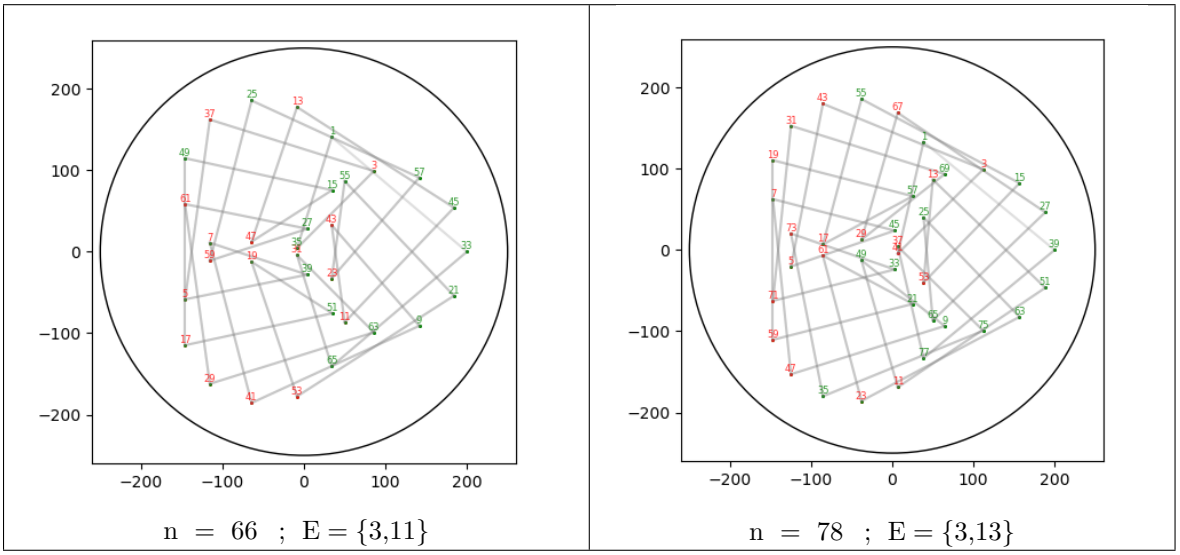
n = 30 ; E = {3,5}

n = 60 ; E = {3,5}



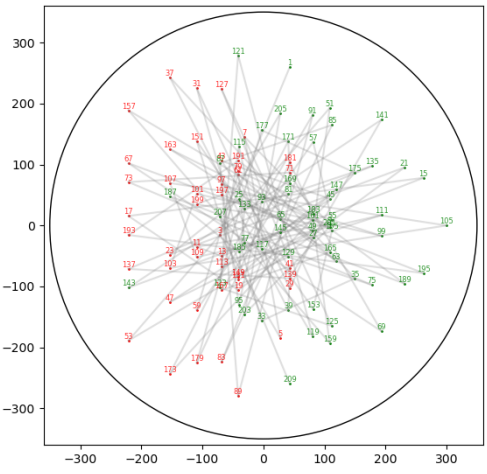
n = 90 ; E = {3,5}

n = 42 ; E = {3,7}

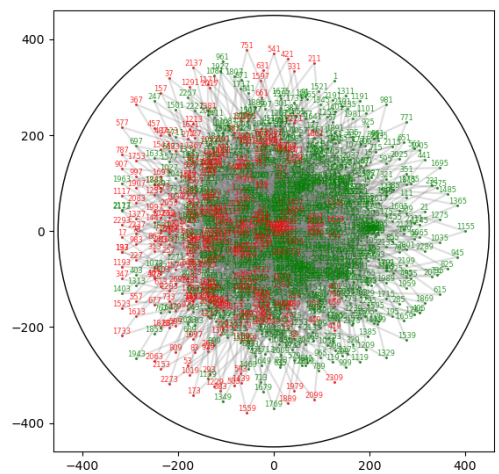


n = 66 ; E = {3,11}

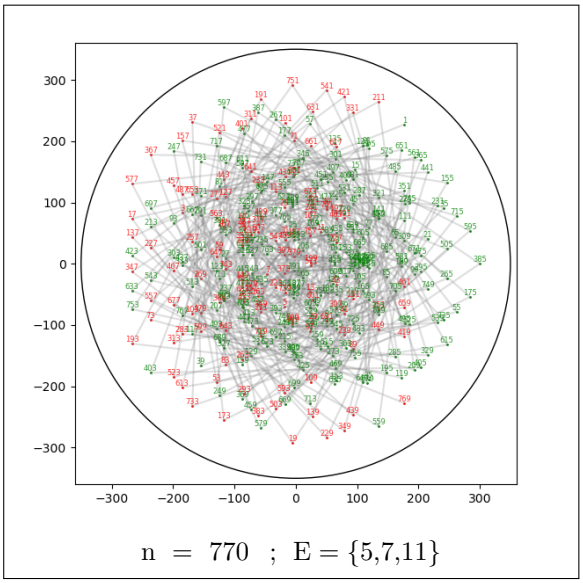
n = 78 ; E = {3,13}



n = 210 ; E = {3,5,7}



n = 2310 ; E = {3,5,7,11}



n = 770 ; E = {5,7,11}

Nouveaux résultats expérimentaux concernant la conjecture de Goldbach

J-M. Deshouillers
Mathématiques Stochastiques, Université Victor Segalen Bordeaux 2
F-33076 Bordeaux Cedex, France
J-M.Deshouillers@u-bordeaux2.fr

H.J.J. te Riele
CWI, P.O. Box 94079, 1090 GB Amsterdam, Pays-Bas
Herman.te.Riele@cwi.nl

Y. Saouter
Institut de Recherche en Informatique de Toulouse
118 route de Narbonne, F-31062 Toulouse Cedex, France
Yannick.Saouter@irit.fr

31 mars 1998

Résumé La conjecture de Goldbach énonce que tout entier pair ≥ 4 peut être écrit comme une somme de deux nombres premiers. On sait qu'elle est vraie jusqu'à 4×10^{11}. Dans cet article, de nouvelles expérimentations menées sur un super-ordinateur Cray C916 et sur un serveur de calcul SGI à 18 R8000 CPU sont décrites, qui étendent cette limite à 10^{14}. Deux conséquences sont que (1) sous l'hypothèse de Riemann généralisée, tout nombre impair ≥ 7 peut s'écrire comme une somme de trois nombres premiers, et (2) en supposant vraie l'hypothèse de Riemann, tout entier positif pair peut s'écrire comme une somme d'au plus quatre nombres premiers. De plus, nous avons vérifié la conjecture de Goldbach pour tous les nombres pairs dans les intervalles $[10^{5i}, 10^{5i} + 10^8]$, pour $i = 3, 4, \ldots, 20$ et $[10^{10i}, 10^{10i} + 10^9]$, pour $i = 20, 21, \ldots, 30$.
Un modèle heuristique est fourni qui prédit le nombre moyen d'étapes nécessaires pour vérifier la conjecture de Goldbach sur un intervalle donné. Nos résultats expérimentaux sont en bon accord avec cette prédiction. Cela ajoute à l'évidence de la vérité de la conjecture de Goldbach.

Remerciements. Le premier auteur a bénéficié du support du CNRS et des universités Bordeaux 1 et Bordeaux 2. Le second auteur remercie Walter Lioen pour son aide à prouver la primalité de nombreux grands nombres avec les programmes de Cohen, Lenstra et Winter, et de Bosma et Van der Hulst. L'accès à l'ordinateur de calcul vectoriel Cray C916 au Centre de Calcul académique d'Amsterdam (SARA) a été possible grâce à la Fondation nationale allemande d'aide au calcul (NCF). L'accès au serveur de calcul Power Challenge Array R10000 a été possible grâce au Centre Charles Hermite de Nancy, et à l'INRIA de Lorraine.

Modélisation, analyse et simulation (MAS), Rapport MAS-R9804, 31 mars 1998.

1991 Classification des sujets mathématiques : Primaire 11P32. Secondaire 11Y99.
1991 Système de classification des résultats de calculs : F.2.1.
Mots et phrases clefs : conjecture de Goldbach, somme de nombres premiers, test de primalité, calcul vectoriel, Cray C916, cluster de stations de travail.
Note : Cet article apparaîtra dans les Proceedings de ANTS-III (Symposium de théorie des nombres algorithmique III, Reed College, Portland, Oregon, USA, 21-25 juin 1998).
La contribution du second auteur a été financée par le projet MAS2.5 "Théorie des nombres computationnelle et sécurité des données".

1. Introduction

La conjecture *binaire* de Goldbach (BGC) énonce que tout nombre entier pair ≥ 4 peut être exprimé comme une somme de deux nombres premiers. Par G_2 nous dénotons la plus petite borne supérieure pour le nombre G avec la propriété que tous les nombres pairs n tels que $4 \leq n \leq G$ peuvent s'écrire comme somme de deux nombres premiers. On sait que $G_2 \geq 4 \times 10^{11}$ [15, 17, 7, 16].
La conjecture de Goldbach *ternaire* (TGC) énonce que tout nombre entier impair ≥ 7 peut être exprimé comme la somme de trois nombres premiers. De façon évidente, la vérité de BGC implique la vérité de TGC.
En 1923, Hardy et Littlewood [8] ont prouvé que, sous l'hypothèse d'une faible version de l'hypothèse de Riemann généralisée (GRH), il existe un entier positif M_0 tel que TGC est vérifiée par tous les nombres entiers impairs $\geq M_0$. En 1937, Vinogradov [18] a prouvé, inconditionnellement, qu'il existe un entier positif N_0 tel que TGC est vérifiée pour tous les nombres entiers impairs $\geq N_0$.
En 1989, Chen et Wang [3] ont montré qu'on peut prendre $N_0 = 10^{43000}$, et en 1993 [4] ils ont montré, en supposant GRH, que l'on peut prendre $M_0 = 10^{50}$. Très récemment, Zinoviev [19] a prouvé, en supposant GRH, qu'on peut prendre $M_0 = 10^{20}$. En utilisant des calculs classiques, selon Schoenfeld [14], ce résultat implique [6] le

Théorème A *Si GRH est vraie et si $G_2 \geq 1.615 \times 10^{12}$, alors tout nombre entier impair ≥ 7 peut être exprimé comme la somme de trois nombres premiers.*

C'est une de nos motivations pour la présente étude.

Remarque Dans [13], le troisième auteur a prouvé, inconditionnellement, la vérité de TGC jusqu'à 10^{20} en calculant une séquence croissante d'environ 2.5×10^8 nombres premiers $q_0, q_1, \ldots\ldots, q_Q$ telle que $q_0 < 4 \times 10^{11}$, $q_{i+1} - q_i < 4 \times 10^{11}$ pour tout $0 \leq i \leq Q-1$ et $q_Q > 10^{20}$. Cela montre que près de tout nombre impair $N < 10^{20}$, il y a un nombre premier q tel que $N - q < 4 \times 10^{11}$ et par [16] $N - q$ peut être exprimé comme une somme de deux nombres premiers.

Une seconde motivation était le résultat suivant de Kaniecki [10] :

Théorème B *Si l'hypothèse de Riemann (RH) est vraie et si $G_2 \geq 1.405 \times 10^{12}$, alors tout nombre entier positif pair peut être exprimé comme la somme d'au plus quatre nombres premiers.*

Sans aucune hypothèse, Ramaré [12] a prouvé que tout nombre entier pair positif est la somme d'au plus six nombres premiers.
Dans ce papier, nous rapportons les résultats d'expériences extensives dont l'effet est le
Théorème 1 *On a $G_2 \geq 10^{14}$,*
de telle façon que les suppositions sur G_2 dans les Théorèmes A et B sont satisfaites.
De plus, nous avons vérifié que tous les nombres pairs dans quelques intervalles donnés sont sommes de deux nombres premiers, notamment :

Théorème 2 *Tous les nombres entiers pairs dans les intervalles* $[10^{5i}; 10^{5i} + 10^8]$*, pour* $i = 3, 4, \ldots, 20$ *et* $[10^{10i}; 10^{10i} + 10^9]$*, pour* $i = 20, 21, \ldots, 30$ *sont sommes de deux nombres premiers.*

Nous avons vérifié BGC avec un algorithme qui a été utilisé, mais non fourni explicitement par Mok-Kong Shen [15]. De plus pour étendre l'intervalle sur lequel BGC est connue comme étant vraie d'un facteur de 250, nous donnons un modèle heuristique qui prédit le nombre moyen d'étapes nécessaires pour vérifier BGC avec cet algorithme. Cela ajoute une évidence théorique à l'évidence numérique déjà accablante de la vérité de BGC.

2. Deux algorithmes pour vérifier la conjecture de Goldbach binaire sur $[a, b]$

Les algorithmes connus pour vérifier la conjecture de Goldbach sur un intervalle donné $[a, b]$ consistent à trouver deux ensembles de nombres premiers $\mathcal{P}$ et $\mathcal{Q}$ tels que $\mathcal{P} + Q$ couvre tous les nombres pairs dans $[a, b]$.

Appelons p_i le i-ème nombre premier impair. Une approche, telle qu'appliquée en [17, 7, 16], consiste à trouver, pour tout nombre pair $e \in [a, b]$, le plus petit nombre premier impair p_i tel que $e - p_i$ est un nombre premier. Cela revient à prendre pour $\mathcal{P}$ les nombres premiers impairs $p_1, p_2, \ldots, p_m$ pour m adéquat et à prendre

$$\mathcal{Q} = \mathcal{Q}(a, b) = \{q \mid q \text{ premier et } a - \epsilon_a \leq q \leq b\}$$

pour un ϵ_a convenablement choisi. Une série d'ensembles de nombres pairs $\mathcal{E}_0 \subset \mathcal{E}_1 \subseteq \mathcal{E}_2 \subseteq \ldots$ est alors générée, définie par $\mathcal{E}_0 = \varnothing$,

$$\mathcal{E}_{i+1} = \mathcal{E}_i \cup (\mathcal{Q}(a, b) + p_{i+1}), \qquad i = 0, 1, \ldots, \text{ [1]}$$

jusqu'à ce que pour un certain j l'ensemble $\mathcal{E}_j$ couvre *tous* les nombres pairs dans l'intervalle $[a, b]$. L'ensemble $\mathcal{Q}(a, b)$ est engendré avec le crible d'Eratosthène : c'est la partie du calcul la plus chronophage. Pour le choix de ϵ_a, il est suffisant que ϵ_a soit plus grand que le plus grand nombre premier impair p_j utilisé pour engendrer les ensembles $\mathcal{E}_j$. Cette approche permet de fournir, pour tout nombre entier pair $e \in [a, b]$, le plus petit nombre premier p tel que $e - p$ est premier (la paire $(p, e - p)$ est alors appelée la décomposition de Goldbach minimale de e). Dans les calculs utilisés pour vérifier la conjecture de Goldbach jusqu'à 4×10^{11} [16], le plus grand *petit* nombre premier impair nécessaire a été $p_{446} = 3163$ (c'est le plus petit nombre premier p pour lequel $244\,885\,595\,672 - p$ est un nombre premier). Une partie coûteuse de cette approche est que quasiment tous les nombres premiers de l'intervalle $[a, b]$ doivent être déterminés.

Une approche plus efficace, comme celle appliquée dans [15], est de trouver, pour tout nombre pair $e \in [a, b]$, un nombre premier q, proche de a, pour lequel $e - q$ est un nombre premier. Cela revient à choisir pour $\mathcal{P}$ l'ensemble de tous les nombres premiers impairs environ jusqu'à $b - a$ et pour $\mathcal{Q}$ les k plus grands nombres premiers $q_1 < q_2 \ldots < q_k$ en-dessous de a, pour k adéquat. Pour la vérification effective de l'intervalle $[a, b]$, on génère l'ensemble des nombres pairs $\mathcal{F}_0 \subset \mathcal{F}_1 \subseteq \mathcal{F}_2 \subseteq \ldots$, défini par $\mathcal{F}_0 = \varnothing$,

$$\mathcal{F}_{i+1} = \mathcal{F}_i \cup (\mathcal{P} + q_{i+1}), \qquad i = 0, 1, \ldots,$$

jusqu'à ce que pour un certain j l'ensemble F_j couvre *tous* les nombres pairs dans l'intervalle $[a, b]$. Le grand ensemble $\mathcal{P}$ est engendré avec le crible d'Eratosthène, mais le travail n'a à être fait qu'une seule fois si nous fixons la longueur $b - a$ des intervalles $[a, b]$. Les nombres premiers dans $\mathcal{Q}$ dépendent de a et pourraient aussi

1. Par $\mathcal{Q}(a, b) + p_{i+1}$, nous voulons dire, comme d'habitude, l'ensemble $\{q + p_{i+1} | q \in \mathcal{Q}(a, b)\}$.

être générés par le crible d'Eratosthène. Pourtant, puisque nous n'avons besoin que de quelques centaines de tels nombres premiers et puisqu'ils n'excèdent pas 10^{14}, il est plus économique d'utiliser les résultats de Jaeschke [9] par lesquels pour tout nombre premier nous avons seulement besoin de faire quelques tests de pseudo-primalité, aussi longtemps qu'ils n'excèdent pas 3.4×10^{14}. Un inconvénient de cette approche est qu'en général, elle *ne trouve pas*, la décomposition de Goldbach *minimale*.
Dans cette étude, nous avons choisi d'implémenter la seconde approche. En plus d'étendre G_2 autant qu'il est possible, nous sommes intéressés par le *nombre d'étapes nécessaires* dans les algorithmes ci-dessus, pour vérifier BGC. Dans la prochaine section, nous discutons d'un modèle heuristique qui est capable de prédire le nombre d'étapes *moyen* précisément.

3. Prédire le nombre moyen d'étapes nécessaires pour vérifier BGC sur $[a, b]$

Nous présentons quelques heuristiques pour estimer le nombre moyen d'étapes nécessaires pour générer les ensembles $\mathcal{F}_i, i = 0, 1, \ldots$ jusqu'à ce que tous les nombres pairs dans $[a, b]$ soient couverts.
Prenons $l = b - a$ assez grand, comparé avec a, de telle façon que nous puissions trouver suffisamment de nombres premiers q dans le voisinage de a pour répondre à notre objectif. Le nombre de nombres premiers dans $\mathcal{P}$ est approximativement $\pi(l)$. Pour chaque nombre premier $q \in \mathcal{Q}$, l'ensemble $\mathcal{P}+q$ couvre environ $\pi(l)$ éléments dans $[a, b]$, i.e. une proportion d'environ $1-2\pi(l)/l$ des nombres pairs dans $[a, b]$ n'est pas couverte. Si l'on suppose, ce qui n'est pas le cas, une indépendance statistique entre le fait d'être couvert par $\mathcal{P} + q$ et le fait d'être couvert par $\mathcal{P} + q'$ ainsi qu'une hypothèse supplémentaire d'uniformité, on peut s'attendre à ce que, en moyenne, tous les nombres pairs soient couverts à l'aide de k éléments q quand $(1 - 2\pi(l)/l)^k$ est grosso-modo égal à $2/l$, l'inverse du nombre de nombres pairs dans $[a, b]$. Si $l = 10^8$, cela amène à $k \approx 145$ et pour $l = 10^9$ cela entraîne $k \approx 187$. Une étude plus détaillée du modèle probabiliste amène à un comportement de Poisson pour le nombre d'entiers qui ne sont pas couverts ; dans ce modèle, pour $k \approx 148$ dans le cas où $l = 10^8$ (et $k \approx 191$ quand $l = 10^9$) la probabilité de couvrir l'intégralité de l'intervalle $[a, b]$ est proche de $1/2$. Pourtant, cela n'est pas en accord avec nos observations expérimentales décrites dans les sections suivantes. Bien qu'une sorte de quasi-indépendance statistique semble une hypothèse naturelle, la distribution uniforme des nombres premiers n'est définitivement pas une hypothèse décente.
Un premier défaut d'uniformité vient de la raréfaction des nombres premiers (la densité locale des nombres premiers autour de x décroît lorsque x croît). Considérer seulement les grands nombres premiers, par exemple, ceux entre 10^7 et 10^8 pour couvrir un intervalle de longueur 9×10^7, amène à la valeur $k \approx 150$; ceci est en accord avec la valeur moyenne expérimentale des k observés (cf. Section 5.1).
Un second et plus important manque d'uniformité est de nature arithmétique. Choisissons un petit nombre premier r et considérons les décompositions de Goldbach de tous les nombres pairs qui sont premiers à $R = 3.5 \ldots r$. Pour chaque grand nombre q (premier, et donc premier à R), tous les nombres premiers $p \in P$ qui vérifient $(p + q, R) > 1$ ne peuvent être utilisés pour décomposer nos nombres. Le nombre de classes admissibles de premiers est ainsi $(3-2)(5-2)\ldots(r-2)$ et la proportion de nombres premiers utiles dans $\mathcal{P}$ est donc $\frac{(3-2)(5-2)\ldots(r-2)}{(3-1)(5-1)\ldots(r-1)}$. Ainsi, pour chaque nombre premier q, l'ensemble $\mathcal{P} + q$ contient environ $\frac{(3-2)(5-2)\ldots(r-2)}{(3-1)(5-1)\ldots(r-1)}\pi(l)$ nombres pairs différents et alors, la proportion de nos nombres pairs dans $[a, b]$ qui sont couverts en une étape est

$$\frac{(3-2)(5-2)\ldots(r-2)}{(3-1)(5-1)\ldots(r-1)}\pi(l)/\left(\frac{(3-1)(5-1)\ldots(r-1)}{3.5\ldots r}\frac{l}{2}\right)$$

i.e.,

$$2 \prod_{\substack{3 \le s \le r \\ s \text{ premier}}} \left(1 - \frac{1}{(s-1)^2}\right) \frac{\pi(l)}{l} = C(r) \frac{\pi(l)}{l}.$$

Par le même raisonnement que ci-dessus, nous nous attendons à ce que k soit proche de la solution de $\left(1 - C(r)\frac{\pi(l)}{l}\right)^k = \frac{2R}{\phi(R)l}$. Pour $r = 97$ et $l = 10^8$, cela amène à $k \approx 206$ et pour $l = 10^9$ nous trouvons $k \approx 270$.

Cela s'accorde bien aux résultats de nos expérimentations et cela implique, comme on peut s'y attendre, que pour les nombres pairs dans $[a, b]$ qui ne sont pas premiers à $R = 3.5 \dots r$, il est en général plus facile de trouver une décomposition de Goldbach que pour ceux qui sont premiers à R. À nouveau, si nous améliorons ce modèle par des considérations probabilistes de Poisson, et en considérant la raréfaction des nombres premiers, nous sommes amenés à $k \approx 214$ quand $l = 10^8$, qui est, ici aussi, en bon accord avec les données expérimentales de la Section 5.1. Ce raisonnement probabiliste sera développé dans un article à venir.

4. Calculs qui étendent G_2 de 4×10^{11} à 10^{14}

Nous avons adopté l'approche de Shen, décrite dans la Section 2, pour vérifier la conjecture de Goldbach aussi loin que possible au-delà de la borne connue de 4×10^{11}.

Les intervalles $[a, b]$ ont été choisis de façon à être de longueur 10^8 ou 128×10^6 ou 10^9. Le plus grand nombre premier dont on a besoin dans l'ensemble $\mathcal{P}$ est proche de $b - q_1$. Par le théorème des nombres premiers, $q_1 \approx a - k \log a$, de telle façon que $b - q_1 \approx b - a + k \log a$. Comme valeurs *maximum* de k, nous avons trouvé dans nos expériences que $k = 430$ était une valeur suffisante. Pour $a \approx 10^{14}$, cela implique que le plus grand nombre premier dans l'ensemble $\mathcal{P}$ doive avoir une taille d'au moins $10^9 + 1.4 \times 10^4$ pour $b - a = 10^9$. Dans notre implémentation effective, nous avons choisi que $\mathcal{P}$ contienne les nombres premiers impairs jusqu'à $10^8 + 10^5$ dans le cas $b - a = 10^8$, et ceux jusqu'à $10^9 + 10^6$ dans le cas $b - a = 10^9$.

Pour la génération effective des nombres premiers proches de a, nous avons utilisé les résultats des calculs de Jaeschke [9], qui énoncent que si un entier positif $n < 215\,230\,289,8747$ [2] est un fort pseudo-premier par rapport aux cinq premiers nombres premiers $2, 3, 5, 7, 11$, alors n est premier ; les bornes correspondantes pour les six premiers et sept premiers nombres premiers sont 3 474 749 660 383 et 341 550 071 728 321, respectivement.

Initialement, à la fois le second et le troisième auteurs ont vérifié la BGC jusqu'à 10^{13}, indépendamment, sur un calculateur vectoriel Cray C916 resp. sur un serveur de calcul SGI avec 18 R8000 CPU. Après avoir appris les résultats l'un de l'autre, ils ont décidé de travailler ensemble pour atteindre la borne 10^{14}. Le second auteur a vérifié la BGC sur les intervalles $x \times 10^{13}$ pour $x = [2, 4], [6, 8], [9, 10]$ et le troisième auteur ceux pour les cas $x = [1, 2], [4, 6], [8, 9]$.

4.1 *Expérimentations sur le calculateur vectoriel Cray C916*

Nous avons implémenté l'algorithme de Shen sur un ordinateur vectoriel Cray C916 comme suit.

Au grand ensemble de nombres premiers impairs $\mathcal{P}$, nous associons un tableau d'entiers longs (long-bit) appelé ODD, dans lequel chaque bit représente un nombre impair $< 10^9 + 10^6$, le bit valant 1 si le nombre impair correspondant est premier, et 0 s'il est composé. À $\mathcal{F}_i$, nous associons un tableau de bits similaire

2. ?

appelé SIEVE, ayant la même longueur que ODD. Le premier bit de SIEVE représente le nombre impair $q_1 + 3$, le second bit $q_1 + 5$, et, et en général, le bit i représente le nombre pair $q_1 + 2i + 1$. Initialement, ODD est copié dans SIEVE, rendant le bit i du tableau SIEVE égal à 1 si $2i + 1$ est premier, indiquant que $q_1 + 2i + 1$ peut s'écrire comme une somme de deux nombres premiers q_1 et $2i + 1$. Maintenant le tableau SIEVE représente l'ensemble $\mathcal{F}_1$. Dans un second temps, le tableau SIEVE est "ou"-é avec une version décalée à droite du tableau ODD, où le décalage est égal à $(q_2 - q_1)/2$. Il est facile de voir que maintenant, le tableau SIEVE représente l'ensemble $\mathcal{F}_2 = \mathcal{F}_\infty \cup (\mathcal{P} + q_2)$. En général, $\mathcal{F}_{i+1}$ est engendré à partir de $\mathcal{F}_i$ en faisant une opération "ou" entre le tableau SIEVE et le tableau ODD, décalé à droite d'un décalage de $(q_{i+1} - q_1)/2$.

Bien sûr, ces étapes peuvent être effectuées très efficacement sur un Cray C916. Nous avons compressé 64 bits en un mot et vectorialisé les opérations "ou". La vérification pour savoir si *tous* les bits du tableau SIEVE sont devenus des 1 n'est effectuée que lorsque la chance d'occurrence de cet événement est devenue suffisamment grande (après 170 étapes, dans notre programme). Dès que le nombre de bits 0 est tombé en-dessous de 4, les nombres restant "obstinément" pairs sont listés de manière à "voir" une sortie intermédiaire.

En une passe typique, on traite 1000 intervalles consécutifs de longueur 10^9. Proche de 10^{14}, le temps pour générer 1000×430 grands nombres premiers était environ 5000 secondes CPU, et le temps de crible total était environ de 13 200 secondes. Le nombre moyen d'étapes (sur 1000 intervalles consécutifs) à chaque passe varie entre 269 et 271 avec une déviation standard entre 18 et 20. Le temps total CPU (en ayant une priorité faible) utilisé pour couvrir les intervalles $[4 \times 10^{11}, 10^{13}], [2-4] \times 10^{13}, [6-8] \times 10^{13}$, et $[9-10] \times 10^{13}$ a été approximativement de 75 heures CPU pour générer les grands nombres premiers, et 225 heures CPU pour le crible. Ce dernier temps signifie que pendant le temps utilisé pour effectuer la partie crible, une moyenne de 3.2×10^8 mots de 64 bits par seconde CPU étaient "ou"-és. Le plus grand nombre de nombres premiers que nous ayons eu à calculer a été 413 : pour $e = 33\ 836\ 446\ 494\ 106$ et le premier $q_1 = 33\ 835\ 999\ 990\ 007$, il s'avéra que $e - q_i$ est composé pour $i = 1, \ldots, 412$, et premier pour $i = 413$, ($q_{413} = 33\ 836\ 000\ 002\ 499$ et $e - q_{413} = 446\ 491\ 607$).

4.2 *Expérimentations sur un serveur de calcul SGI avec 18 R8000 CPU*

L'algorithme implémenté sur la station de travail SGI est très proche de celui du Cray C916. Les nombres premiers jusqu'à 128×10^6 sont représentés dans un tableau binaire, que nous appelons ODD, d'un million d'entrées qui sont des longs entiers 64 bits : le j-ième bit du i-ième élément du tableau est égal à 1 si et seulement si $128i + 2j + 3$ est un nombre premier. De façon similaire, un autre tableau de la même taille, correspondant au tableau SIEVE de la section précédente, est utilisé pour noter les nombres composés : le j-ième bit du i-ième élément de ce second tableau est égal à 1 si et seulement si $128i + 2j + seed$ est décomposable en une somme de deux nombres premiers, où $seed$ dénote le nombre pair auquel la phase commence.

À ce point, la tâche du programme consiste à remplir toutes les entrées de SIEVE avec le plus grand mot de 64 bits i.e. $2^{64} - 1$. Le programme cherche la dernière entrée i pour laquelle la valeur de SIEVE$[i]$ n'est pas maximum et cherche le plus petit bit j de cette entrée qui n'est pas égal à 1. Alors, le nombre $128i+2j+seed$ n'a toujours pas été écrit comme une somme de deux nombres premiers. Le programme cherche alors la plus petite valeur k pour laquelle $128(i-k) + seed - 3$ et $128k + 2j + 3$ sont tous les deux premiers. Quand une telle valeur de k est trouvée, le tableau SIEVE commençant à l'entrée i peut être combiné avec le tableau ODD commençant à l'entrée k avec une opération "ou" comme précédemment. Le fait d'avoir un écart

de 128 dans la recherche des nombres premiers ne change pas la densité des nombres dont l'on s'attend à ce qu'ils soient premiers et a l'avantage d'éviter le décalage du tableau ODD. À la fin, pour gagner en efficacité, l'adressage dans le tableau SIEVE a été fait en utilisant une liste chaînée : cette liste contient seulement les valeurs i pour lesquelles SIEVE[i] n'est pas maximal. Alors après chaque opération "ou", la valeur résultante est comparée à $2^{64} - 1$ et s'il y a égalité, l'index correspondant est supprimé de la liste chaînée. Ainsi, la taille du tableau décroît au fur et à mesure du temps et globalement aucune opération "ou" inutile n'est faite. L'inconvénient est que l'adressage a été effectué par une redirection indirecte de pointeur et cela ralentit le programme au début de l'exécution. Les implémentations des versions avec et sans liste chaînée ont été testées sur une station DECSTATION 3100 avec des longueurs de mots variées et des tailles différentes des tableaux ODD et SIEVE. Le gain apporté par la version avec liste chaînée est apparu comme étant maximal pour des tableaux d'une longueur de 1.5×10^6 mots de 32 bits, avec un facteur de 1.59. Plus tard, quelques comparaisons ont été faites avec une version avec des entrées qui sont les nombres premiers jusqu'à 10^9. Le ratio entre les temps d'exécution était égal à 0.82 au bénéfice des dernières versions. D'autres améliorations n'ont pas été implémentées, e.g. anticipant les décompositions en blocs de nombres pairs suivant celui du tableau courant SIEVE, quand les indices sortent des limites de ce second tableau.

Des exécutions typiques consistent à vérifier 1350 intervalles consécutifs de nombres pairs de longueur 128.10^6 avec une exécution sur chacun des 18 processeurs R8000 de la station de travail SGI. Sept telles exécutions furent nécessaires pour traiter un intervalle de 2.10^{13}. Les intervalles qui ont été vérifiés sont $[10^{13}, 2.10^{13}]$, $[4.10^{13}, 6.10^{13}]$, et $[8.10^{13}, 9.10^{13}]$. Un nombre total de 324 exécutions a été nécessaire pour réaliser la tâche dans son intégralité. Les temps CPU utilisateur pour les différentes exécutions ont varié de 10 heures 33 mn pour l'exécution commençant à 9 158 401 000 000 et finissant à 9 331 201 000 000, jusqu'à 17 heures 12 mn pour l'exécution de 2 937 601 000 000 à 3 110 401 000 000. Le temps séquentiel total a été de 4083 heures 38 mn et le temps réel, qui est environ 18 fois plus petit, est environ 227 heures. Ces temps incluent la recherche des nombres premiers et le criblage. Le nombre de nombres premiers nécessités pour vérifier la décomposition de 64.10^6 entiers pairs consécutifs varie de 160 pour les intervalles commençant à 16 182 785 000 000 et 53 917 312 000 000, à 184 pour les intervalles commençant à 145 793 000 000. Quand on a vérifié sur les intervalles de longueur 10^9, le nombre moyen de nombres premiers a grossi jusqu'à 218.

5. Vérifier BGC près des hautes puissances de dix

En plus d'étendre G_2, nous avons également vérifié la conjecture binaire de Goldbach sur des intervalles de longueur 10^8 et 10^9 près des grandes puissances de dix. Le second auteur a vérifié les intervalles $[10^{5i}, 10^{5i} + 10^8]$, pour $i = 3, 4, \ldots, 20$, et le troisième auteur a vérifié les intervalles $[10^{10i}, 10^{10i} + 10^9]$, pour $i = 20, 21, \ldots, 30$.

5.1 Les intervalles $[10^{5i}, 10^{5i} + 10^8]$*, pour* $i = 3, 4, \ldots, 20$

Pour chaque intervalle $[B, B + 10^8]$, les 300 plus grands nombres premiers $\leq B$ ont été générés. Ici, les résultats de Jaeschke ne peuvent plus être utilisés parce que les nombres sont trop grands. À la place, nous avons d'abord généré les 300 plus grands nombres $\leq B$ qui sont passés à travers un test de pseudo-premier fort pour une base sélectionnée au hasard, et après nous avons prouvé la primalité de ces nombres avec un programme développé par H. Cohen, A.K. Lenstra, et D.T. Winter [5] : tous ces nombres s'avérèrent être premiers. Pour l'ensemble $\mathcal{P}$, nous avons pris les nombres premiers impairs inférieurs à $10^8 + 10^6$. La technique de crible était la même que celle utilisée sur le Cray C916 pour les nombres pairs jusqu'à 10^{14}.

Une sélection des résultats est donnée dans la Table 1. La seconde colonne donne la valeur de $(q_{300} - q_1) = (299 \log 10)$ qui serait proche de $\log_{10} B$, selon le théorème des nombres premiers. Elle illustre que le comportement local des nombres premiers peut dévier considérablement du comportement global connu. Le nombre moyen d'étapes nécessaire (sur les 18 intervalles considérés) était 217, avec une déviation standard de 23. Pour une distribution *uniforme* des bits dans le tableau ODD (plutôt que la distribution induite par les nombres premiers), le nombre moyen d'étapes a été 152, avec une déviation standard de 9. Cela s'accorde bien avec le nombre attendu d'étapes (214 dans le cas des nombres premiers et 150 dans le cas d'une distribution uniforme) mentionnée en Section 3.

5.2 Les intervalles $[10^{10i}, 10^{10i} + 10^9]$*, pour* $i = 20, 21, \ldots, 30$

À nouveau, le serveur de calcul SGI a été utilisé pour faire une implémentation similaire. Pour un intervalle de la forme $[B, B + 10^9]$, comme dans l'implémentation pour les décompositions jusqu'à 10^{14}, les nombres pairs étaient représentés par des bits dans un tableau de 7812500 mots de 64-bit. La technique du crible était la même que précédemment et utilisait aussi des listes chaînées. Pourtant, à cause de la taille des nombres, à nouveau les résultats de Jaeschke ne pouvaient pas être utilisés pour établir la primalité. Au lieu de ça, nous avons fait passer aux nombres candidats le test de pseudo-primalité de Miller-Rabin pour les bases 2, 3, 5 et 7 après un crible rapide d'essais de divisions. L'implémentation de cette phase a été faite en utilisant le système PARI. Dans une seconde phase, nous avons certifié la primalité de ces nombres par le programme de François Morain [1, 11] de preuve de la primalité par les courbes elliptiques. Sur un nœud R10000, les temps CPU pour la version C d'ECPP, que le troisième auteur avait à sa disposition variaient de 4 minutes pour des nombres à 200 chiffres décimaux à 60 minutes pour des nombres à 300 chiffres décimaux. Comme comparaison, la primalité de certains de ces nombres a été prouvée par le programme de Cohen, Lenstra et Winter [5] (pour des nombres jusqu'à 220 chiffres décimaux ; le temps CPU était de deux minutes par nombre sur une station de travail 180 MHZ IP32 SGI) et par le programme de Bosma et Van der Hulst [2] (pour les nombres ayant plus de 220 chiffres décimaux ; le temps moyen CPU était de sept minutes par nombre sur la même station de travail 180 MHZ IP32 SGI [3]). Le nombre de nombres premiers nécessité pour vérifier la BGC sur un intervalle de longueur 10^9 était en fait assez stable, variant de 222 à 231 pour les intervalles considérés avec une valeur moyenne de 225.
La Table 2 résume les résultats.

Table 1 Vérifier la conjecture de Goldbach sur les intervalles $[B, B + 10^8]$, pour $B = 10^{15}, 10^{20}, \ldots, 10^{100}$.

Notation

3. Le temps CPU nécessité par ce programme grossit avec la taille du nombre premier, mais de façon très erratique.

$[B, B+10^8]$: l'intervalle sur lequel la conjecture de Goldbach est vérifiée ;

$q_1, \ldots, q_{300}$: les plus grands 300 nombres premiers consécutifs $\leq B$, générés sur une station de travail SGI avec un processeur 100 MHZ IP22 ;

T_{pr} : la somme des temps-CPU en minutes utilisés pour générer les 300 plus grands nombres pseudo-premiers forts $< B$ (qui ont passé un test de pseudo-primalité sévère pour une base choisie au hasard) avec le package PARI et dont la primalité a été prouvée avec un code Fortran/C basé sur l'algorithme de preuve de primalité de Cohen-Lenstra (tous les pseudo-premiers "forts" se sont avérés être premiers) ;

N_1 : le plus petit entier positif tel que pour chaque nombre pair $e \in [B, B+10^8]$, il y a un indice i avec $1 \leq i \leq N_1$ tel que $e - q_i$ est un nombre premier ;

W : le "pire des cas" dans le test de la conjecture de Goldbach de l'intervalle $[B, B+10^8]$, i.e., $W - q_i$ est composé pour $i = 1, 2, \ldots, N_1 - 1$, mais premier pour $i = N_1$.

$\log_{10} B$	$\frac{q_{300}-q_1}{299 \log 10}$	$B - q_1$	$B - q_{300}$	T_{pr}	N_1	$W - B$
15	16.2	11159	11	1.9	243	87831838
20	19.8	13611	11	2.4	210	40249602
25	24.6	17063	123	3.0	240	91143618
30	31.9	21941	11	4.2	216	70421718
35	34.1	23477	23	5.7	182	84348372
40	37.2	25649	17	6.0	202	61919718
45	51.5	35469	9	8.0	283	80017866
50	45.3	31247	57	11	198	84955228
55	56.8	39183	111	16	218	88062574
60	58.9	40721	161	25	210	68370894
65	66.4	45951	269	32	193	80085838
70	72.6	50093	93	43	210	56324104
75	80.7	55779	191	53	224	31058458
80	82.6	56907	11	65	206	24403128
85	76.8	52919	27	82	203	45500944
90	95.6	65981	143	94	209	70588714
95	92.9	64011	53	122	207	88980634
100	99.0	68969	797	150	250	41229036

Table 2 Vérifier la conjecture de Goldbach sur les intervalles $[B, B+10^9]$, pour $B = 10^{200}, 10^{210}, \ldots, 10^{300}$.

Notation

$[B, B+10^9]$	:	l'intervalle sur lequel la conjecture de Goldbach est vérifiée ;
$q_1, q_2, \ldots$	:	la liste des nombres premiers nécessaire à la vérification de BGC sur $[B, B+10^9]$;
N_q	:	la cardinalité de l'ensemble précédent ;
T_{gen}	:	le temps nécessaire pour cribler l'intervalle et pour générer les N_q nombres pseudo-premiers forts avec le package PARI sur un seul nœud du SGI ;
T_{pp}	:	le temps séquentiel total utilisé par ECPP pour prouver la primalité des N_q pseudo-premiers (cette tâche a été en fait distribuée sur tous les nœuds du serveur de calcul) ;
N_{def}	:	le nombre de pseudo-premiers qui n'ont pas pu être certifiés par ECPP ;
W	:	le "pire des cas" pour la vérification de BGC dans l'intervalle, i.e. $W - q_i$ est composé pour $i = 1, 2, \ldots, N_q - 1$ mais premier pour $i = N_q$.

$\log_{10} B$	N_q	$\frac{qN_q - q_1}{64.N_q.\log 10}$	$B - q_1$	$qN_q - B$	$W - B$
200	224	30281.7	97283	999497853	999786382
210	222	30559.2	243203	999503869	999686796
220	222	30557.9	177539	999530109	999620578
230	228	29754.9	112643	999634941	999983752
240	228	29763.3	191747	999836541	999872854
250	231	29381.5	701699	999488509	999991806
260	226	30026.2	174467	999837181	999864924
270	223	30418.1	112643	999503997	999697006
280	225	30151.7	32003	999714813	999837064
290	226	30023.9	115331	999819517	999872646
300	227	29885.3	32771	999692157	999821434

$\log_{10} B$	T_{gen}	T_{pp}	N_{def}
200	33 mn 52 s	14 h 20 mn 02 s	4
210	35 mn 56 s	39 h 20 mn 31 s	5
220	44 mn 47 s	48 h 45 mn 05 s	21
230	48 mn 43 s	62 h 57 mn 39 s	25
240	55 mn 32 s	72 h 20 mn 47 s	23
250	1 h 11 mn 38 s	91 h 10 mn 43 s	19
260	1 h 16 mn 42 s	104 h 25 mn 31 s	17
270	1 h 13 mn 02 s	120 h 35 mn 28 s	15
280	1 h 45 mn 09 s	98 h 31 mn 15 s	27
290	1 h 39 mn 24 s	177 h 37 mn 48 s	17
300	2 h 04 mn 44 s	219 h 54 mn 59 s	41

Bibliographie

[1] A.O.L. Atkin and F. Morain. *Elliptic curves and primality proving*, Mathematics of Computation, 61 :29-68, 1993.

[2] W. Bosma and M.-P. van der Hulst. *Primality proving with cyclotom*, PhD thesis, University of Amsterdam, December 1990.

[3] J.R. Chen and T.Z. Wang, *On the odd Goldbach problem*, Acta Math. Sinica **32** (1989), p. 702-718 (in Chinese).

[4] J.R. Chen and T.Z. Wang, *On odd Goldbach problem under General Riemann Hypothesis*, Science in China **36** (1993), p. 682-691.

[5] H. Cohen and A.K. Lenstra, *Implementation of a new primality test*, Math. Comp. **48** (1987), p. 103-121.

[6] J-M. Deshouillers, G. Effinger, H. te Riele and D. Zinoviev, *A complete Vinogradov 3-primes theorem under the Riemann hypothesis*, Electronic Research Announcements of the AMS **3** (1997), p. 99-104 (September 17, 1997). http://www.ams.org/journals/era/home-1997.html.

[7] A. Granville, J. van de Lune and H.J.J. te Riele, *Checking the Goldbach conjecture on a vector computer*, Number Theory and Applications (R.A. Mollin, ed.), Kluwer, Dordrecht, 1989, p. 423-433.

[8] G.H. Hardy and L.E. Littlewood, *Some problems of 'Partitio Numerorum'; III : On the expression of a number as a sum of primes*, Acta Math. **44** (1922/3), p. 1-70.

[9] G. Jaeschke, *On strong pseudoprimes to several bases*, Math. Comp. **61** (1993), p. 915-926.

[10] L. Kaniecki, *On Šnirelman's constant under the Riemann hypothesis*, Acta. Arithm. **72** (1995), p. 361-374.

[11] F. Morain. *Courbes Elliptiques et Tests de Primalité*, PhD thesis, L'Université Claude Bernard, Lyon I, September 1990. Introduction in French, body in English.

[12] O. Ramaré, *On Šnirelman's Constant*, Ann. Scuola Norm. Sup. Pisa **22** (1995), p. 645-706.

[13] Y. Saouter, *Checking the odd Goldbach conjecture up to* 10^{20}, Math. Comp., **67** (1998), p. 863-866.

[14] L. Schoenfeld, *Sharper Bounds for the Chebyshev Functions* $\theta(x)$ *and* $\psi(x)$ II, Math. Comp. **30** (1976), p. 337-360.

[15] M.-K. Shen, *On Checking the Goldbach conjecture*, BIT **4** (1964), p. 243-245.

[16] M.K. Sinisalo, *Checking the Goldbach conjecture up to* 4.10^{11}, Math. Comp. **61** (1993), p. 931-934.

[17] M.L. Stein and P.R. Stein, *Experimental results on additive 2 bases*, Math. Comp. **19** (1965), p. 427-434.

[18] I.M. Vinogradov, *Representation of an odd number as a sum of three primes*, Comptes Rendus (Doklady) de l'Académie des Sciences de l'URSS, **15** (1937), p. 291-294.

[19] D. Zinoviev, *On Vinogradov's constant in Goldbach's ternary problem*, J. Number Th. **65** (1997), p. 334-358.

Des formes (Denise Vella-Chemla, 27.9.2020)

On voudrait fournir ici des visualisations produites par de multiples expérimentations informatiques, qui nous semblent bien faire appréhender le lien nombre-forme. Ces visualisations commencent par surprendre, puis ce qui est découvert *de visu* peut ensuite être rationnellement compris (sans pour autant qu'on le démontre).

Dernièrement, on a choisi de représenter les nombres par des points sur un disque "normalisé". On se place dans une sorte de "base de représentation", un ensemble $E = \{b_1, b_2, \ldots\}$ de nombres non obligatoirement premiers, et selon lesquels vont être calculés les restes modulaires des nombres successifs. On représente alors géométriquement chaque nombre entier x (inférieur à une certaine valeur) par sa somme de complexes associée :

$$g(x) = \sum_{k=1}^{k=|E|} e^{\frac{2i\pi(x \bmod b_k)}{b_k}}$$

Les programmes utilisés sont une variation du programme python ci-dessous :

```
import math
from math import *
import matplotlib
from matplotlib import *
import matplotlib.pyplot as plt
from matplotlib.pyplot import *

def prime(atester):
  pastrouve = True ; k = 2 ;
  if (atester in [0,1]): return False ;
  if (atester in [2,3,5,7]): return True ;
  while (pastrouve):
    if ((k * k) > atester): return True
    else:
      if ((atester % k) == 0): return False
      else: k=k+1

n = 42*42 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [41,43]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)

#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec= 0, 0
while x <= n:
  x0, y0 = 0, 0
  for element in Ens:
    t = 2*pi*(x % element)/element
    x0 = x0 + xmax*math.cos(t)
    y0 = y0 + xmax*math.sin(t)
  #print(str(x0)+'   '+str(y0))
  if (prime(x)):
    plt.plot(x0, y0, 'r', marker='o', markersize=1)
    ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
  #else:
    #plt.plot(x0, y0, 'g', marker='o', markersize=1)
    #ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
  #if (x != 1):
    #if (x <= 42*42/2):
      #ax.plot([xprec,x0],[yprec,y0],'red', alpha=0.25)
    #else:
      #ax.plot([xprec,x0],[yprec,y0],'blue', alpha=0.25)
  xprec = x0
  yprec = y0
  ax.set_aspect('equal')
  x = x+1
plt.show()
sys.exit(0)
```

Si on prend $E = \{3, 103\}$, et que l'on visualise successivement sur les trois dessins ci-dessous les nombres jusqu'à 103, puis jusqu'à 206, et jusqu'à 309, on obtient :

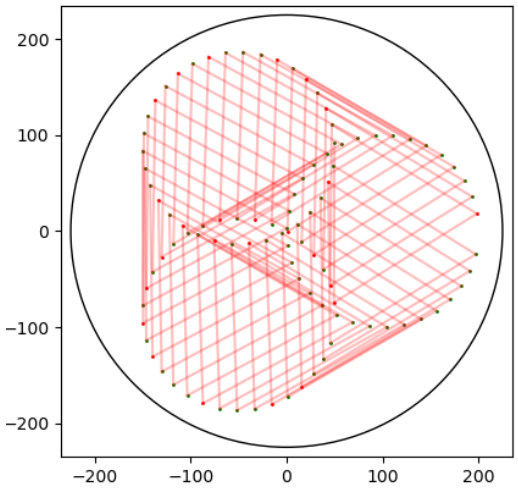


Fig. 1 : x < 103, E = {3,103}

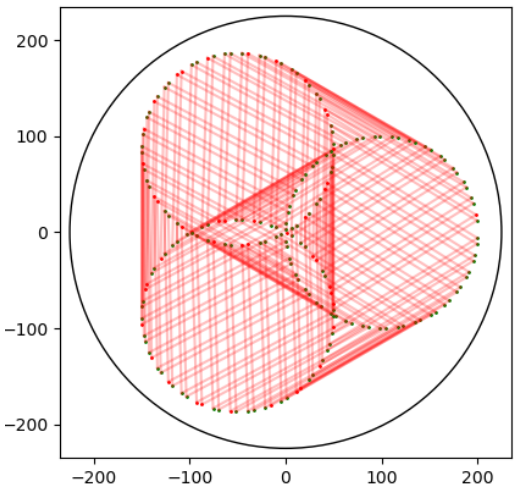


Fig. 2 : x < 206, E = {3,103}

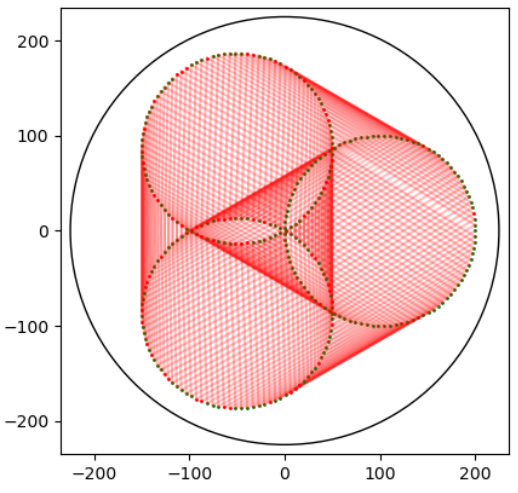


Fig. 3 : x < 309, E = {3,103}

En haut à gauche, il y a les nombres de la forme $3x+1$, en bas à gauche les nombres de la forme $3x+2$ et à droite les multiples de 3. Le parcours général pour passer d'un nombre au suivant suit la forme :

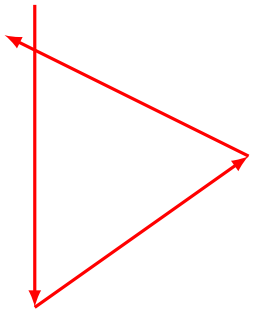


Fig. 4 : le "motif" de base parcouru dans les Figures 1, 2 et 3

Les trois cercles sont d'abord remplis au tiers de leur "espace", puis aux 2/3, puis complètement.

À l'origine de ces expérimentations, il y a le fait qu'on pensait que la démonstration qu'Alain Connes avait fournie pour le théorème de Morley était peut-être généralisable à tout polygone régulier quel qu'en soit le nombre de côtés : on avait cet espoir d'établir un petit lien "nombre, forme" (en oubliant grandeur mais elle y est par les racines de l'unité) en mettant en correspondance un polygone à neuf côtés, avec les trois triangles qui le constituent, en utilisant les racines de l'unité sur les polygones en question. Du côté des polynômes, au nombre 9 par exemple correspond le polynôme $1+x+x^2+x^3+x^4+x^5+x^6+x^7+x^8$ et de même qu'on a $9=3\times 3$, on a du côté des polynômes le fait qu'on peut factoriser : $1+x+x^2+x^3+x^4+x^5+x^6+x^7+x^8$ en $(1+x+x^2)+x^3(1+x+x^2)+x^6(1+x+x^2)$ et on voit ainsi que du côté des formes, on peut mettre des triangles dans le 9-gone. Cette idée est très naturelle et montre bien que "faire 9 fois un truc", c'est simplement "faire 3 fois un truc plus petit qui lui-même consiste à faire 3 fois un truc". Dans nos dessins, le truc consiste à suivre une fois le petit parcours de la figure 4.

On fournit ensuite les dessins en prenant $\mathrm{E}=\{5,103\}$ jusqu'à 515 et à droite, en ne "plottant" que les positions des seuls nombres premiers.

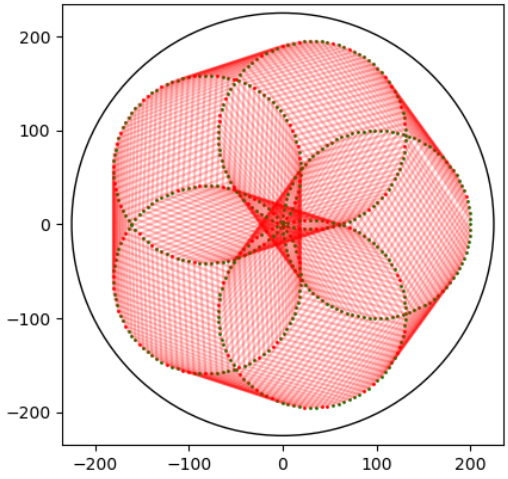


Fig. 5 : x < 515, E = {5,103}

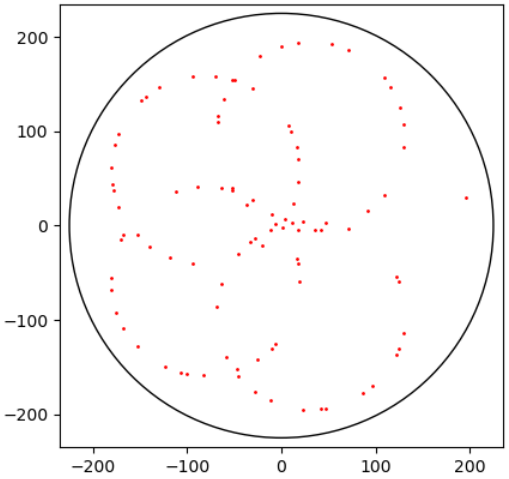


Fig. 6 : seuls les premiers, E = {5,103}

Prenons maintenant $\mathrm{E}=\{11,13\}$. On détecte bien la périodicité normale des restes qui fait que l'on "revient au début" pour le nombre $144=12\times 12$, qui est le nombre qui respecte le système de congruences

$$\begin{cases} x\equiv 1 \pmod{11} \\ x\equiv 1 \pmod{13} \end{cases}.$$

On a utilisé 4 couleurs pour voir l'atteinte du centre par la spirale "entrante vers 0" à 36 (elle part en rouge de 1, légèrement au nord-est), puis "repartant vers l'extérieur" jusqu'à 72 (en bleu), puis "rerentrant vers le centre" à 108 (en vert), puis (en orange) "repartant vers l'extérieur" pour rejoindre l'origine du graphique (le nombre 1, qui sera "repassé" par 144).

À droite, c'est un graphique selon la même idée exactement, si ce n'est qu'il est plus "touffu" car on a pris les restes selon les nombres premiers jumeaux 41 et 43 (et deux couleurs seulement, jusqu'à la moitié, ne laissant apparaître que la symétrie haut-bas et non la symétrie droite-gauche).

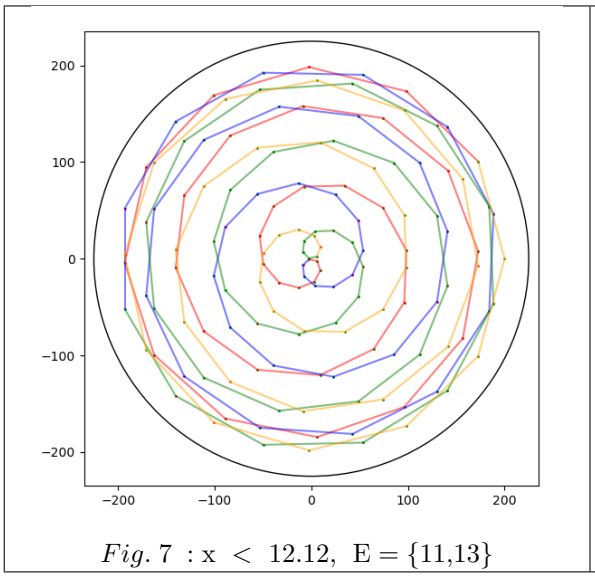


Fig. 7 : x < 12.12, E = {11,13}

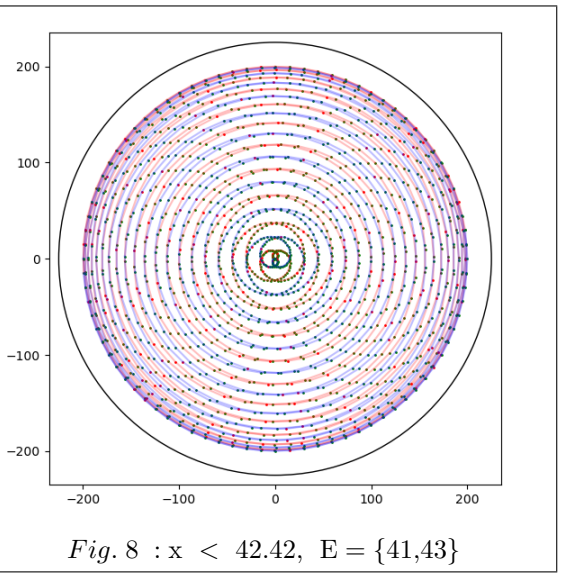


Fig. 8 : x < 42.42, E = {41,43}

Ci-dessous, on ne montre que les nombres premiers sur le dessin (mod 41, mod 43). On voit apparaître des alignements en rayons de roue de vélo, qu'on souhaite comprendre.

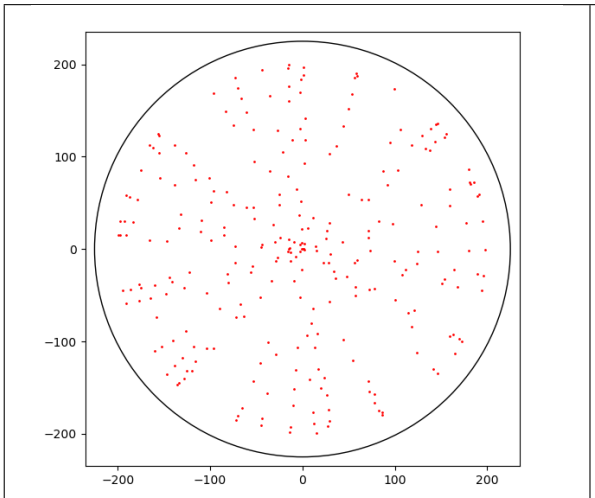


Fig. 9 : seuls les premiers, E = {41,43}

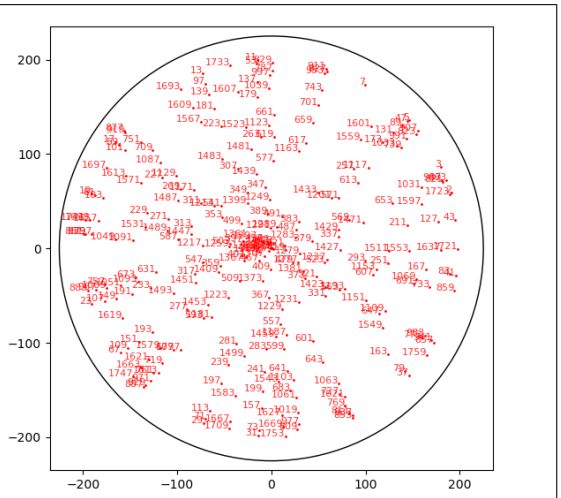


Fig. 10 : seuls les premiers, E = {41,43}

Prenons les nombres alignés 113, 197, 239 et 281 et écrivons les comme des couples de coordonnées par leurs restes modulaires selon les nombres premiers 41 et 43. Leur correspondent les couples :

$$\binom{31}{27}, \binom{33}{25}, \binom{34}{24}, \binom{35}{23}.$$

On voit que les coordonnées des points alignés sont linéairement liées. De même, pour les nombres 1511 $\binom{35}{6}$, 1553 $\binom{36}{5}$, 1637 $\binom{38}{3}$, 1721 $\binom{40}{1}$ ou enfin les nombres 1693 $\binom{12}{16}$, 1609 $\binom{10}{18}$, 1567 $\binom{9}{19}$ et 1483 $\binom{7}{21}$.

On constate également dans la visualisation de la Figure 8 (les restes modulaires utilisées sont ceux modulo 41 et 43) que du fait de la proximité des nombres premiers choisis (des nombres premiers jumeaux), les branches sortant d'un noeud sont proches, ce qui nous permet de voir parfois comme des petits doublons de points rapprochés. Étudions les propriétés de points proches : on a 179 $\binom{15}{7}$

qui est proche de 1081 $\binom{15}{6}$ d'un côté tandis qu'ailleurs, 704 $\binom{7}{16}$ est proche de 1564 $\binom{6}{16}$, à moins qu'il ne forme plutôt un petit doublon avec 1606 $\binom{7}{15}$. On découvre ces doublons *de visu*, et on constate ensuite qu'on a comme attendu $1506 - 1564 = 42$, ou bien $1564 - 704 = 860 = 43.20$ ou encore $1606 - 704 = 902 = 41.22$ La proximité de points sur le dessin correspond à des propriétés de divisibilités particulières des nombres que ces points représentent.

Ayant suffisamment étudié les nombres premiers jumeaux, on décide de passer aux nombres premiers dits "sexy", 23 et 29, pour voir se former au centre une rosace à 6 pétales (l'écart entre 23 et 29), ce qui semble logique : il y avait 2 pétales au centre pour les nombres premiers jumeaux. À droite, la visualisation habituelle ne "plotte" que les nombres premiers. Ensuite, sur la visualisation suivante (Fig. 13), on a l'idée de visualiser les restes modulaires selon d'autres bases, dont les deux nombres seraient aussi écartés de 6, mais qui ne seraient pas des nombres premiers, en l'occurrence 30 et 36 et là, la belle symétrie est brisée, seule la partie droite des schémas habituels est conservée. Comprenons ce qui se passe : quand revient-on au début ? i.e. quand retombe-t-on sur 1, c'est à dire quel est le nombre qui ait à la fois un reste de 1 quand on le divise par 30 et un reste de 1 quand on le divise par 36. 30 a pour factorisation 2.3.5 tandis que 36 est égal à $2^2.3^2$. Le nombre recherché est 180, dont la factorisation "couvre" au sens de l'inclusion ensembliste les deux factorisations de 30 et 36 : la factorisation de 180 est en effet $2^2.3^2.5$ qui "contient" en termes ensemblistes à la fois l'ensemble d'éléments comptés avec multiplicités $\{2, 3, 5\}$ et l'ensemble $\{2, 2, 3, 3\}$.

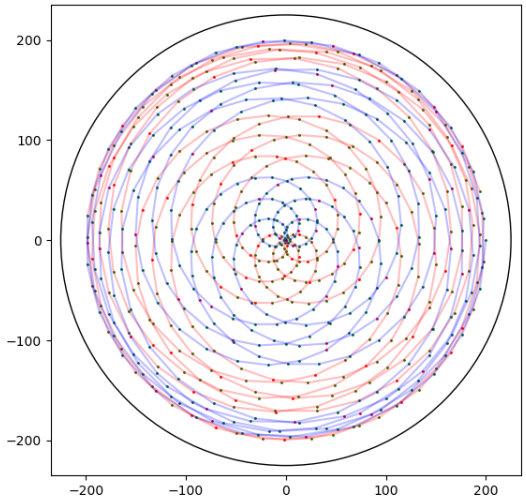


Fig. 11 : x < 23.29, E = {23,29}

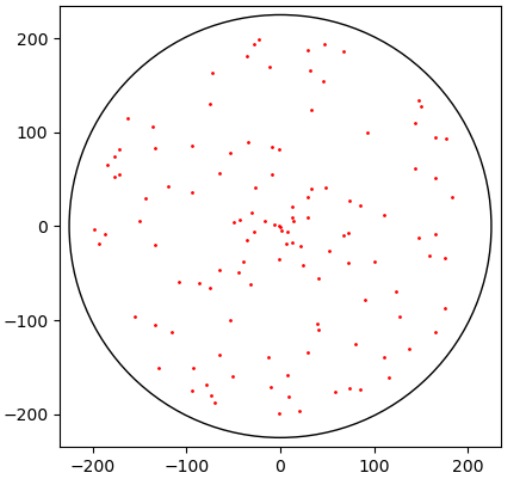


Fig. 12 : seuls les premiers, E = {23,29}

Pour comprendre encore (Fig. 14, 15 et 16), fixons-nous pour terminer sur les ensembles $E = \{3, 36\}$, puis $E = \{6, 36\}$ et enfin $E = \{12, 36\}$. On obtient les dernières visualisations ci-dessous. Ces 3 dernières visualisations illustrent également, que selon les angles que font le segment entrant en un point et le segment sortant de ce point, on a parfois une impression de continuité (de "lissité", Fig. 16) des courbes et parfois une impression d'être face à un caractère plus discret, moins lisse, plein d'aspérités et de singularités (Figures 15 et 14). On perçoit bien là la façon dont la continuité est liée à l'écart choisi entre les nombres, ce qu'on appelle l'unité de longueur, la granularité mesurant la sensation de rupture que l'on a lorsqu'on observe un dessin.

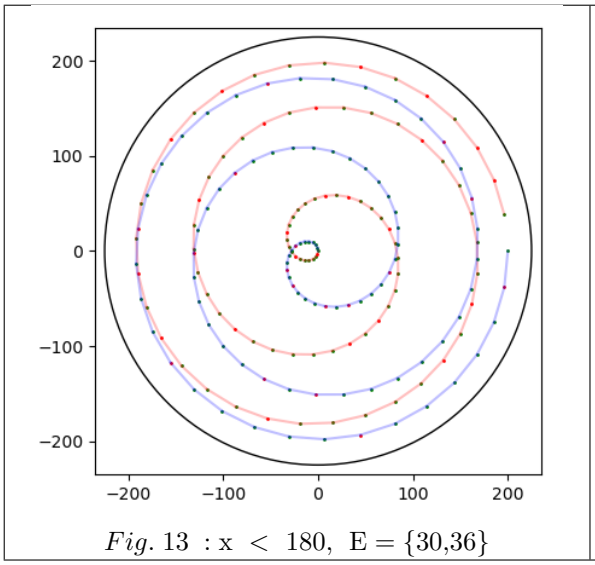


Fig. 13 : x < 180, E = {30,36}

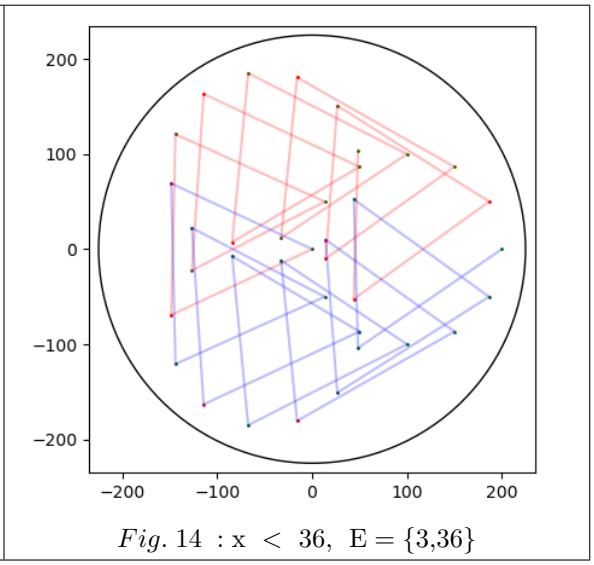


Fig. 14 : x < 36, E = {3,36}

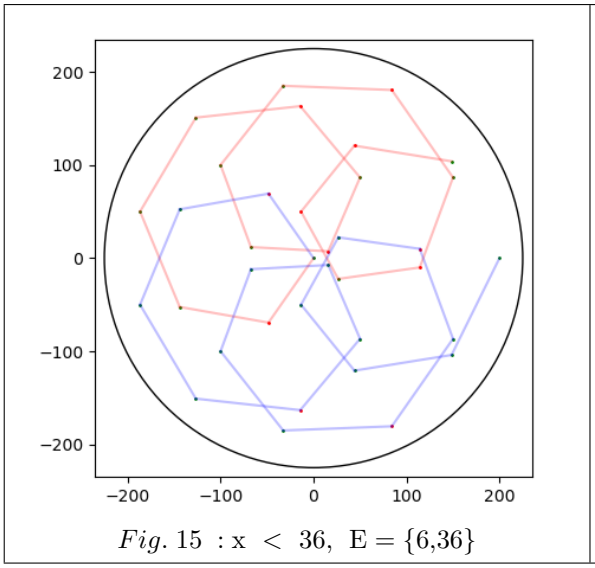


Fig. 15 : x < 36, E = {6,36}

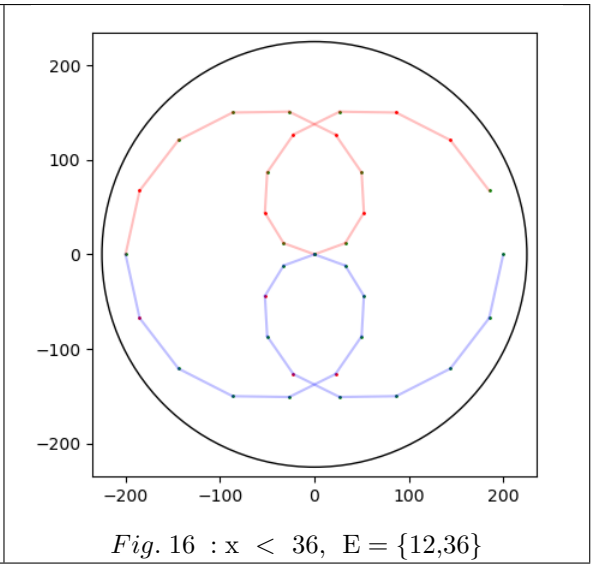


Fig. 16 : x < 36, E = {12,36}

Littérature

Alexander Grothendieck a écrit de superbes passages dans *Récoltes et semailles* sur le lien entre la forme, le nombre et la grandeur, reproduites ici : ce sont les plus belles phrases écrites sur cette quête qui anime les mathématiciens.

Traditionnellement, on distingue trois types de "qualités" ou d'"aspects" des choses de l'Univers, qui soient objet de la réflexion mathématique : ce sont le nombre[1]*, la grandeur, et la forme. On peut aussi les appeler l'aspect "arithmétique", l'aspect "métrique" (ou "analytique"), et l'aspect "géométrique" des choses. Dans la plupart des situations étudiées dans la mathématique, ces trois aspects sont présents simultanément et en interaction étroite.*

1. Il est entendu ici qu'il s'agit des "nombres" dits "entiers naturels" 0, 1, 2, 3 etc, ou (à la rigueur) des nombres (tels les nombres fractionnaires) qui s'expriment à l'aide de ceux-ci par des opérations de nature élémentaire. Ces nombres ne se prêtent pas, comme les "nombres réels", à mesurer une grandeur susceptible de variation continue, telle la distance entre deux points variables sur une droite, dans un plan ou dans l'espace.

[...]

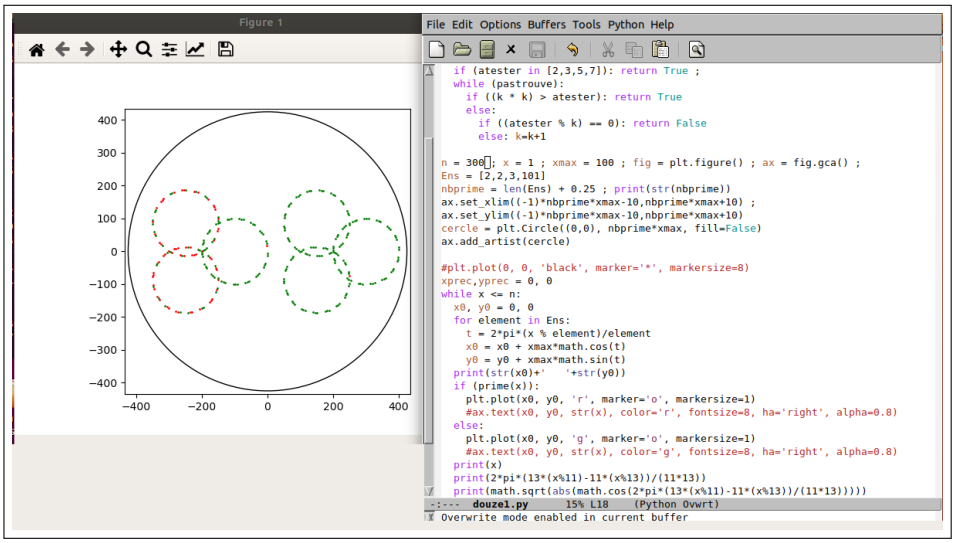
C'est dire que s'il y a une chose en mathématique qui (depuis toujours sans doute) me fascine plus que toute autre, ce n'est ni "le nombre", ni "la grandeur", mais toujours ***la forme****. Et parmi les mille-et-un visages que choisit la forme pour se révéler à nous, celui qui m'a fasciné plus que tout autre et continue à me fasciner, c'est la structure cachée dans les choses mathématiques.*

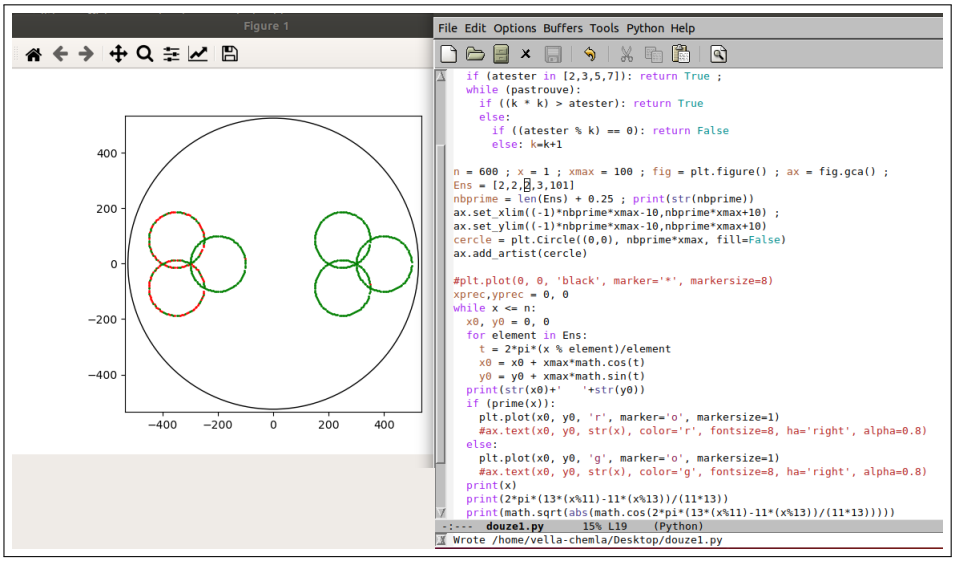
La structure d'une chose n'est nullement une chose que nous puissions "inventer". Nous pouvons seulement la mettre à jour patiemment, humblement en faire connaissance, la "découvrir". S'il y a inventivité dans ce travail, et s'il nous arrive de faire œuvre de forgeron ou d'infatigable bâtisseur, ce n'est nullement pour "façonner", ou pour "bâtir", des "structures". Celles-ci ne nous ont nullement attendues pour être, et pour être exactement ce qu'elles sont ! Mais c'est pour exprimer, le plus fidèlement que nous le pouvons, ces choses que nous sommes en train de découvrir et de sonder, et cette structure réticente à se livrer, que nous essayons à tâtons, et par un langage encore balbutiant peut-être, à cerner. Ainsi sommes-nous amenés à constamment "inventer" le langage apte à exprimer de plus en plus finement la structure intime de la chose mathématique, et à "construire" à l'aide de ce langage, au fur et à mesure et de toutes pièces, les "théories" qui sont censées rendre compte de ce qui a été appréhendé et vu. Il y a là un mouvement de va-et-vient continuel, ininterrompu, entre l'appréhension des choses, et l'expression de ce qui est appréhendé, par un langage qui s'affine et se re-crée au fil du travail, sous la constante pression du besoin immédiat.

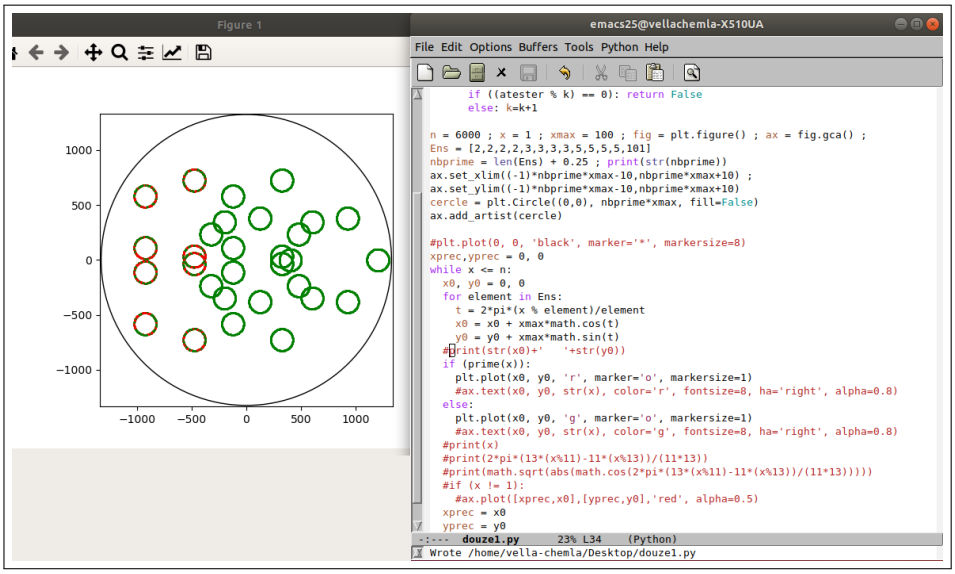
[...]

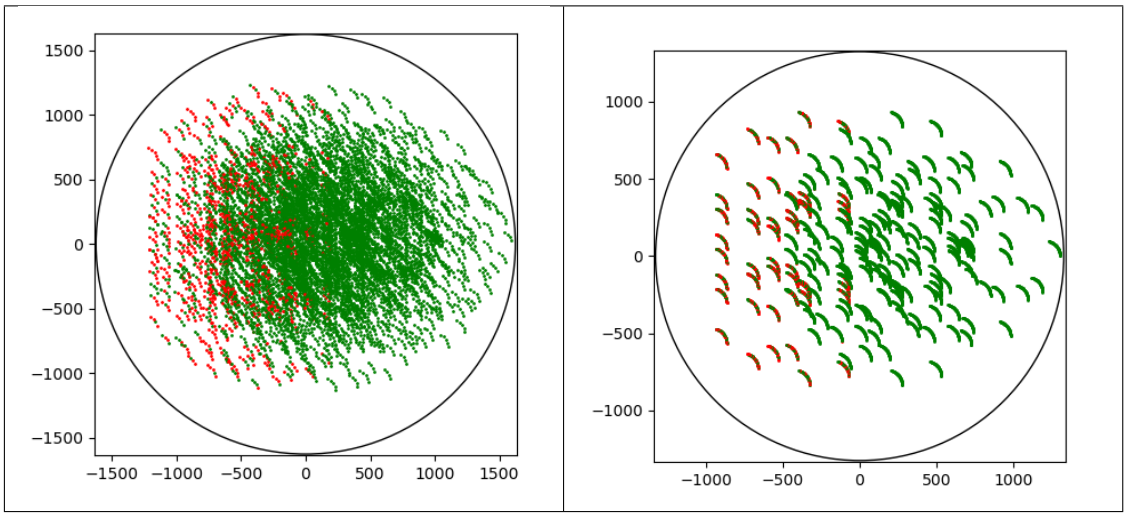
On peut dire que "le nombre" est apte à saisir la structure des agrégats "discontinus", ou "discrets" : les systèmes, souvent finis, formés d'"éléments" ou "objets" pour ainsi dire isolés les uns par rapport aux autres, sans quelque principe de "passage continu" de l'un à l'autre. "La grandeur" par contre est la qualité par excellence, susceptible de "variation continue" ; par là, elle est apte à saisir les structures et phénomènes continus : les mouvements, espaces, "variétés" en tous genres, champs de force etc. Ainsi, l'arithmétique apparaît (grosso modo) comme la science des structures discrètes, et l'analyse, comme la science des structures continues.

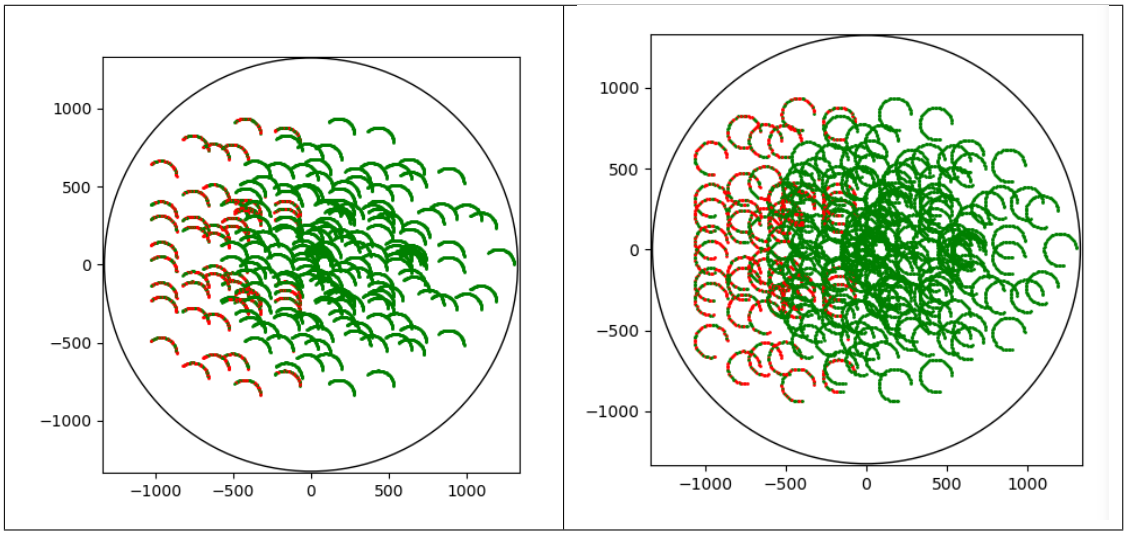
Dessins supplémentaires (Denise Vella-Chemla, 29.9.2020)

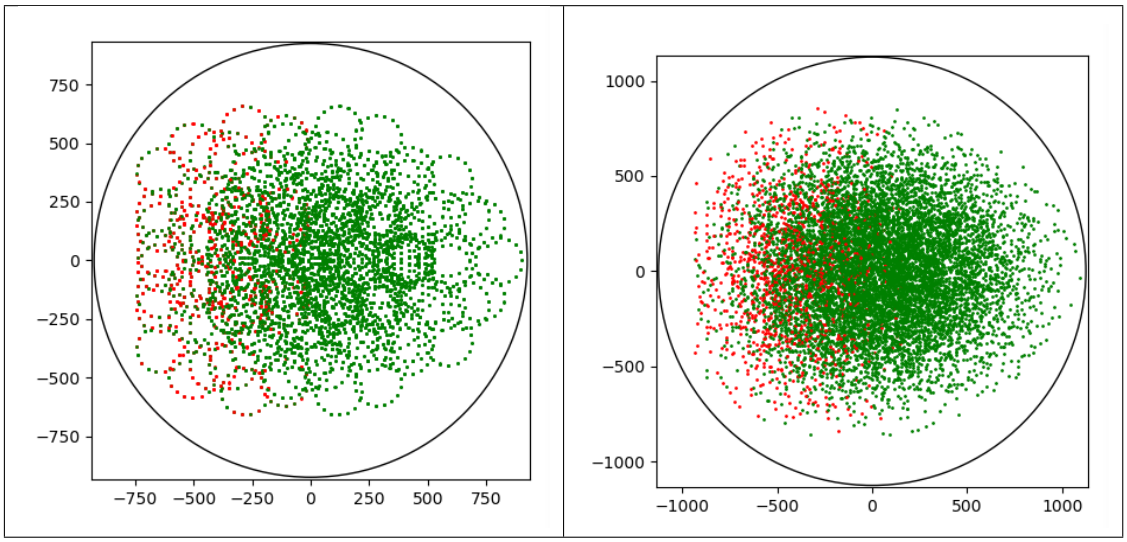







1500
1000
500
0
−500
−1000
−1500
−1500 −1000 −500 0 500 1000 1500
1000
500
0
−500
−1000
−1000 −500 0 500 1000


1000
500
0
−500
−1000
−1000 −500 0 500 1000
1000
500
0
−500
−1000
−1000 −500 0 500 1000


750
500
250
0
−250
−500
−750
−750 −500 −250 0 250 500 750
1000
500
0
−500
−1000
−1000 −500 0 500 1000

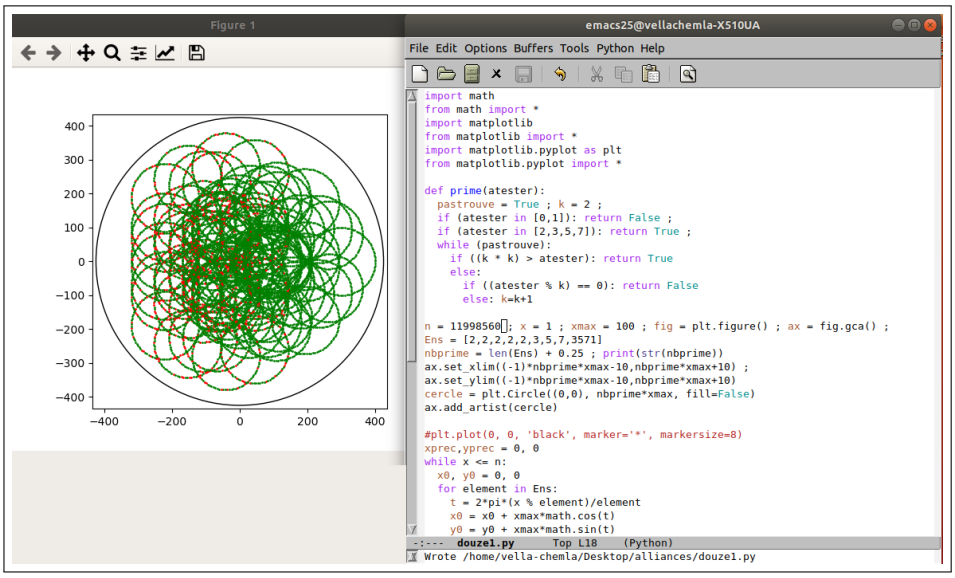


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from math import *
import matplotlib
from matplotlib import *
import matplotlib.pyplot as plt
from matplotlib.pyplot import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
n = 11998560; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [2,2,2,2,2,3,5,7,3571]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= n:
x0, y0 = 0, 0
for element in Ens:
t = 2*pi*(x % element)/element
x0 = x0 + xmax*math.cos(t)
y0 = y0 + xmax*math.sin(t)
-:--- douze1.py Top L18 (Python)
Wrote /home/vella-chemla/Desktop/alliances/douze1.py

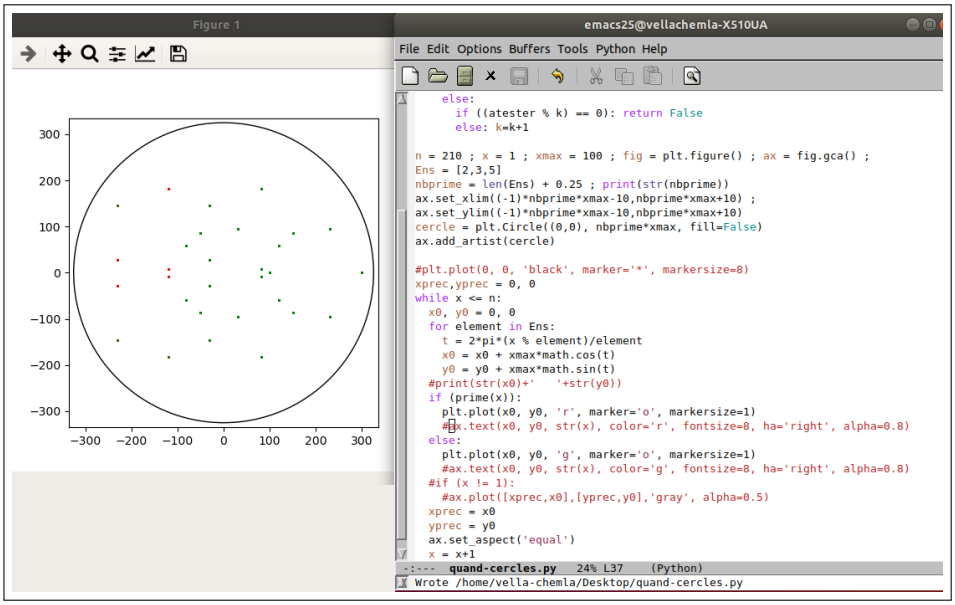


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
else:
if ((atester % k) == 0): return False
else: k=k+1
n = 210 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [2,3,5]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= n:
x0, y0 = 0, 0
for element in Ens:
t = 2*pi*(x % element)/element
x0 = x0 + xmax*math.cos(t)
y0 = y0 + xmax*math.sin(t)
#print(str(x0)+' '+str(y0))
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
#if (x != 1):
#ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
x = x+1
-:--- quand-cercles.py 24% L37 (Python)
Wrote /home/vella-chemla/Desktop/quand-cercles.py

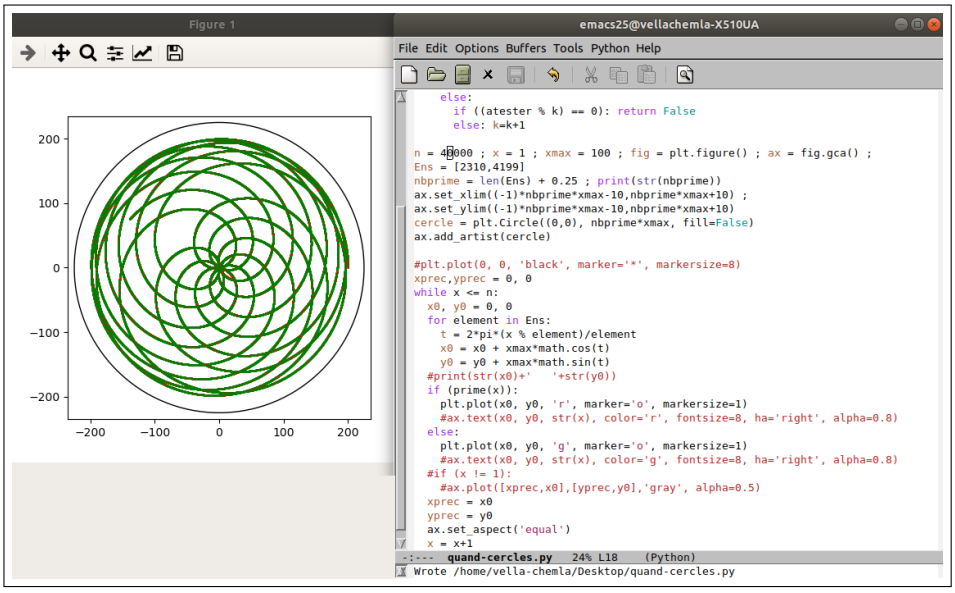


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
else:
if ((atester % k) == 0): return False
else: k=k+1
n = 40000 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [2310,4199]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= n:
x0, y0 = 0, 0
for element in Ens:
t = 2*pi*(x % element)/element
x0 = x0 + xmax*math.cos(t)
y0 = y0 + xmax*math.sin(t)
#print(str(x0)+' '+str(y0))
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
#if (x != 1):
#ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
x = x+1
-:--- quand-cercles.py 24% L18 (Python)
Wrote /home/vella-chemla/Desktop/quand-cercles.py

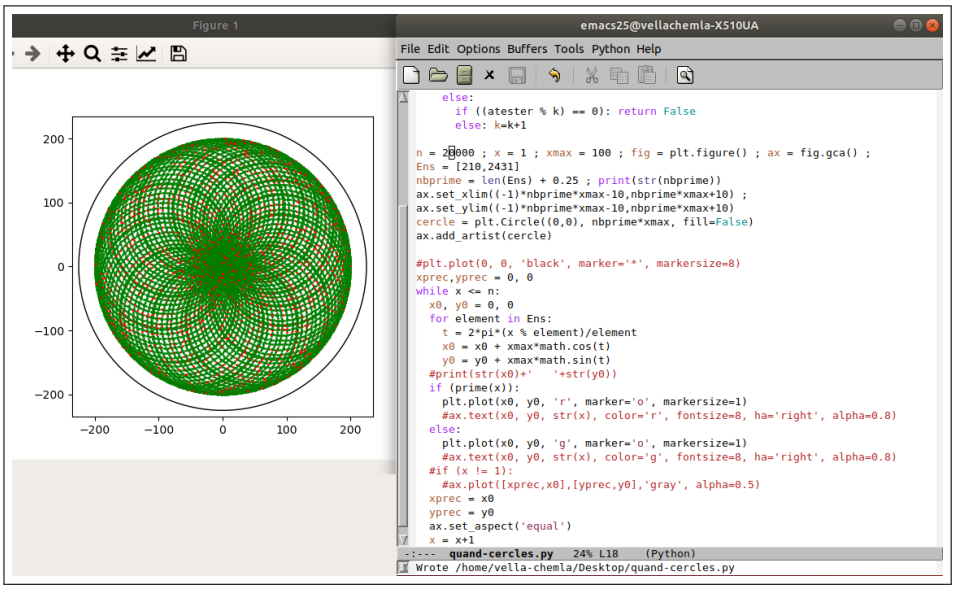


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
else:
if ((atester % k) == 0): return False
else: k=k+1
n = 20000 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [210,2431]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= n:
x0, y0 = 0, 0
for element in Ens:
t = 2*pi*(x % element)/element
x0 = x0 + xmax*math.cos(t)
y0 = y0 + xmax*math.sin(t)
#print(str(x0)+' '+str(y0))
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
#if (x != 1):
#ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
x = x+1
-:--- quand-cercles.py 24% L18 (Python)
Wrote /home/vella-chemla/Desktop/quand-cercles.py

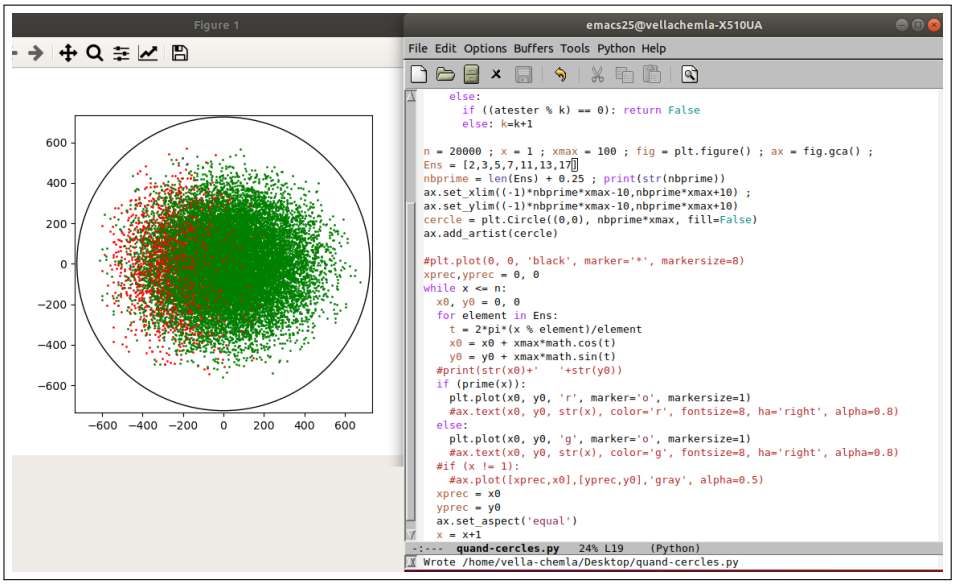


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
else:
if ((atester % k) == 0): return False
else: k=k+1
n = 20000 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [2,3,5,7,11,13,17]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= n:
x0, y0 = 0, 0
for element in Ens:
t = 2*pi*(x % element)/element
x0 = x0 + xmax*math.cos(t)
y0 = y0 + xmax*math.sin(t)
#print(str(x0)+' '+str(y0))
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
#if (x != 1):
#ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
x = x+1
-:--- quand-cercles.py 24% L19 (Python)
Wrote /home/vella-chemla/Desktop/quand-cercles.py

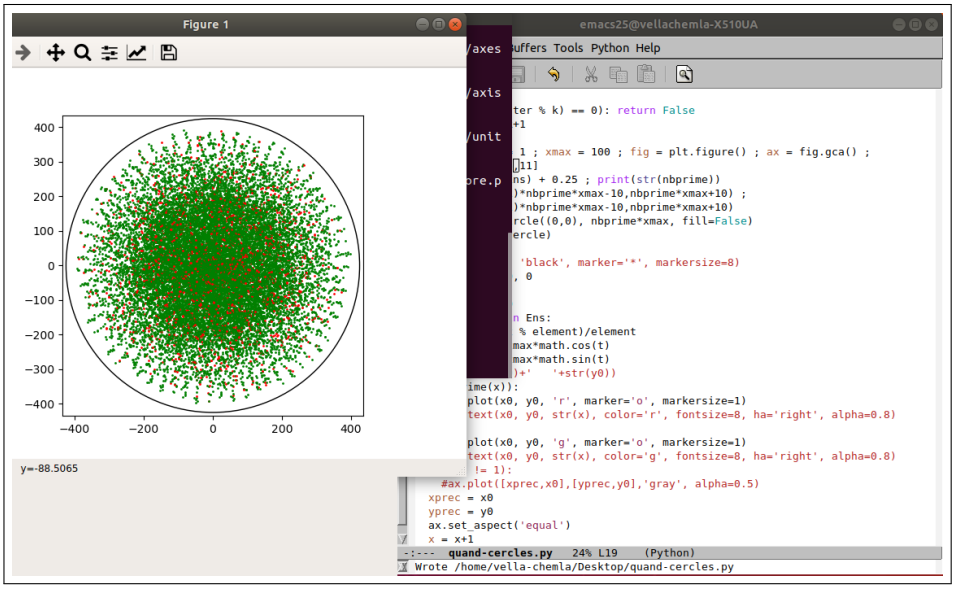


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
y=-88.5065
-:--- quand-cercles.py 24% L19 (Python)
Wrote /home/vella-chemla/Desktop/quand-cercles.py

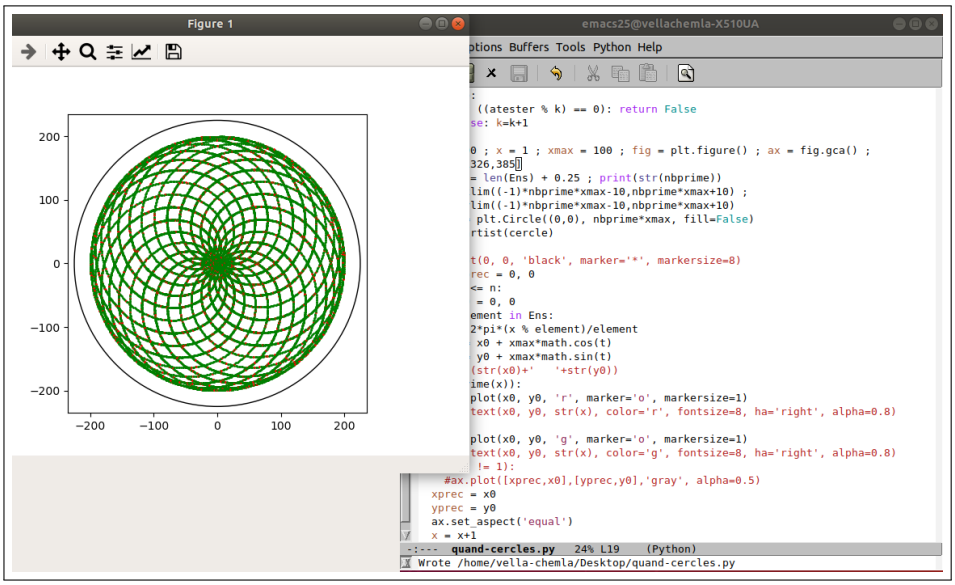
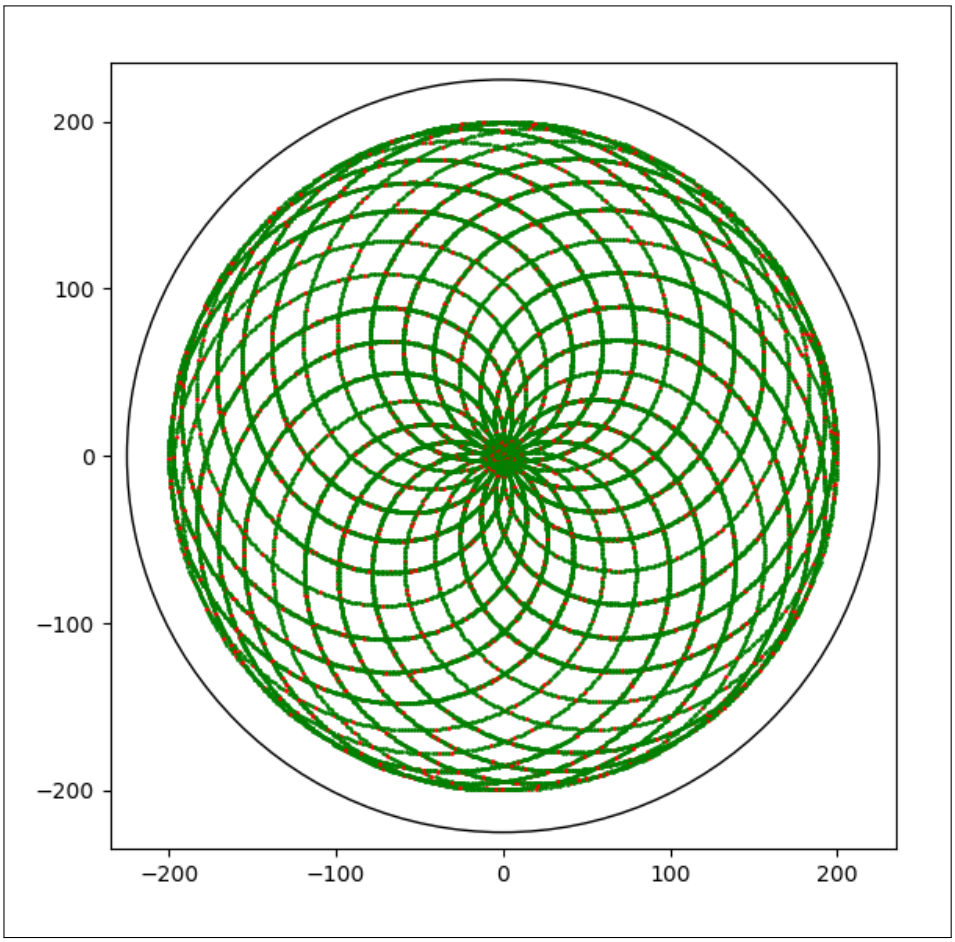


Figure 1
emacs25@vellachemla-X510UA
ptions Buffers Tools Python Help
((atester % k) == 0): return False
se: k=k+1
0 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
326,385
= len(Ens) + 0.25 ; print(str(nbprime))
lim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
lim((-1)*nbprime*xmax-10,nbprime*xmax+10)
plt.Circle((0,0), nbprime*xmax, fill=False)
rtist(cercle)
t(0, 0, 'black', marker='*', markersize=8)
rec = 0, 0
<= n:
= 0, 0
ement in Ens:
2*pi*(x % element)/element
x0 + xmax*math.cos(t)
y0 + xmax*math.sin(t)
(str(x0)+' '+str(y0))
ime(x)):
plot(x0, y0, 'r', marker='o', markersize=1)
text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
plot(x0, y0, 'g', marker='o', markersize=1)
text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
!= 1):
#ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
x = x+1
-:--- quand-cercles.py 24% L19 (Python)
Wrote /home/vella-chemla/Desktop/quand-cercles.py
200
100
0
−100
−200
−200
−100
0
100
200


200
100
0
−100
−200
−200
−100
0
100
200

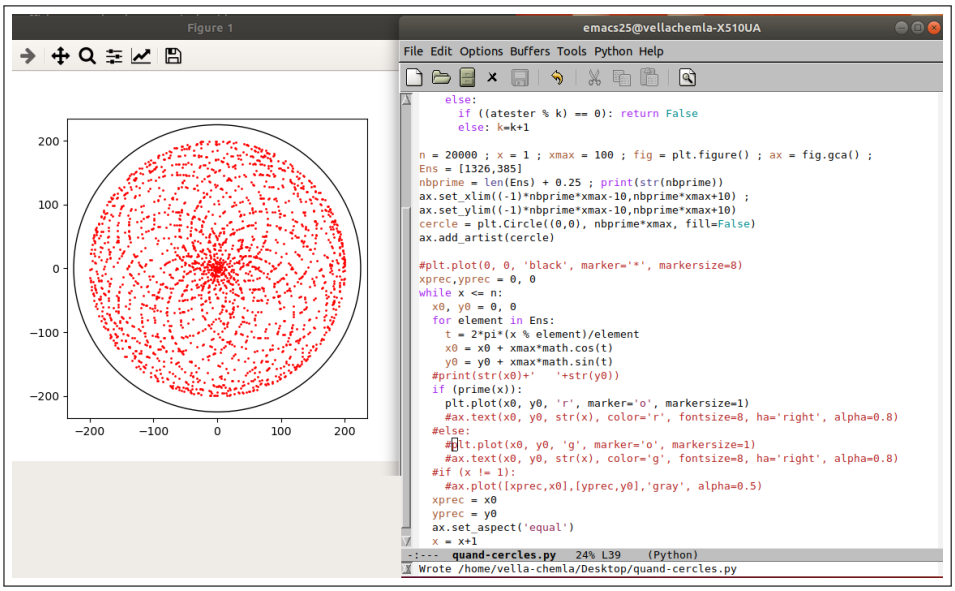


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
else:
if ((atester % k) == 0): return False
else: k=k+1
n = 20000 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [1326,385]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= n:
x0, y0 = 0, 0
for element in Ens:
t = 2*pi*(x % element)/element
x0 = x0 + xmax*math.cos(t)
y0 = y0 + xmax*math.sin(t)
#print(str(x0)+' '+str(y0))
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
#else:
#plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
#if (x != 1):
#ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
x = x+1
-:--- quand-cercles.py 24% L39 (Python)
Wrote /home/vella-chemla/Desktop/quand-cercles.py

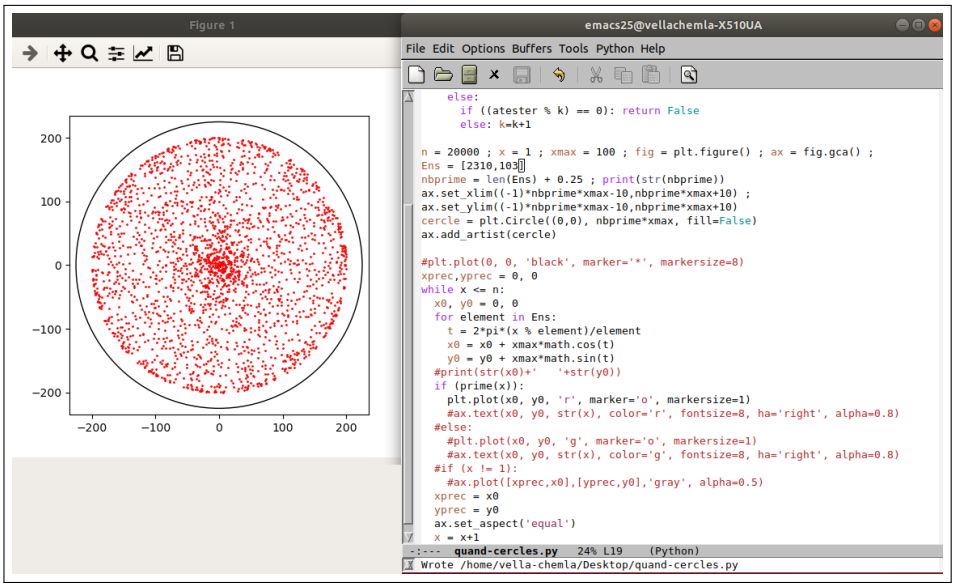


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
else:
if ((atester % k) == 0): return False
else: k=k+1
n = 20000 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [2310,103]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= n:
x0, y0 = 0, 0
for element in Ens:
t = 2*pi*(x % element)/element
x0 = x0 + xmax*math.cos(t)
y0 = y0 + xmax*math.sin(t)
#print(str(x0)+' '+str(y0))
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
#else:
#plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
#if (x != 1):
#ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
x = x+1
-:--- quand-cercles.py 24% L19 (Python)
Wrote /home/vella-chemla/Desktop/quand-cercles.py

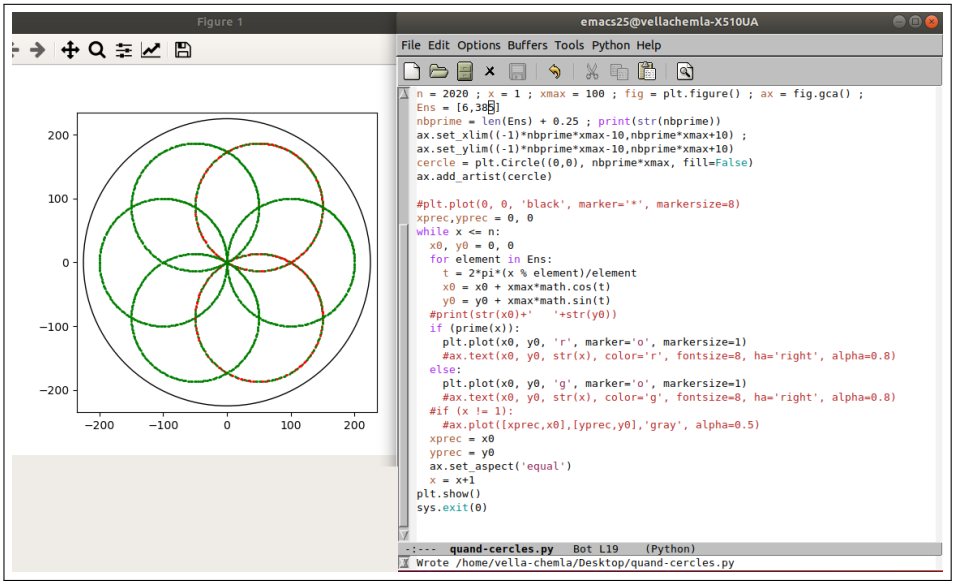


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
n = 2020 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [6,385]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= n:
x0, y0 = 0, 0
for element in Ens:
t = 2*pi*(x % element)/element
x0 = x0 + xmax*math.cos(t)
y0 = y0 + xmax*math.sin(t)
#print(str(x0)+' '+str(y0))
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
#if (x != 1):
#ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
x = x+1
plt.show()
sys.exit(0)
-:--- quand-cercles.py Bot L19 (Python)
Wrote /home/vella-chemla/Desktop/quand-cercles.py

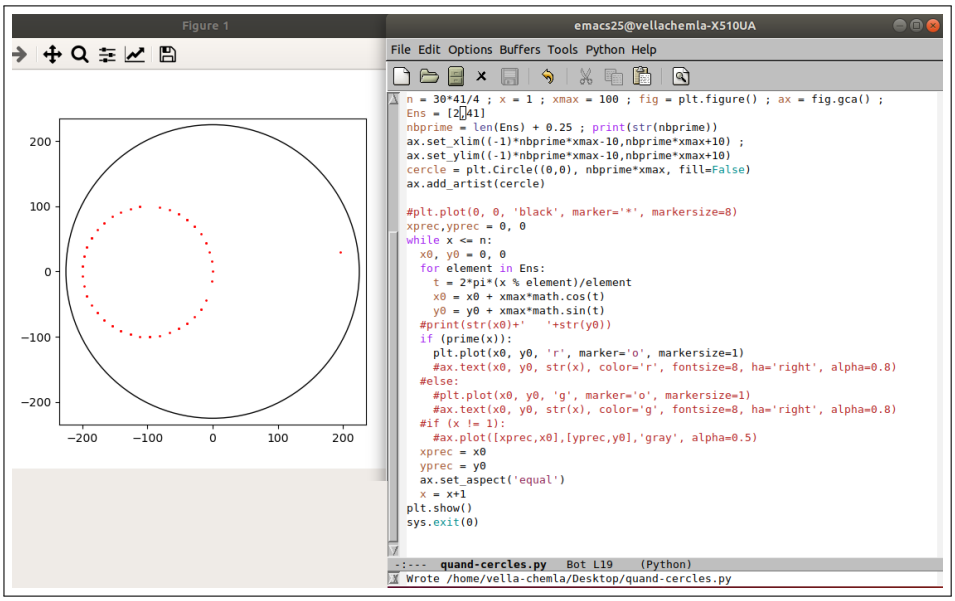


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
n = 30*41/4 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [2,41]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= n:
x0, y0 = 0, 0
for element in Ens:
t = 2*pi*(x % element)/element
x0 = x0 + xmax*math.cos(t)
y0 = y0 + xmax*math.sin(t)
#print(str(x0)+' '+str(y0))
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
#else:
#plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
#if (x != 1):
#ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
x = x+1
plt.show()
sys.exit(0)
-:--- quand-cercles.py Bot L19 (Python)
Wrote /home/vella-chemla/Desktop/quand-cercles.py

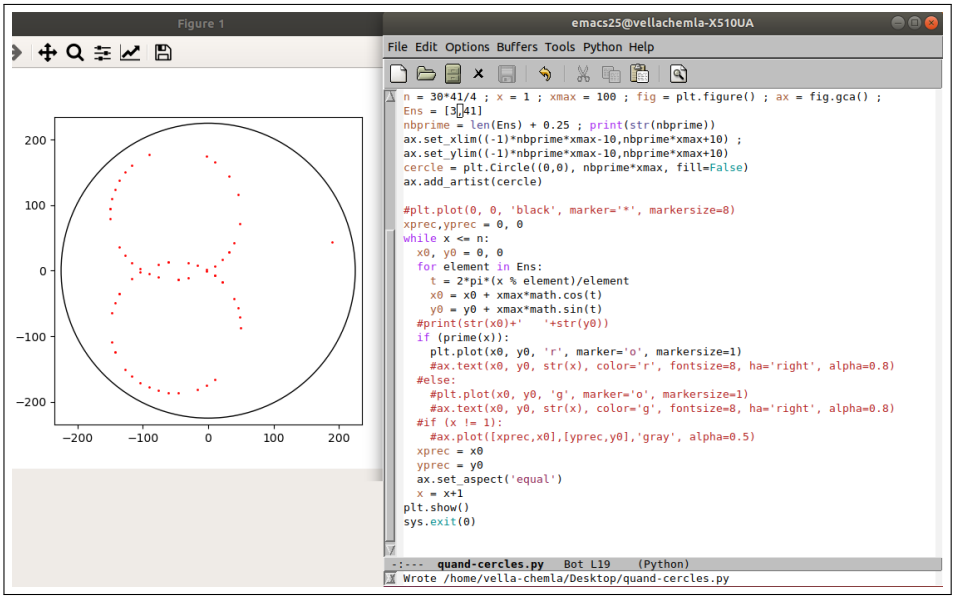


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
n = 30*41/4 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [3,41]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= n:
x0, y0 = 0, 0
for element in Ens:
t = 2*pi*(x % element)/element
x0 = x0 + xmax*math.cos(t)
y0 = y0 + xmax*math.sin(t)
#print(str(x0)+' '+str(y0))
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
#else:
#plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
#if (x != 1):
#ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
x = x+1
plt.show()
sys.exit(0)
-:--- quand-cercles.py Bot L19 (Python)
Wrote /home/vella-chemla/Desktop/quand-cercles.py

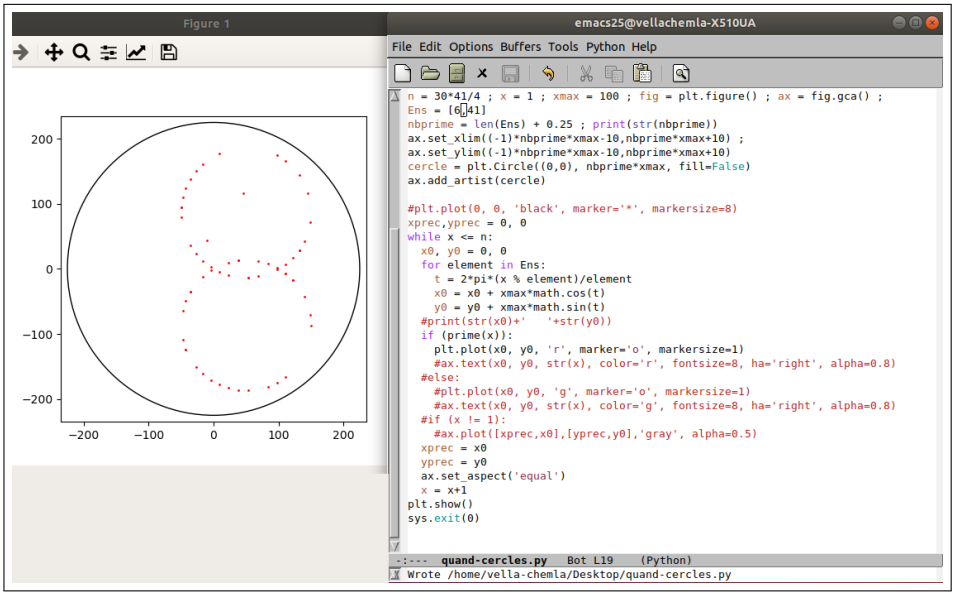


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
n = 30*41/4 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [6,41]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= n:
x0, y0 = 0, 0
for element in Ens:
t = 2*pi*(x % element)/element
x0 = x0 + xmax*math.cos(t)
y0 = y0 + xmax*math.sin(t)
#print(str(x0)+' '+str(y0))
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
#else:
#plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
#if (x != 1):
#ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
x = x+1
plt.show()
sys.exit(0)
-:--- quand-cercles.py Bot L19 (Python)
Wrote /home/vella-chemla/Desktop/quand-cercles.py

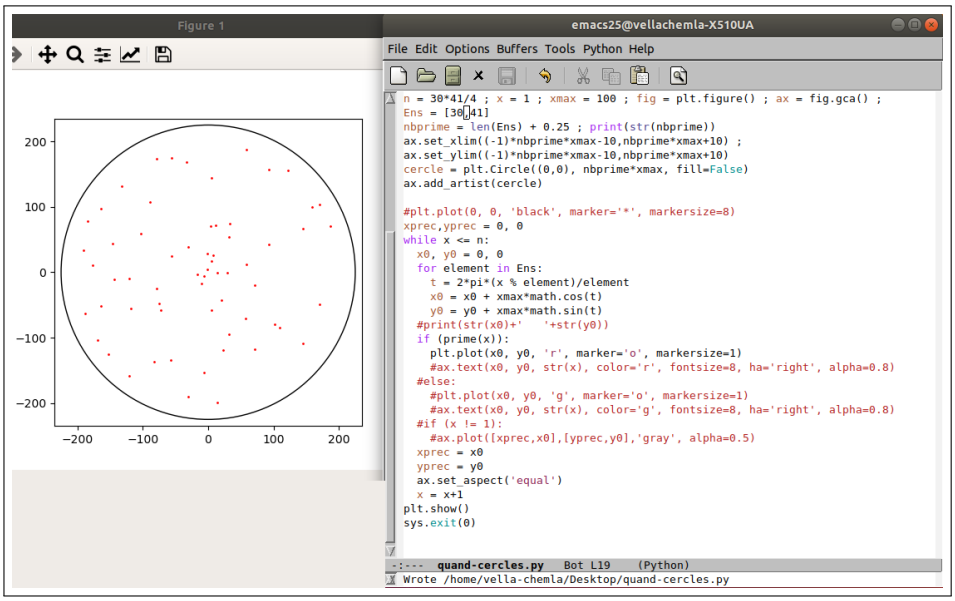


Figure 1
200
100
0
−100
−200
−200
−100
0
100
200
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
n = 30*41/4 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [30,41]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= n:
x0, y0 = 0, 0
for element in Ens:
t = 2*pi*(x % element)/element
x0 = x0 + xmax*math.cos(t)
y0 = y0 + xmax*math.sin(t)
#print(str(x0)+' '+str(y0))
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
#else:
#plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
#if (x != 1):
#ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
x = x+1
plt.show()
sys.exit(0)
-:--- quand-cercles.py Bot L19 (Python)
Wrote /home/vella-chemla/Desktop/quand-cercles.py

Alignés par Frobenius ? (Denise Vella-Chemla, 3.10.2020)

Il s'agit ici de donner témoignage de propriétés dont on ne comprend pas la cause, qui sont peut-être triviales, mais qui interrogent.

Poursuivant les expérimentations informatiques du dernier mois, on continue de représenter les nombres par des nombres complexes particuliers qui sont leur image par certaines fonctions. Ici, on utilise des variations du programme Python suivant :

```
import math
from math import *
import matplotlib
from matplotlib import *
import matplotlib.pyplot as plt
from matplotlib.pyplot import *
import mpmath
from mpmath import *

def prime(atester):
  pastrouve = True ; k = 2 ;
  if (atester in [0,1]): return False ;
  if (atester in [2,3,5,7]): return True ;
  while (pastrouve):
    if ((k * k) > atester): return True
    else:
      if ((atester % k) == 0): return False
      else: k=k+1

fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')

x = 1 ; n = 100 ;
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x < n:
  x0, y0 = 0, 0
  t = mpmath.sqrt(mpmath.exp(2*pi*x*j/n))                    *****
  tprime=(1.0-(1.0/t)**50)/(1.0-(1.0/t))                     *****
  x0 = x0 + tprime.real
  y0 = y0 + tprime.imag
  #print(str(x0)+'   '+str(y0))
  if (prime(x)):
    plt.plot(x0, y0, 'r', marker='o', markersize=1)
    ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
  else:
    plt.plot(x0, y0, 'g', marker='o', markersize=1)
    ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
  #if (x != 1):
    #ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
  xprec = x0
  yprec = y0
  x = x+1

xmin, xmax, ymin, ymax = ax.axis()
print(xmin)
print(xmax)
print(ymin)
print(ymax)
ax.set_xlim(xmin-10, xmax+10) ;
ax.set_ylim(ymin-10, ymax+10)
plt.show()
sys.exit(0)
```

Comme on le voit dans les lignes de code marquées par cinq étoiles, il s'agit de calculer soit $t = e^{\frac{2i\pi\ x}{n}}$, soit sa racine complexe $t^{1/2}$, puis d'appliquer à t la fonction

$$f(t) = \sum_{k=1}^{v} \left(\frac{1}{t}\right)^k = \frac{1-\left(\frac{1}{t}\right)^{v+1}}{1-\left(\frac{1}{t}\right)}$$

et de "plotter" le complexe correspondant.

La source de l'incompréhension est la suivante : selon les valeurs choisies pour n et v, on parvient à faire s'aligner les points ou pas dans le plan complexe, mais on ne comprend vraiment pas pour quelle raison ils s'alignent ou pas.

Dans la mesure où il est très lourd de fournir tous les programmes et leurs exécutions, on fournit ci-dessous quelques exemples illustrant : [1]

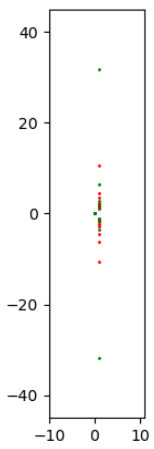


Fig. 1

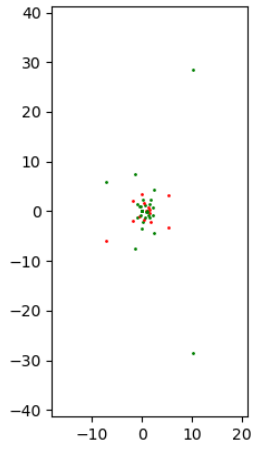


Fig. 2

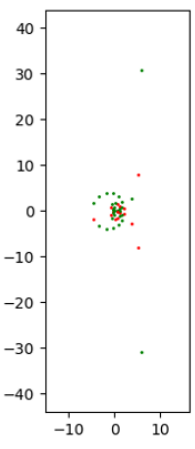


Fig. 3

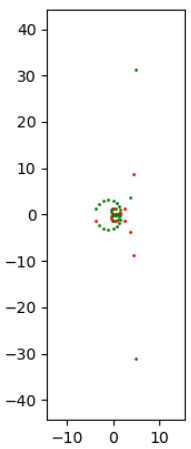


Fig. 4

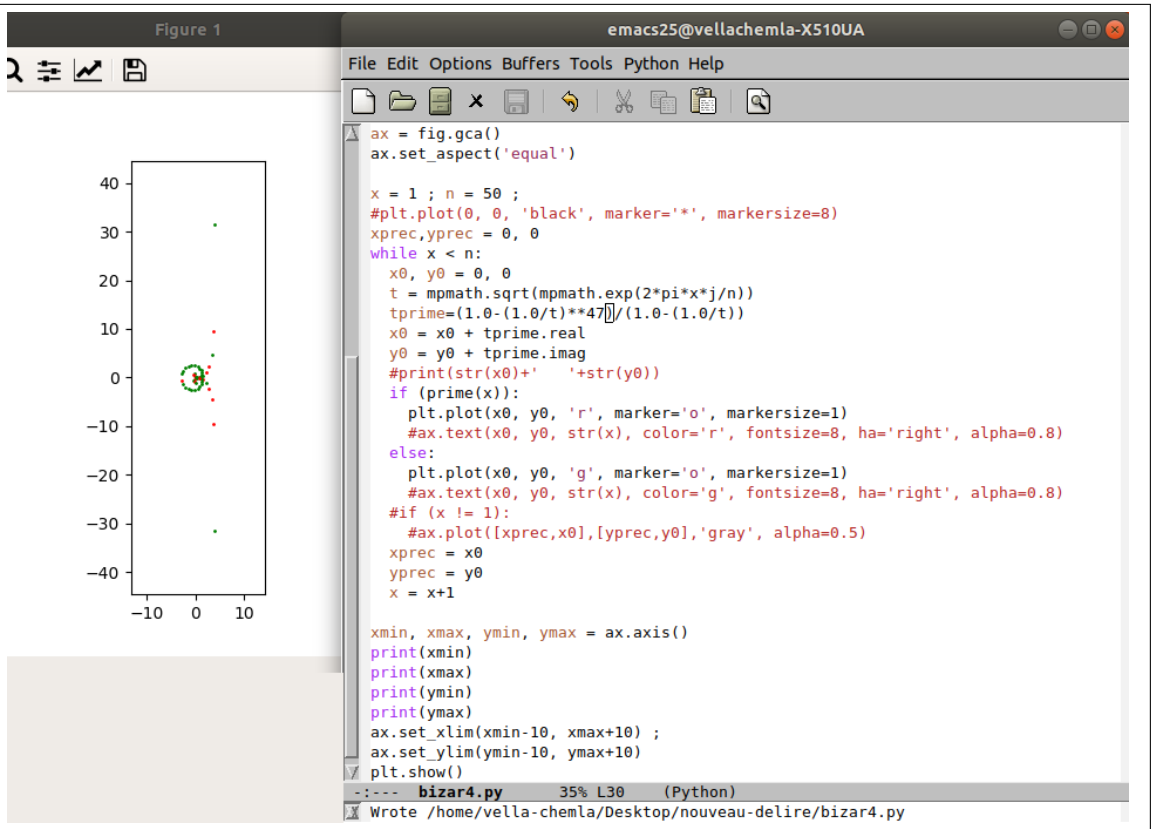


Fig. 5

1. On ne comprend également pas pourquoi les résultats des programmes Python ne sont pas identiques lorsqu'on écrit mpmath.sqrt(mpmath.exp(2*j*pi*x/n)) et mpmath.exp(j*pi*x/n) (*attention* : le complexe $i = \sqrt{-1}$ s'écrit j en Python.)

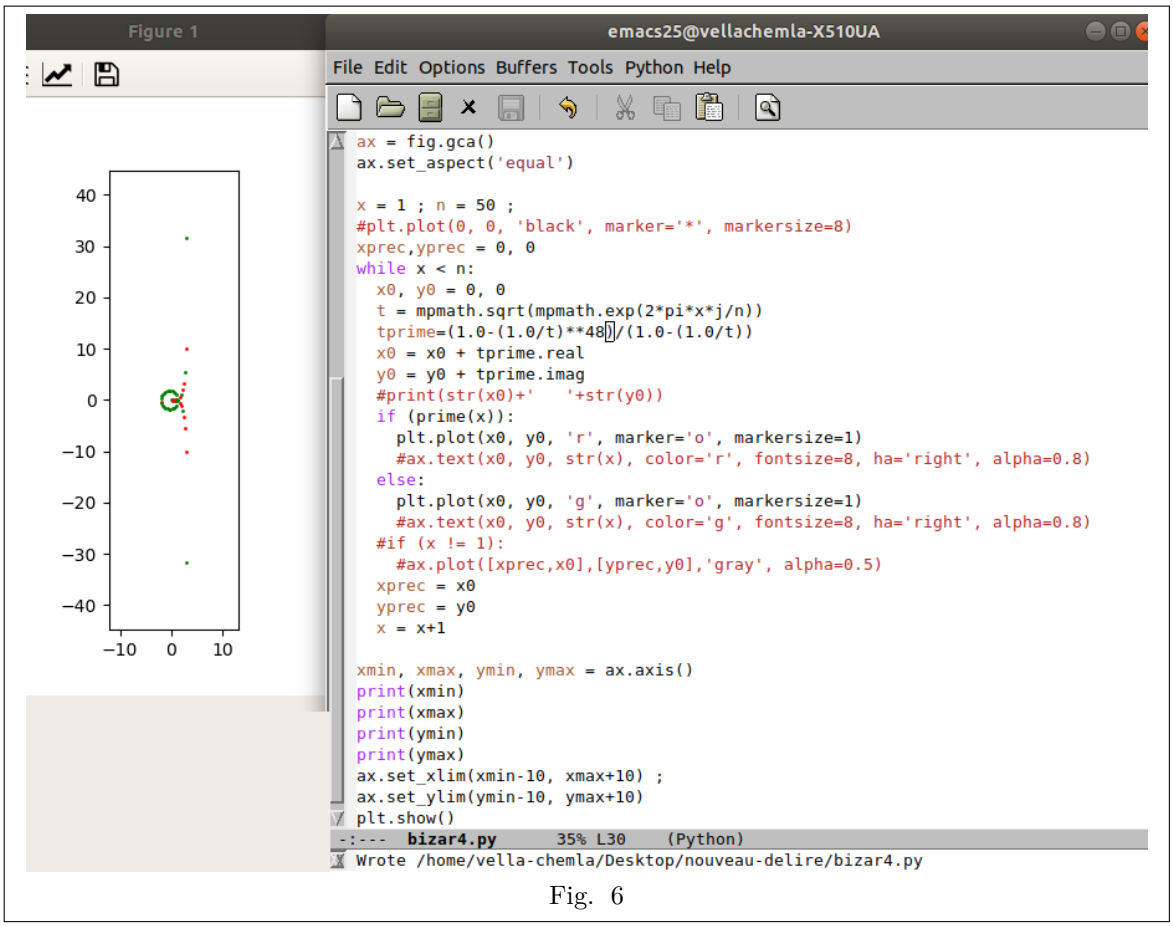


Fig. 6

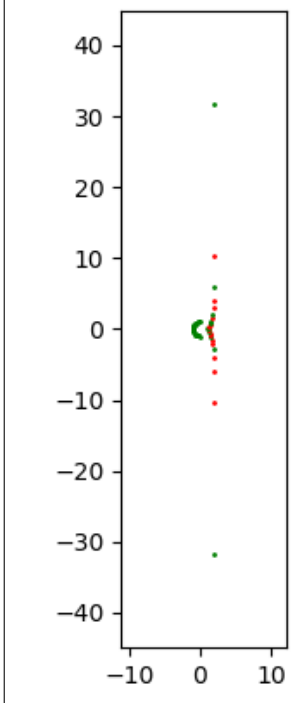


Fig. 7

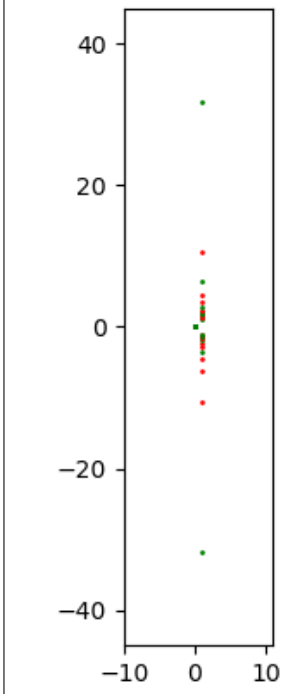


Fig. 8

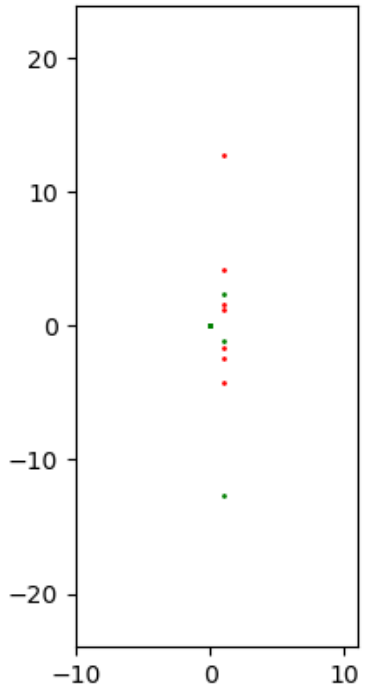


Fig. 9

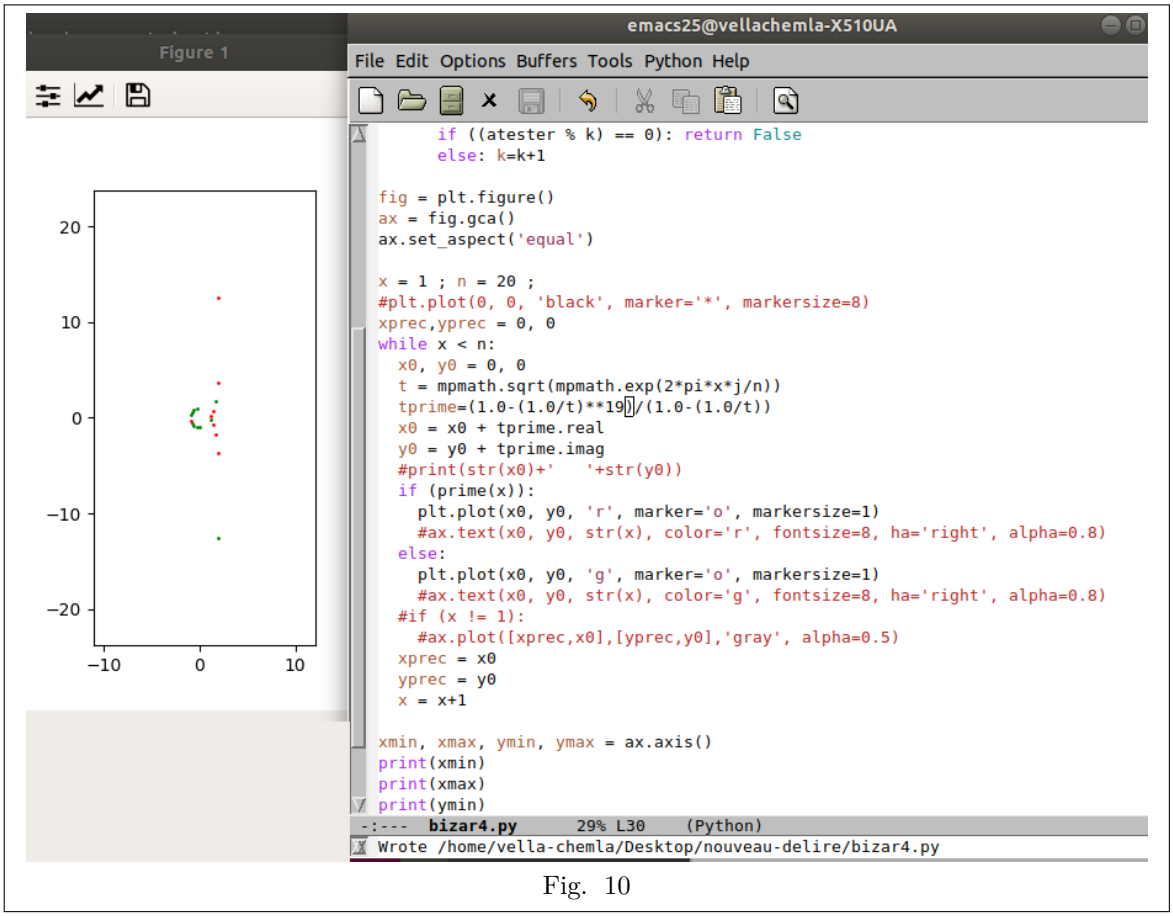


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
x = 1 ; n = 20 ;
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x < n:
x0, y0 = 0, 0
t = mpmath.sqrt(mpmath.exp(2*pi*x*j/n))
tprime=(1.0-(1.0/t)**19)/(1.0-(1.0/t))
x0 = x0 + tprime.real
y0 = y0 + tprime.imag
#print(str(x0)+' '+str(y0))
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
#if (x != 1):
#ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
x = x+1
xmin, xmax, ymin, ymax = ax.axis()
print(xmin)
print(xmax)
print(ymin)
-:--- bizar4.py 29% L30 (Python)
Wrote /home/vella-chemla/Desktop/nouveau-delire/bizar4.py
20
10
0
−10
−20
−10
0
10

Fig. 10

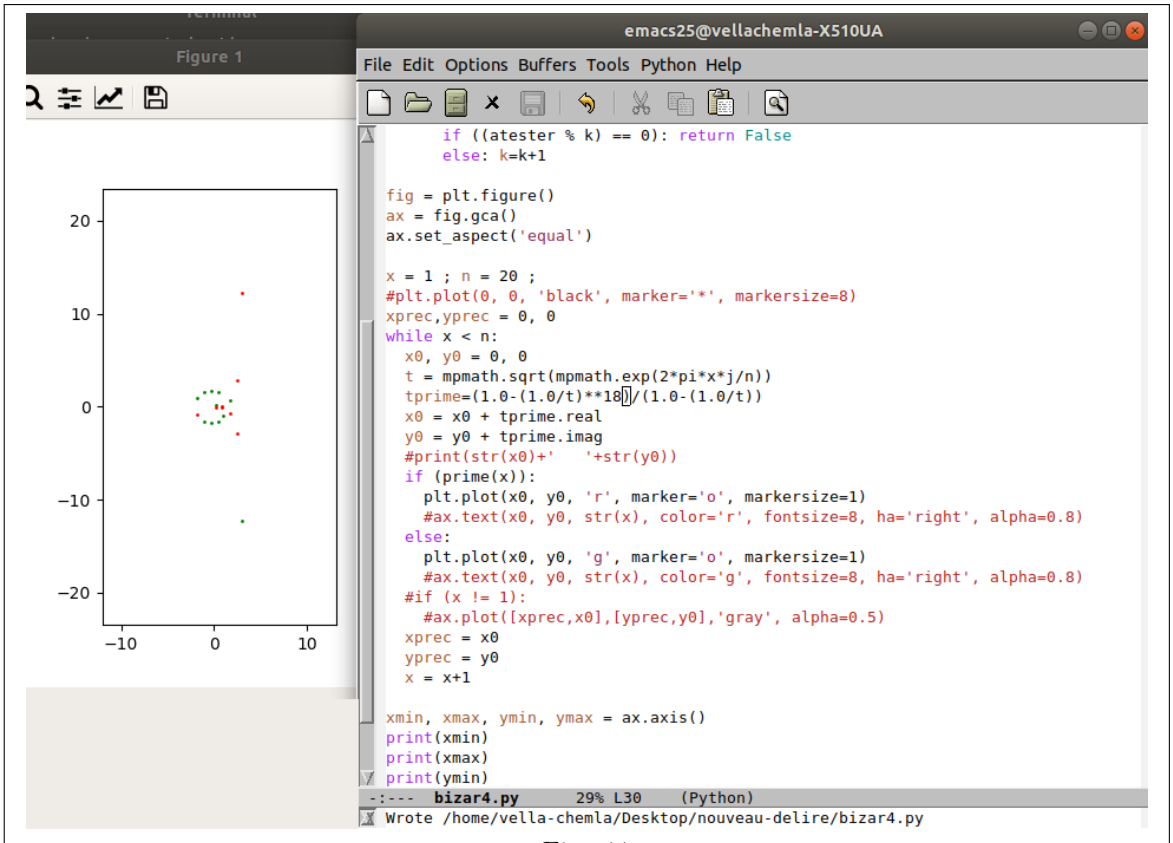


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
x = 1 ; n = 20 ;
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x < n:
x0, y0 = 0, 0
t = mpmath.sqrt(mpmath.exp(2*pi*x*j/n))
tprime=(1.0-(1.0/t)**18)/(1.0-(1.0/t))
x0 = x0 + tprime.real
y0 = y0 + tprime.imag
#print(str(x0)+' '+str(y0))
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
#if (x != 1):
#ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
x = x+1
xmin, xmax, ymin, ymax = ax.axis()
print(xmin)
print(xmax)
print(ymin)
-:--- bizar4.py 29% L30 (Python)
Wrote /home/vella-chemla/Desktop/nouveau-delire/bizar4.py
20
10
0
−10
−20
−10
0
10

Fig. 11

Ci-dessous, le même graphique est fourni que sur la Figure 11 mais en écrivant les nombres dont sont plottées les images.

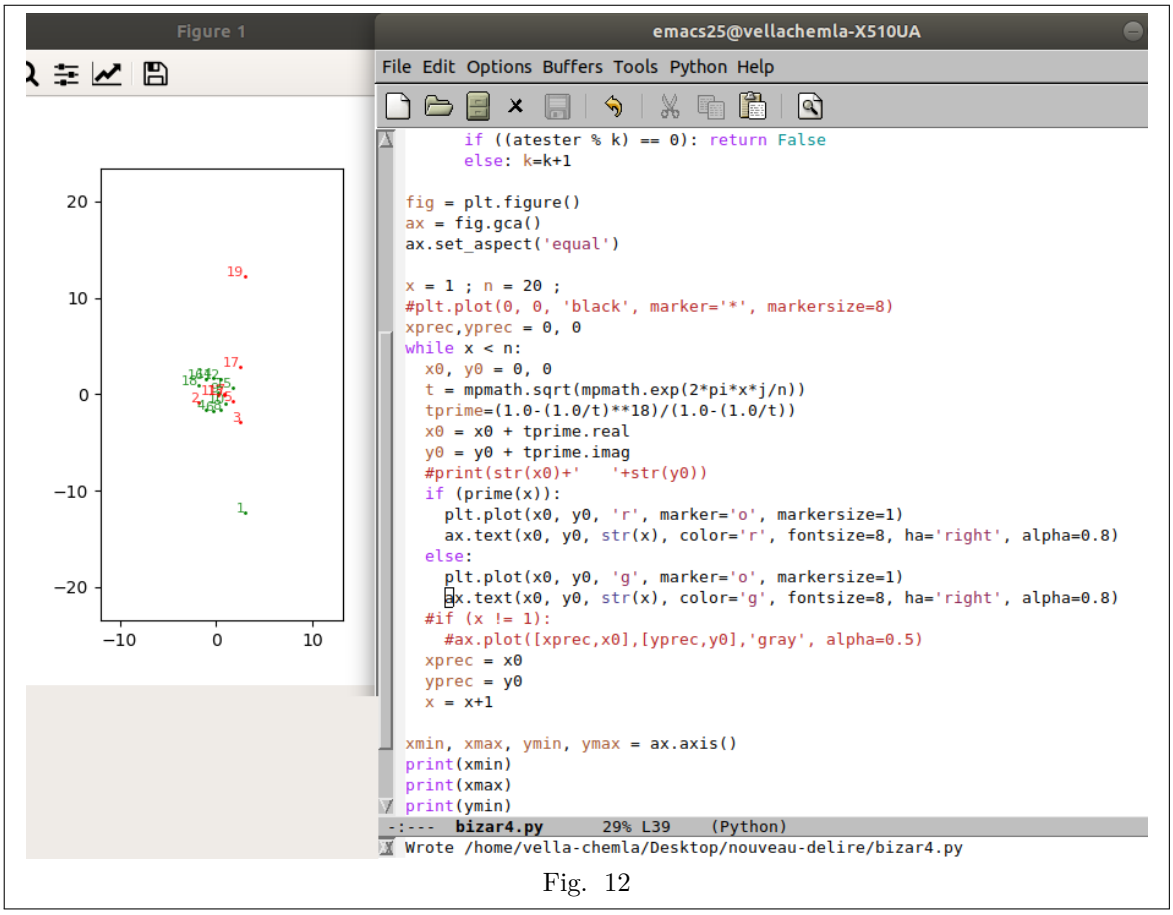


Fig. 12

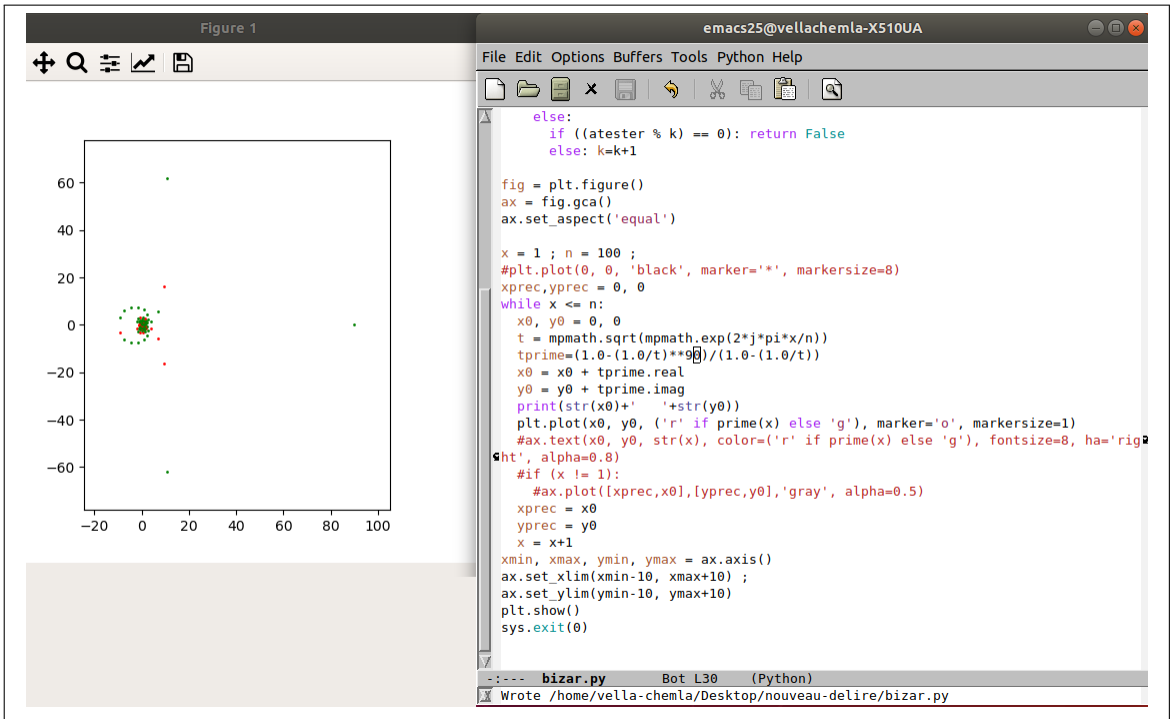


Fig. 13

Dans les figures 1, 8 et 9, pour lesquelles on constate l'alignement des nombres complexes images, ont systématiquement été sommées les puissances jusqu'à la n-ième au numérateur.

Ci-après, sur les Figures 14 et 15, en sommant les puissances jusqu'à la 50ème, pour x compris entre 1 et 100 (donc jusqu'à la $n/2$-ième puissance seulement, et non jusqu'à la n-ième comme précédemment), les nombres complexes images sont alignés sur deux droites, selon que ce sont les images de nombres premiers de la forme $4u+1$ ou $4u+3$ [2].

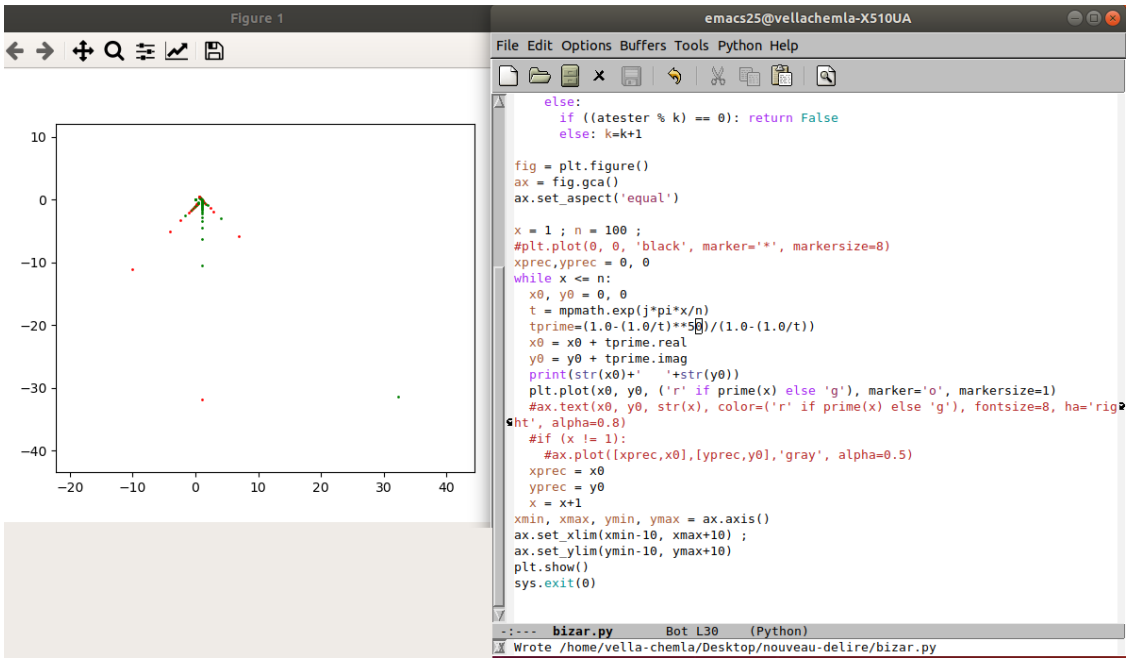


Fig. 14

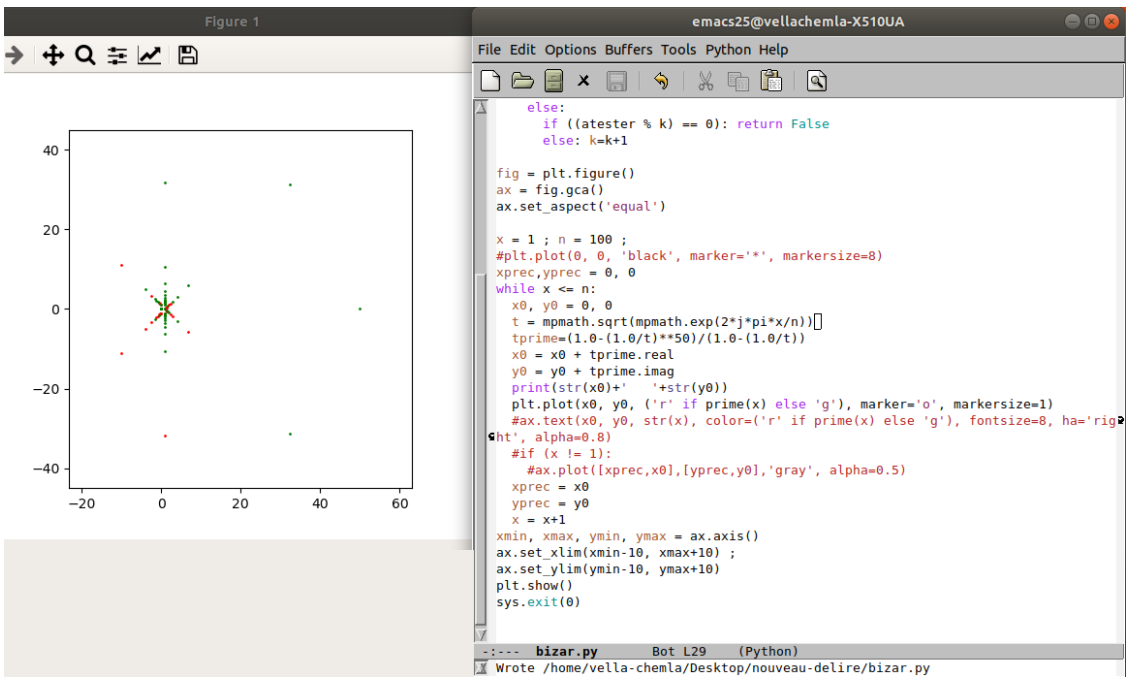


Fig. 15

2. C'est entre ces deux exécutions qu'on a cette incompréhension de la note de bas de page vue plus haut.

On aimerait peut-être relier nos découvertes à cette équation :

$$\left(\frac{1}{\sqrt{3}}+i\right)^3-\left(\frac{1}{\sqrt{3}}-i\right)^3=0$$

et à ses variantes faisant intervenir des puissances autres que des cubes.

Expliquer un alignement par opérateur de Frobenius (Denise Vella-Chemla, 4.10.2020)

Il s'agit de fournir l'explication d'une question qu'on se posait au sujet de l'alignement sur un axe vertical des images complexes qu'on a choisi d'associer aux nombres, en lien avec l'opérateur de Frobenius.

Poursuivant nos expérimentations informatiques dans le plan complexe, on représente les nombres par des nombres complexes particuliers qui sont leur image par certaines fonctions. Voici le programme qu'on utilise :

```
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *

def prime(atester):
  pastrouve = True ; k = 2 ;
  if (atester in [0,1]): return False ;
  if (atester in [2,3,5,7]): return True ;
  while (pastrouve):
    if ((k * k) > atester): return True
    else:
      if ((atester % k) == 0): return False
      else: k=k+1

fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')

n = 100
m = n
for k in range(1,n):
    t = exp(-j*math.pi*k/n)             *****
    z = (1.0 - t**m)/(1.0 - t)          *****
    if prime(k):
        c = 'r' if prime(k) else 'g'
        plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
        ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)

xmin, xmax, ymin, ymax = ax.axis()
ax.set_xlim(xmin-10, xmax+10) ;
ax.set_ylim(ymin-10, ymax+10)
plt.show()
```

Comme on le voit dans les lignes de code marquées de cinq étoiles, le programme calcule l'inverse de la racine de $t = e^{\frac{2i\pi x}{n}}$ et applique à ce nombre la fonction $f(t) = 1 + t + t^2 + t^3 + \ldots$.

L'inverse de la racine carrée s'obtient en opposant l'exposant [1], et en enlevant le 2 du $2i\pi \ldots$ [2].

Le programme calcule donc $t_k = e^{\frac{-i\pi k}{n}}$, puis $z_k = \dfrac{1 - t_k^m}{1 - t_k} = \displaystyle\sum_{l=1}^{l=m-1} t_k^l$ pour $k \in [1, n-1]$ et "plotte" le complexe correspondant.

On a constaté que lorsque $m = n$, les points s'alignent alors qu'ils ne sont pas alignés lorsque $m \neq n$. Les figures 1 et 2 montrent l'alignement lorsque la somme est calculée "jusqu'au bout". Les figures 3 et 4 montrent l'absence d'alignement quand on calcule les sommes de puissances sans aller jusqu'au dernier terme [3]. Sur la figure 3, les nombres jusqu'à 100 sont dessinés en n'allant que jusqu'à la 98-ième puissance dans le calcul de la somme des termes de z_k, tandis que sur la figure 4, ne sont sommés, toujours pour les 100 premiers nombres, que les termes jusqu'à la 47-ième puissance.

1. $\frac{1}{z} = z^{-1}$.
2. $\sqrt{z} = z^{\frac{1}{2}}$
3. Pour que les graphiques soient assez visuels, on a parfois omis les points très "excentrés".

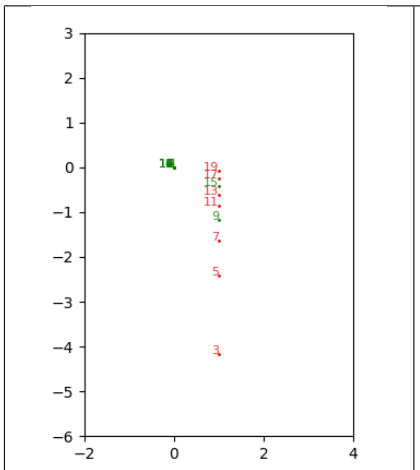


Fig. 1 : $x \leq 20$

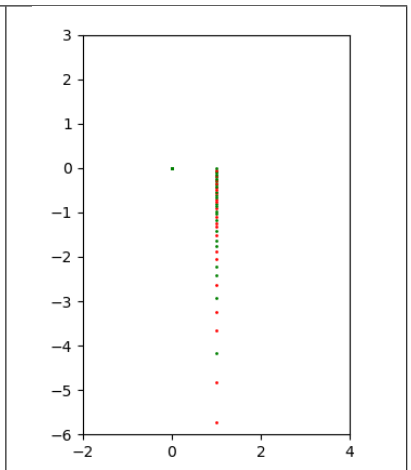


Fig. 2 : $x \leq 100$

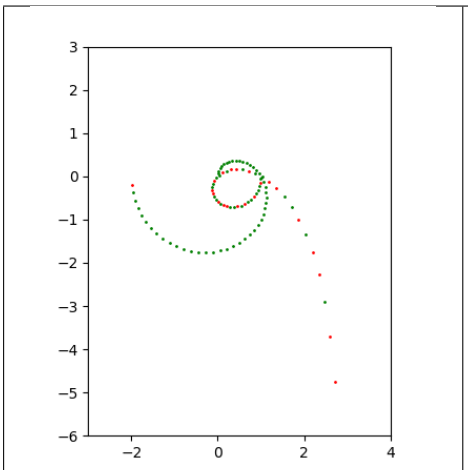


Fig. 3 : 100 nbs, exposants $\leq$ 98

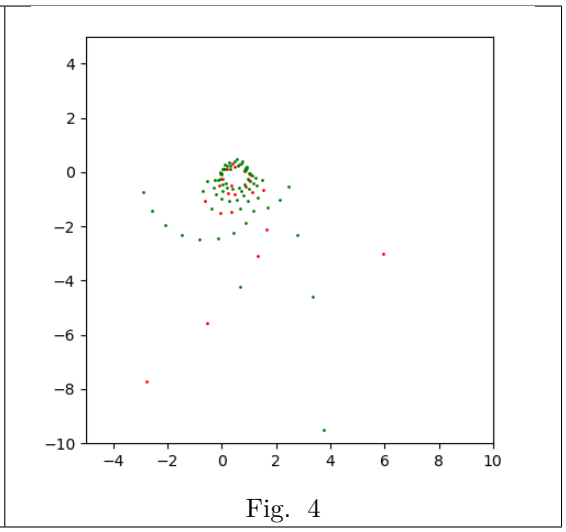


Fig. 4

L'explication de l'alignement de tous les nombres (si ce n'est un point qu'on semble distinguer comme non aligné avec les autres) sur la droite de partie réelle 1 est la suivante :

Puisque $t_k = e^{\frac{-i\pi k}{n}}$ et $z_k = \dfrac{1-t_k^m}{1-t_k}$, avec $k \in [1, n-1]$, si $m = n$, alors on a $t_k^m = e^{-i\pi k}$ et de ce fait,

$$\begin{aligned} z_k &= \frac{1-e^{-i\pi k}}{1-e^{\frac{-i\pi k}{n}}} \\ &= \frac{1-(-1)^k}{1-e^{\frac{-i\pi k}{n}}} \end{aligned}$$

Le dernier numérateur ci-dessus est nul lorsque k est pair.

Si k est impair, on a :

$$\begin{aligned} z_k &= \frac{2}{1-e^{\frac{-i\pi k}{n}}} \\ &= \frac{2\left(1-e^{\frac{i\pi k}{n}}\right)}{\left(1-\cos\ \frac{k\pi}{n}\right)^2+\left(\sin\ \frac{k\pi}{n}\right)^2} \end{aligned}$$

$$
\begin{aligned}
z_k &= \frac{2\left(1-e^{\frac{i\pi k}{n}}\right)}{2-2\cos\ \frac{k\pi}{n}} \\
&= \frac{1-e^{\frac{i\pi k}{n}}}{1-\cos\ \frac{k\pi}{n}} \\
&= \frac{1-\left(\cos\ \frac{k\pi}{n}+i\sin\ \frac{k\pi}{n}\right)}{1-\cos\ \frac{k\pi}{n}} \\
&= \frac{1-\cos\ \frac{k\pi}{n}}{1-\cos\ \frac{k\pi}{n}}+i\frac{\sin\ \frac{k\pi}{n}}{1-\cos\ \frac{k\pi}{n}}
\end{aligned}
$$

qui est de partie réelle constante égale à 1, d'où l'alignement des complexes obtenus comme images des nombres entiers.

La dernière figure ci-dessous montre les nombres pairs mal lisibles et tous collés sur 0. Pour ne pas écraser davantage les nombres les uns sur les autres, on a laissé de côté le nombre 1 qui est bien aligné aussi mais tout en bas à $-32i$.

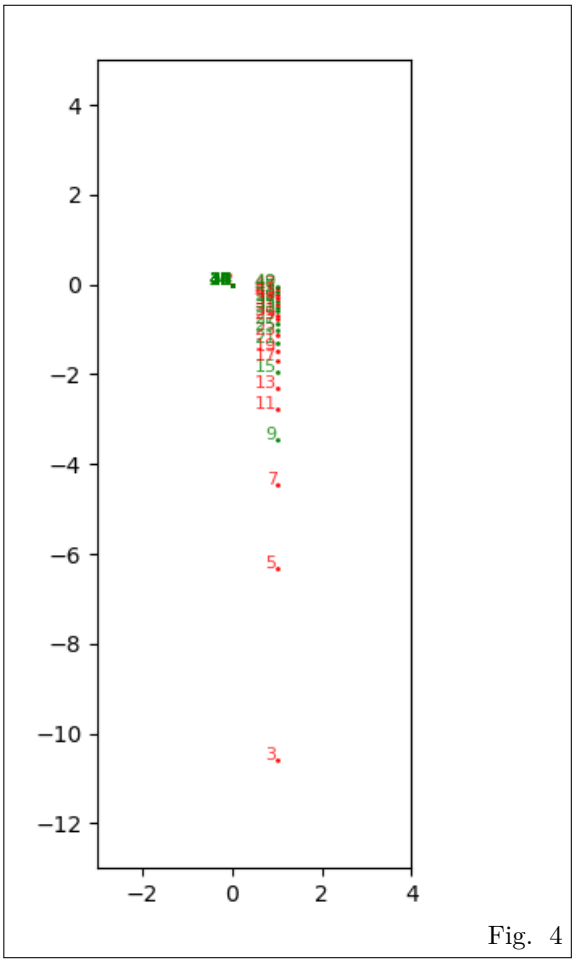
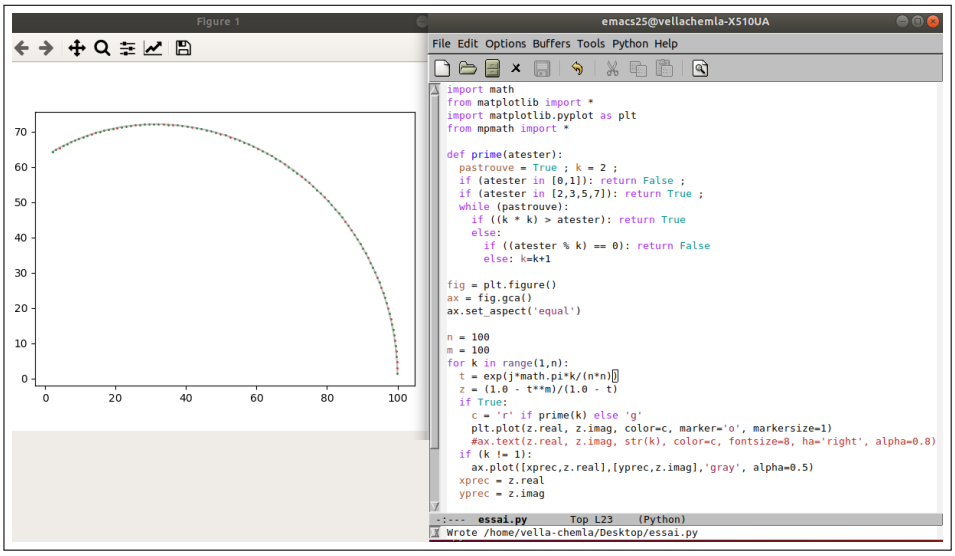
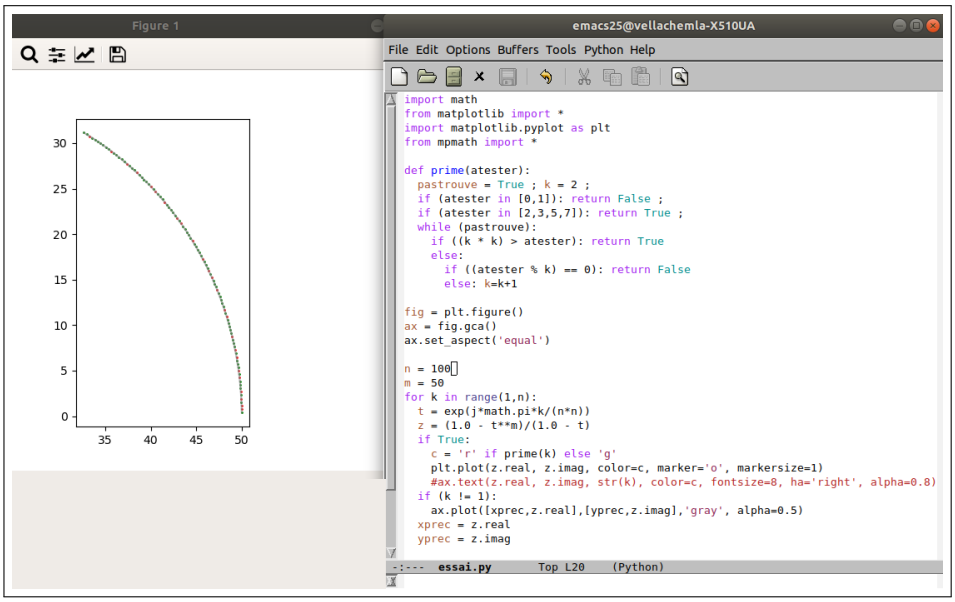


Fig. 4

D'autres dessins (Denise Vella-Chemla, 8.10.2020)





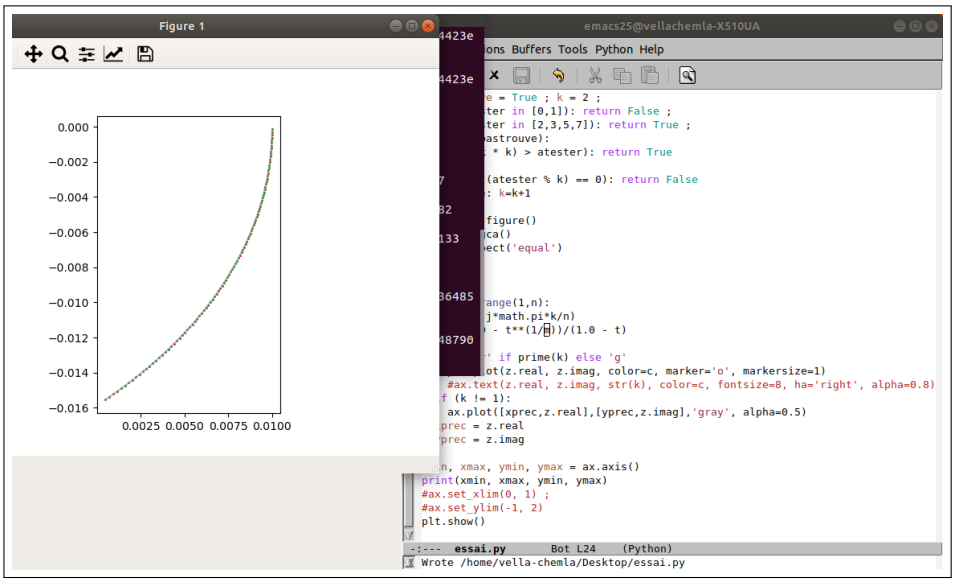


Figure 1
0.000
−0.002
−0.004
−0.006
−0.008
−0.010
−0.012
−0.014
−0.016
0.0025 0.0050 0.0075 0.0100
emacs25@vellachemla-X510UA
ons Buffers Tools Python Help
-:--- essai.py Bot L24 (Python)
Wrote /home/vella-chemla/Desktop/essai.py

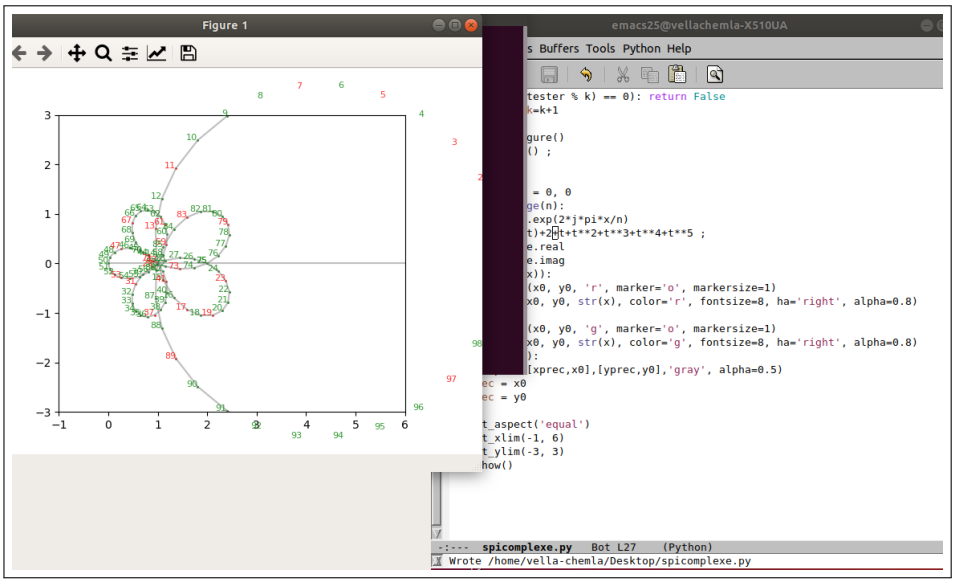


Figure 1
emacs25@vellachemla-X510UA
s Buffers Tools Python Help
-:--- spicomplexe.py Bot L27 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py

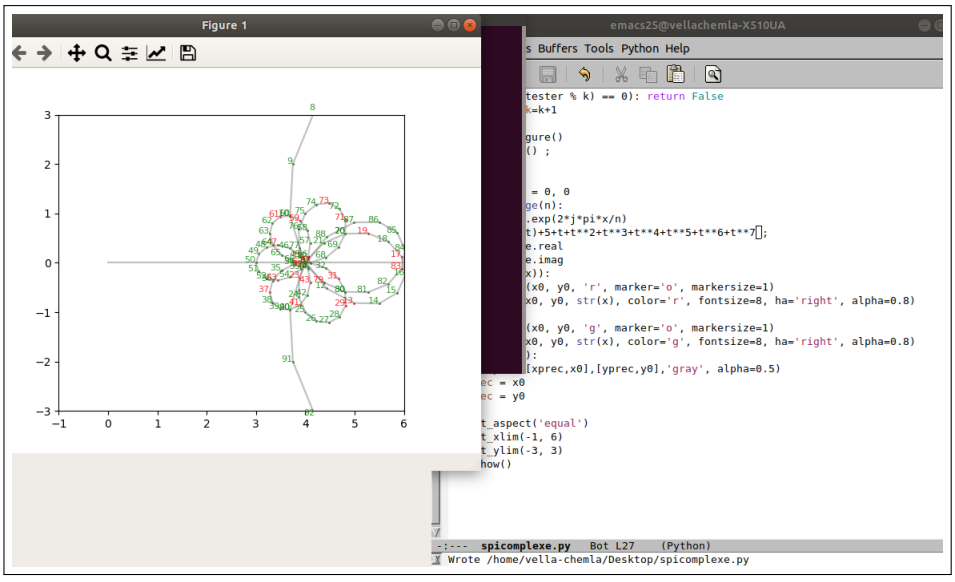


Figure 1
emacs25@vellachemla-X510UA
s Buffers Tools Python Help
-:--- spicomplexe.py Bot L27 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py

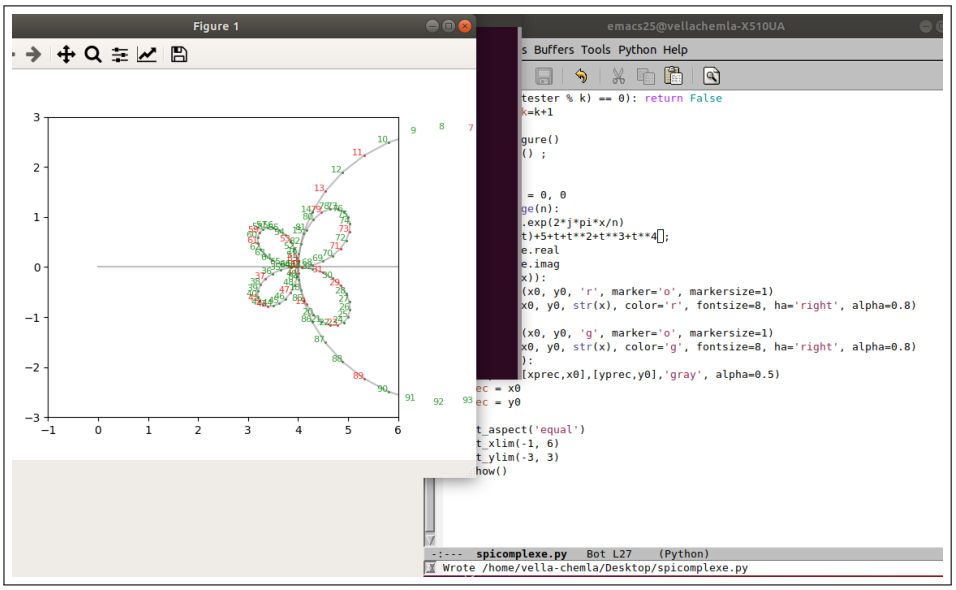


Figure 1
emacs25@vellachemla-X510UA
s Buffers Tools Python Help
tester % k) == 0): return False
k=k+1
gure()
() ;
= 0, 0
ge(n):
.exp(2*j*pi*x/n)
t)+5+t+t**2+t**3+t**4;
e.real
e.imag
x)):
(x0, y0, 'r', marker='o', markersize=1)
x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
(x0, y0, 'g', marker='o', markersize=1)
x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
):
[xprec,x0],[yprec,y0],'gray', alpha=0.5)
ec = x0
ec = y0
t_aspect('equal')
t_xlim(-1, 6)
t_ylim(-3, 3)
how()
-:--- spicomplexe.py Bot L27 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py

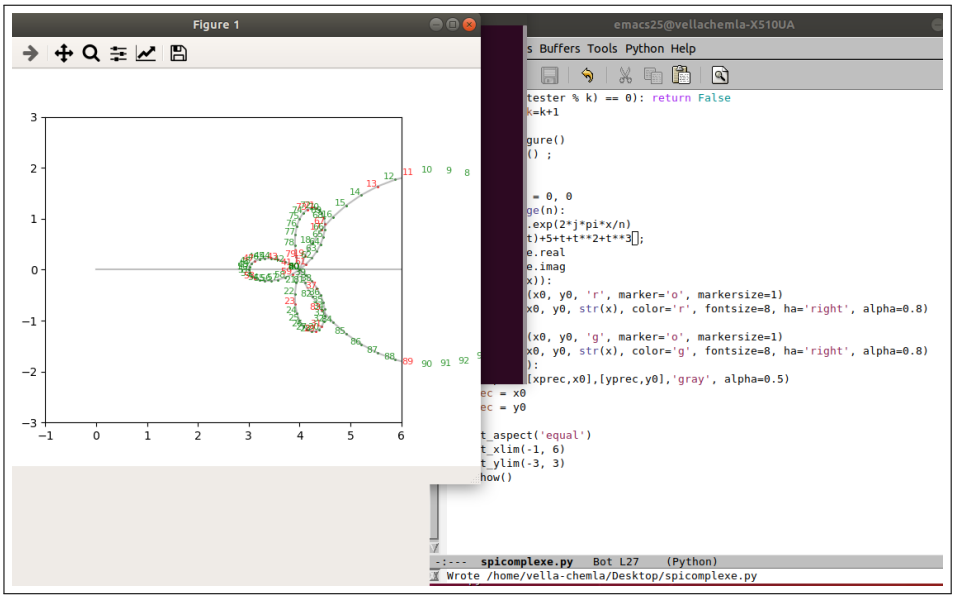


Figure 1
emacs25@vellachemla-X510UA
s Buffers Tools Python Help
tester % k) == 0): return False
k=k+1
gure()
() ;
= 0, 0
ge(n):
.exp(2*j*pi*x/n)
t)+5+t+t**2+t**3;
e.real
e.imag
x)):
(x0, y0, 'r', marker='o', markersize=1)
x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
(x0, y0, 'g', marker='o', markersize=1)
x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
):
[xprec,x0],[yprec,y0],'gray', alpha=0.5)
ec = x0
ec = y0
t_aspect('equal')
t_xlim(-1, 6)
t_ylim(-3, 3)
how()
-:--- spicomplexe.py Bot L27 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py

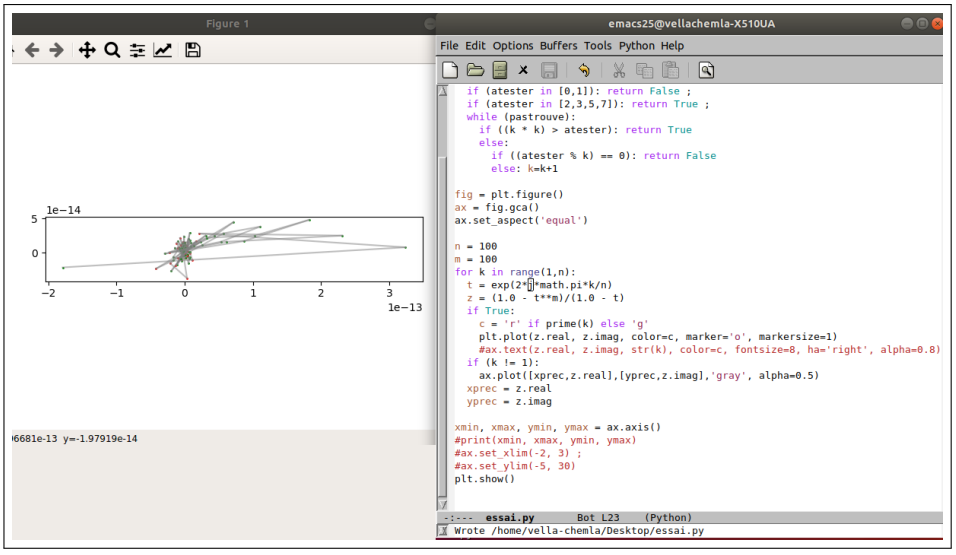


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,n):
t = exp(2*math.pi*k/n)
z = (1.0 - t**m)/(1.0 - t)
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
#ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
#print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-2, 3) ;
#ax.set_ylim(-5, 30)
plt.show()
1e−14
1e−13
-:--- essai.py Bot L23 (Python)
Wrote /home/vella-chemla/Desktop/essai.py

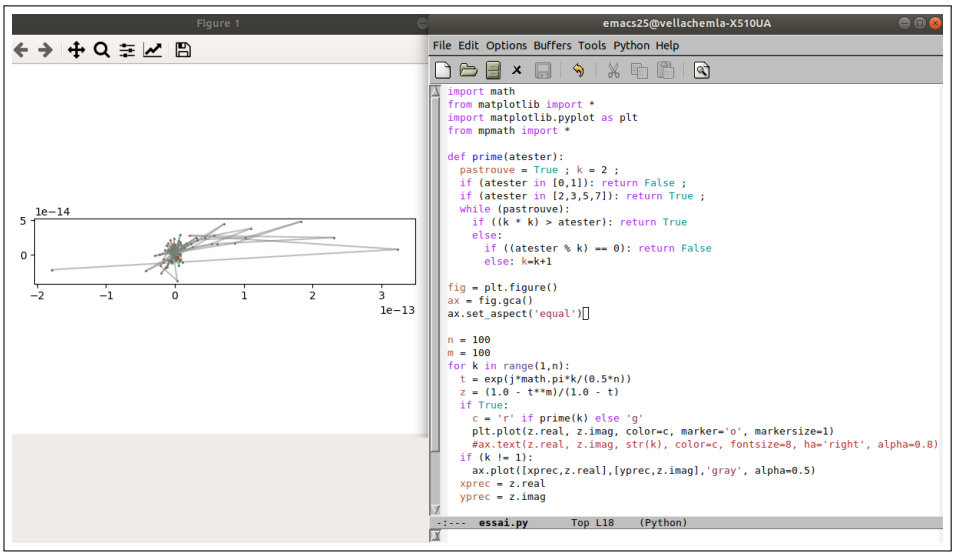


Figure 1
1e−14
5
0
−2
−1
0
1
2
3
1e−13
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,n):
t = exp(j*math.pi*k/(0.5*n))
z = (1.0 - t**m)/(1.0 - t)
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
#ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
-:--- essai.py Top L18 (Python)

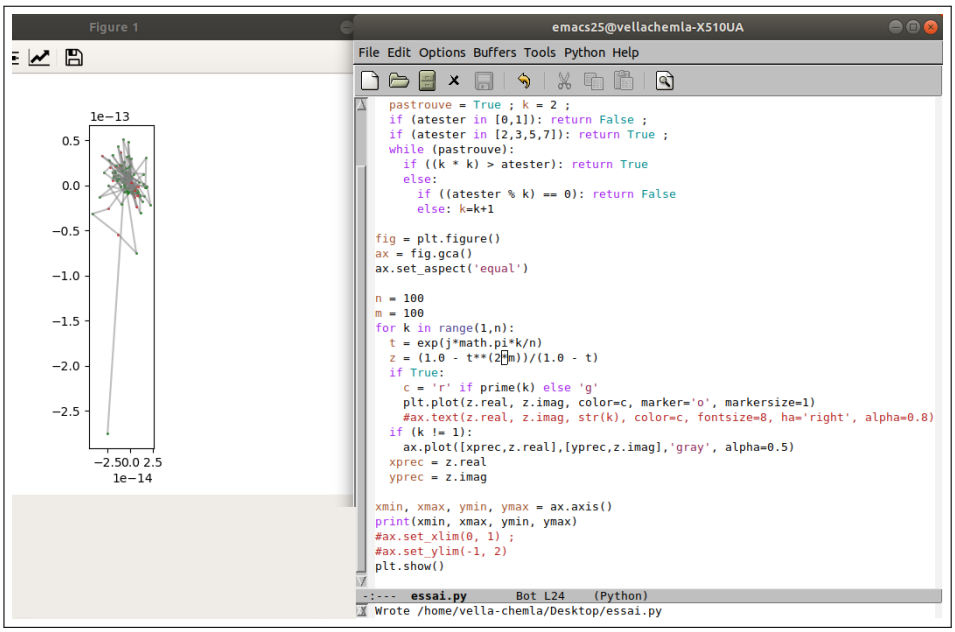


Figure 1
1e−13
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−2.5 0.0 2.5
1e−14
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,n):
t = exp(j*math.pi*k/n)
z = (1.0 - t**(2*m))/(1.0 - t)
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
#ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(0, 1) ;
#ax.set_ylim(-1, 2)
plt.show()
-:--- essai.py Bot L24 (Python)
Wrote /home/vella-chemla/Desktop/essai.py

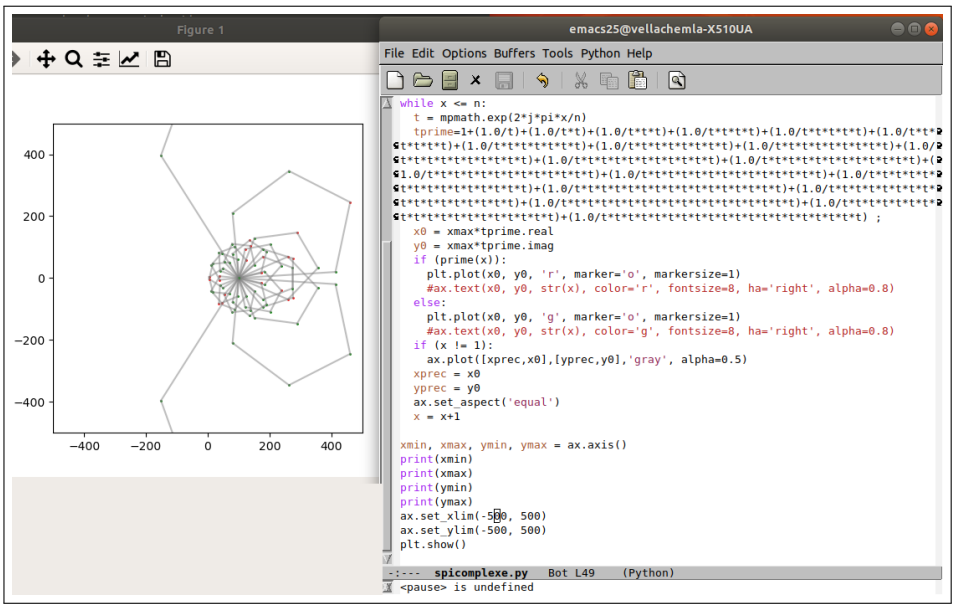
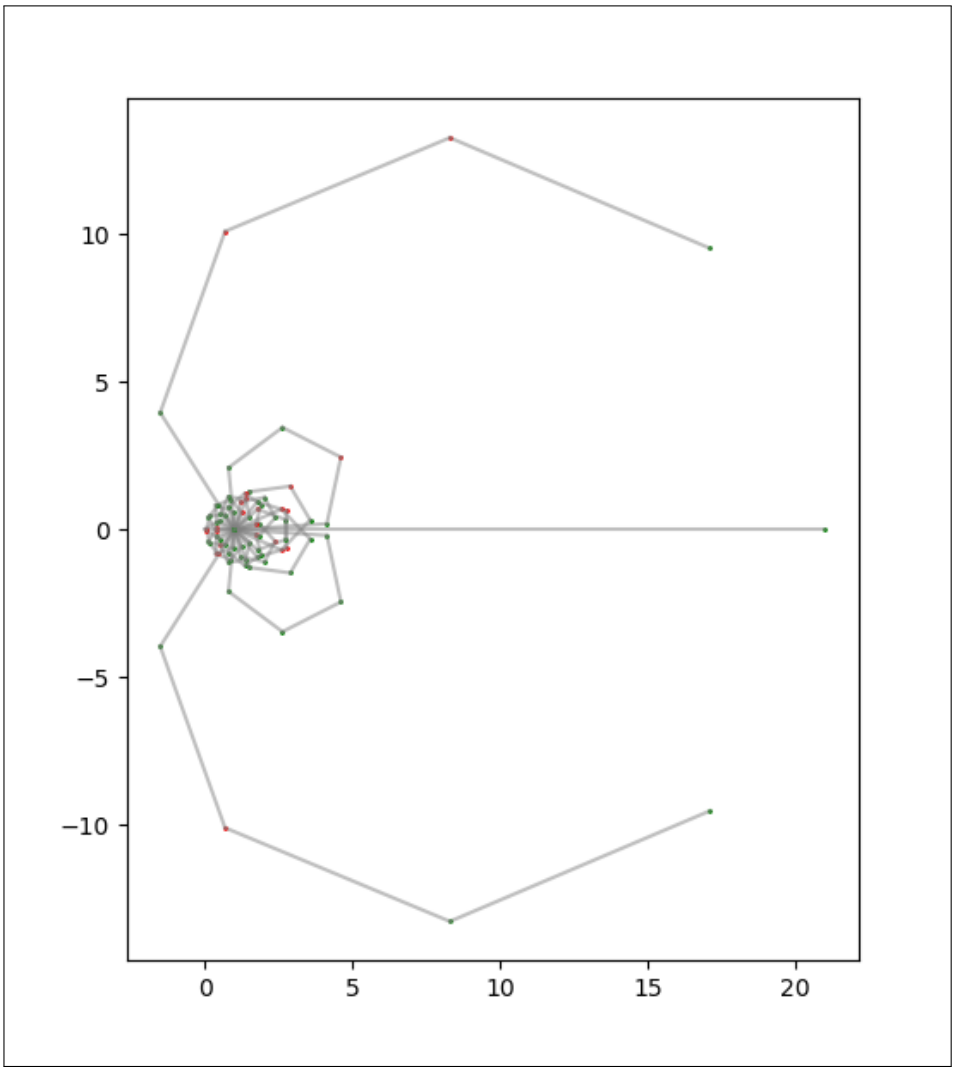


Figure 1
400
200
0
−200
−400
−400 −200 0 200 400
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
while x <= n:
t = mpmath.exp(2*j*pi*x/n)
tprime=1+(1.0/t)+(1.0/t*t)+(1.0/t*t*t)+(1.0/t*t*t*t)+(1.0/t*t*t*t*t)+(1.0/t*t*
$t*t*t*t)+(1.0/t*t*t*t*t*t*t)+(1.0/t*t*t*t*t*t*t*t)+(1.0/t*t*t*t*t*t*t*t*t)+(1.0/
$t*t*t*t*t*t*t*t*t)+(1.0/t*t*t*t*t*t*t*t*t*t)+(1.0/t*t*t*t*t*t*t*t*t*t*t)+(
$1.0/t*t*t*t*t*t*t*t*t*t*t)+(1.0/t*t*t*t*t*t*t*t*t*t*t*t)+(1.0/t*t*t*t*t*t*t*t*
$t*t*t*t*t*t*t*t*t)+(1.0/t*t*t*t*t*t*t*t*t*t*t*t*t)+(1.0/t*t*t*t*t*t*t*t*t*t*t*
$t*t*t*t*t*t*t*t*t)+(1.0/t*t*t*t*t*t*t*t*t*t*t*t*t*t)+(1.0/t*t*t*t*t*t*t*t*t*t*t*
$t*t*t*t*t*t*t*t*t*t*t)+(1.0/t*t*t*t*t*t*t*t*t*t*t*t*t*t*t) ;
x0 = xmax*tprime.real
y0 = xmax*tprime.imag
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
if (x != 1):
ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
x = x+1
xmin, xmax, ymin, ymax = ax.axis()
print(xmin)
print(xmax)
print(ymin)
print(ymax)
ax.set_xlim(-500, 500)
ax.set_ylim(-500, 500)
plt.show()
-:--- spicomplexe.py Bot L49 (Python)
<pause> is undefined


10
5
0
−5
−10
0
5
10
15
20

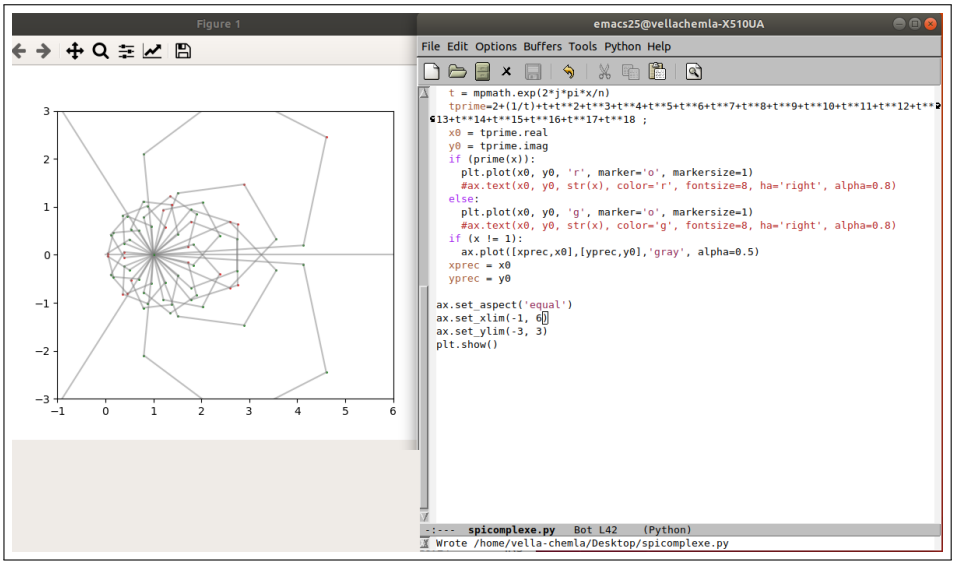
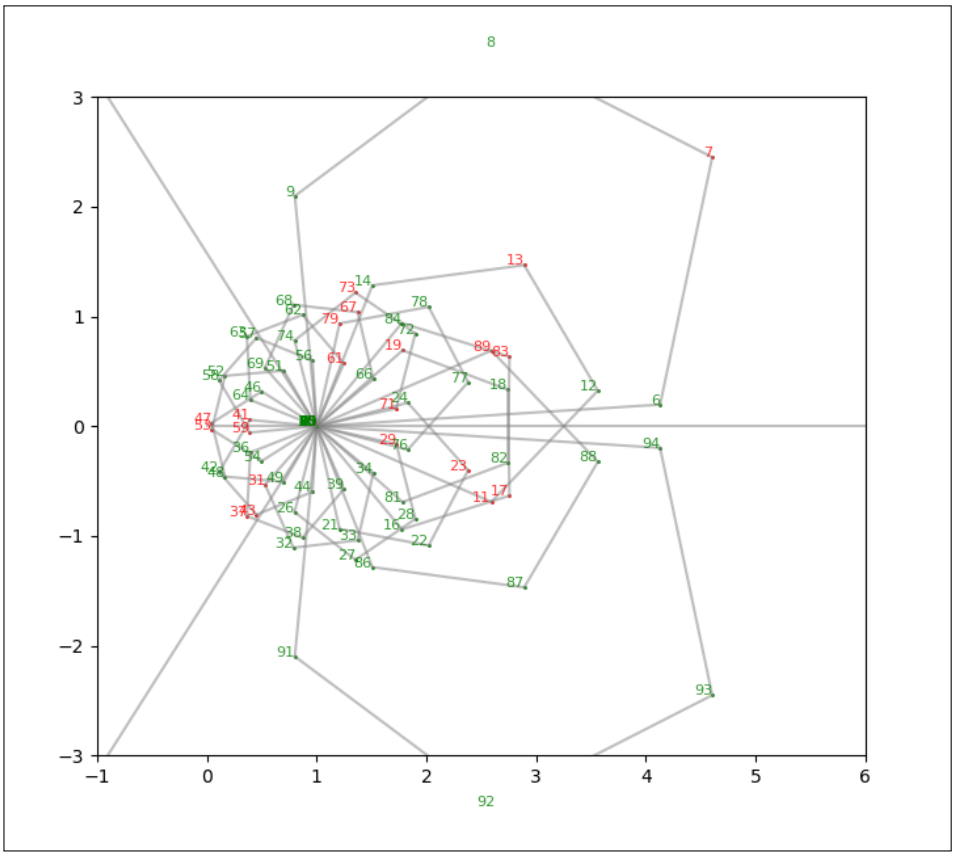


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
t = mpmath.exp(2*j*pi*x/n)
tprime=2+(1/t)+t+t**2+t**3+t**4+t**5+t**6+t**7+t**8+t**9+t**10+t**11+t**12+t**13+t**14+t**15+t**16+t**17+t**18 ;
x0 = tprime.real
y0 = tprime.imag
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
if (x != 1):
ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
ax.set_xlim(-1, 6)
ax.set_ylim(-3, 3)
plt.show()
-:--- spicomplexe.py Bot L42 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py


8
7
9
13
14
73
68
62
67
78
79
84
72
65
57
74
19
89
83
56
61
69
51
66
77
18
12
6
52
46
64
24
71
47
41
59
29
76
94
36
34
82
88
42
31
49
44
39
34
23
81
17
11
37
26
28
21
16
38
33
22
32
27
86
87
91
93
92

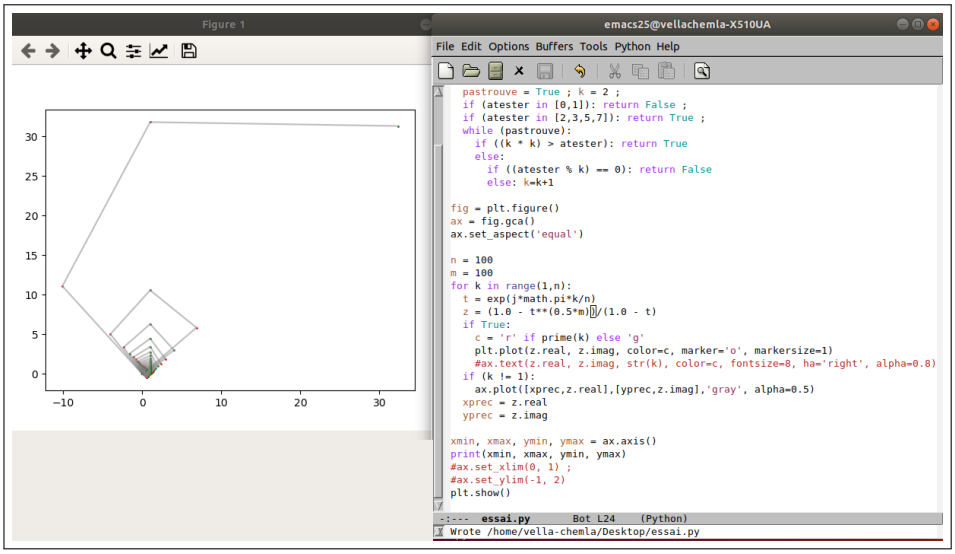


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,n):
t = exp(j*math.pi*k/n)
z = (1.0 - t**(0.5*m))/(1.0 - t)
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
#ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(0, 1) ;
#ax.set_ylim(-1, 2)
plt.show()
-:--- essai.py Bot L24 (Python)
Wrote /home/vella-chemla/Desktop/essai.py

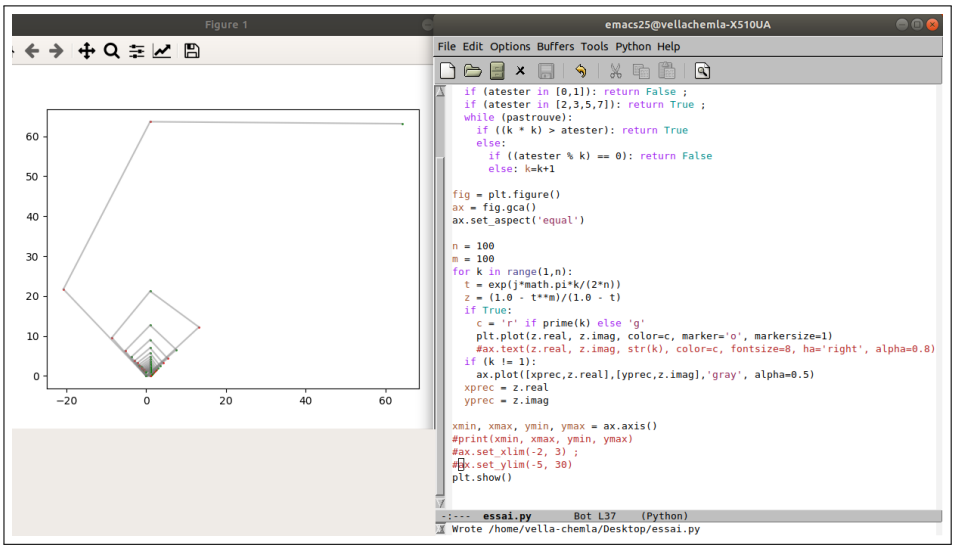


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,n):
t = exp(j*math.pi*k/(2*n))
z = (1.0 - t**m)/(1.0 - t)
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
#ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
#print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-2, 3) ;
ax.set_ylim(-5, 30)
plt.show()
-:---- essai.py Bot L37 (Python)
Wrote /home/vella-chemla/Desktop/essai.py

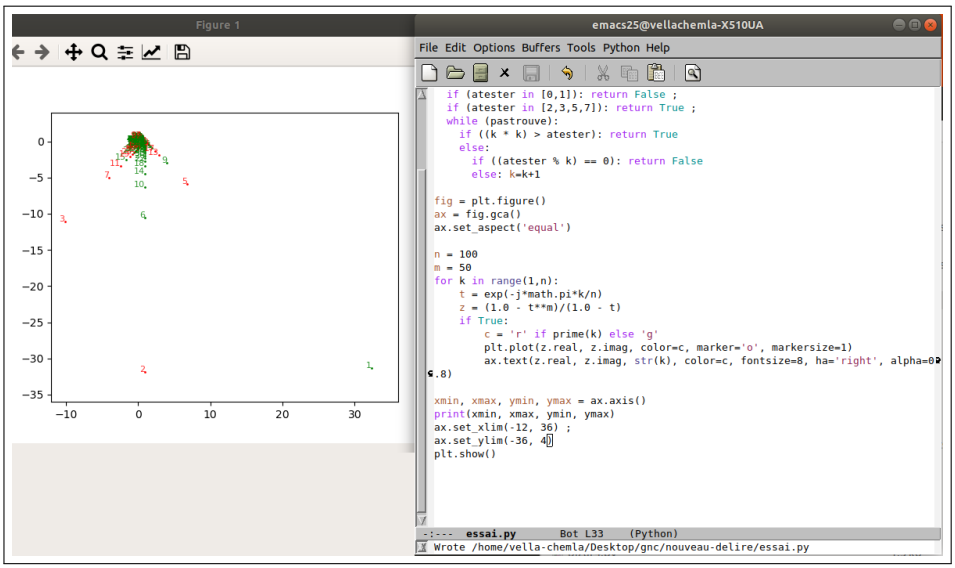


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 50
for k in range(1,n):
t = exp(-j*math.pi*k/n)
z = (1.0 - t**m)/(1.0 - t)
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
ax.set_xlim(-12, 36) ;
ax.set_ylim(-36, 4)
plt.show()
-:---- essai.py Bot L33 (Python)
Wrote /home/vella-chemla/Desktop/gnc/nouveau-delire/essai.py

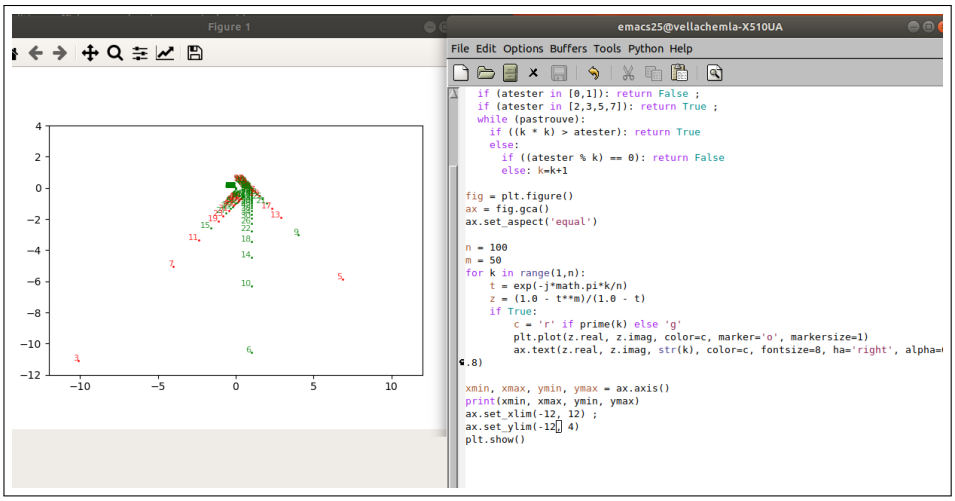


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 50
for k in range(1,n):
t = exp(-j*math.pi*k/n)
z = (1.0 - t**m)/(1.0 - t)
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
ax.set_xlim(-12, 12) ;
ax.set_ylim(-12, 4)
plt.show()

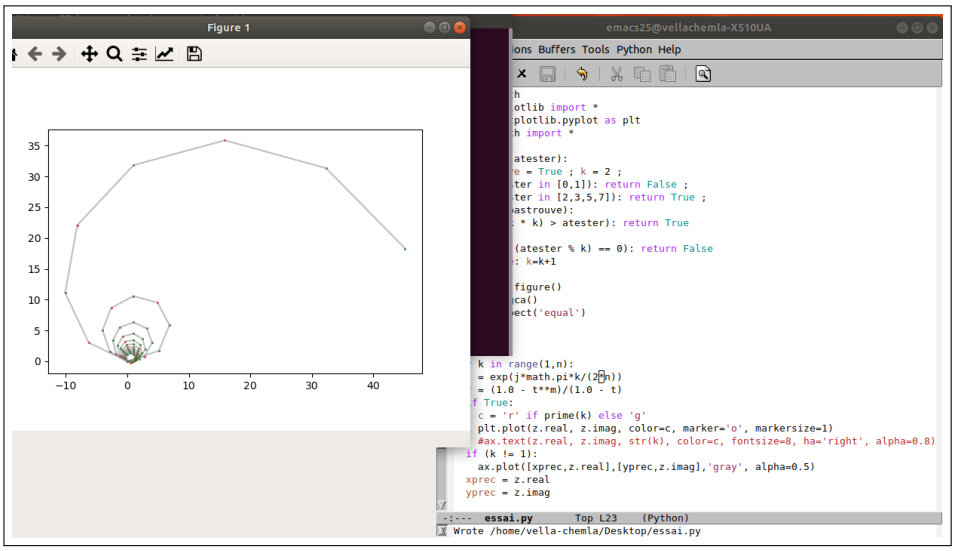


Figure 1
emacs25@vellachemla-X510UA
ons Buffers Tools Python Help
otlib import *
plotlib.pyplot as plt
h import *
atester):
e = True ; k = 2 ;
ter in [0,1]): return False ;
ter in [2,3,5,7]): return True ;
astrouve):
* k) > atester): return True
(atester % k) == 0): return False
: k=k+1
figure()
ca()
ect('equal')
k in range(1,n):
= exp(j*math.pi*k/(2
= (1.0 - t**m)/(1.0 - t)
f True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
#ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
-:--- essai.py Top L23 (Python)
Wrote /home/vella-chemla/Desktop/essai.py

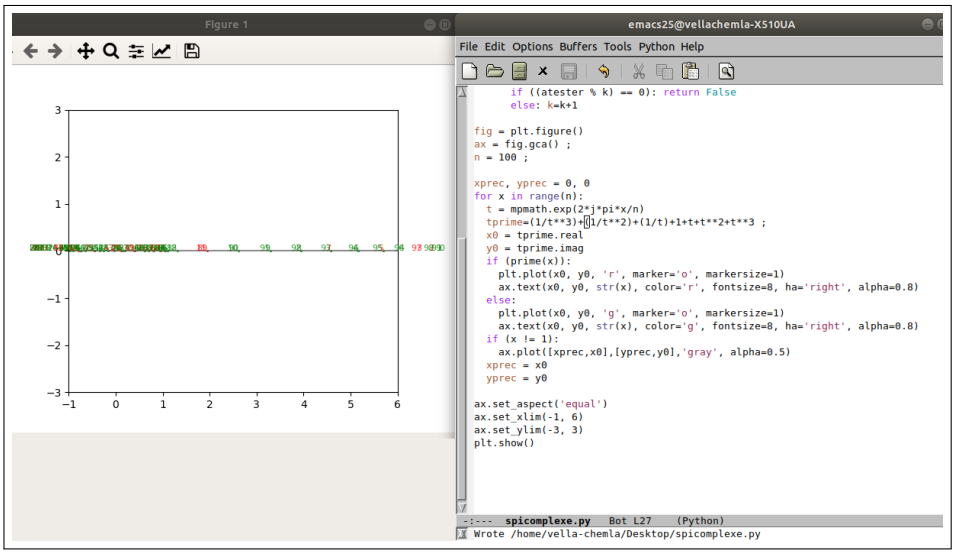


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca() ;
n = 100 ;
xprec, yprec = 0, 0
for x in range(n):
t = mpmath.exp(2*j*pi*x/n)
tprime=(1/t**3)+(1/t**2)+(1/t)+1+t+t**2+t**3 ;
x0 = tprime.real
y0 = tprime.imag
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
if (x != 1):
ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
ax.set_xlim(-1, 6)
ax.set_ylim(-3, 3)
plt.show()
-:--- spicomplexe.py Bot L27 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py

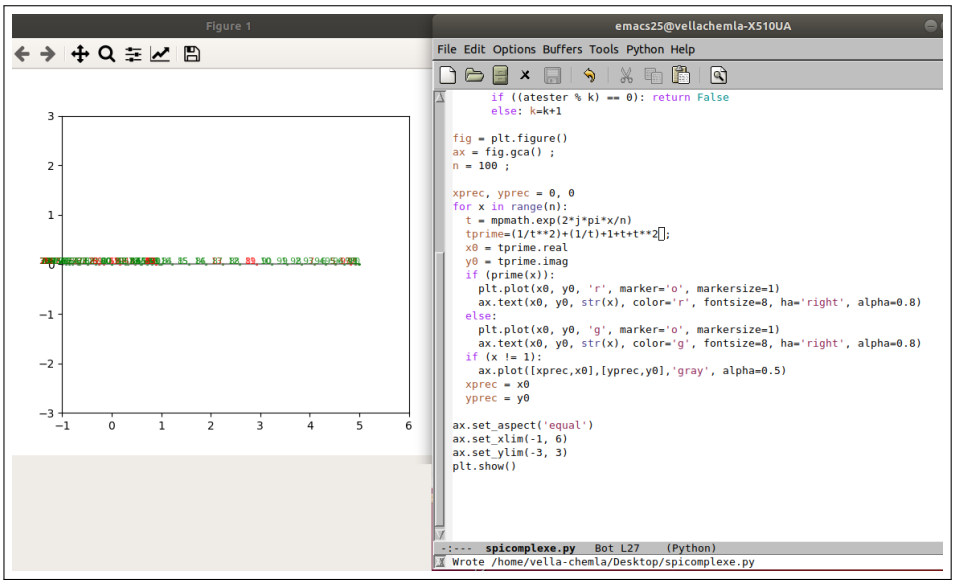


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca() ;
n = 100 ;
xprec, yprec = 0, 0
for x in range(n):
t = mpmath.exp(2*j*pi*x/n)
tprime=(1/t**2)+(1/t)+1+t+t**2;
x0 = tprime.real
y0 = tprime.imag
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
if (x != 1):
ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
ax.set_xlim(-1, 6)
ax.set_ylim(-3, 3)
plt.show()
-:--- spicomplexe.py Bot L27 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py

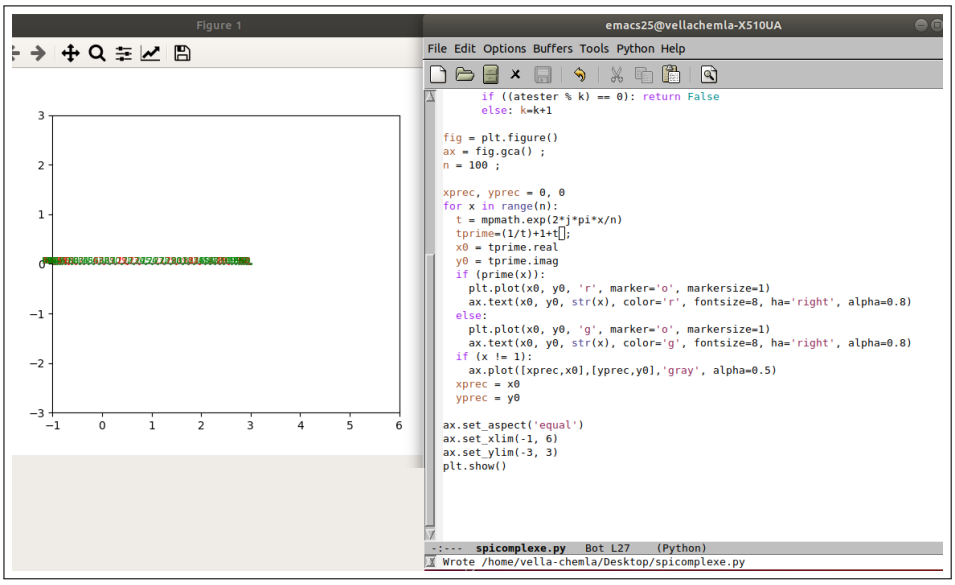


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca() ;
n = 100 ;
xprec, yprec = 0, 0
for x in range(n):
t = mpmath.exp(2*j*pi*x/n)
tprime=(1/t)+1+t;
x0 = tprime.real
y0 = tprime.imag
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
if (x != 1):
ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
ax.set_xlim(-1, 6)
ax.set_ylim(-3, 3)
plt.show()
-:--- spicomplexe.py Bot L27 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py

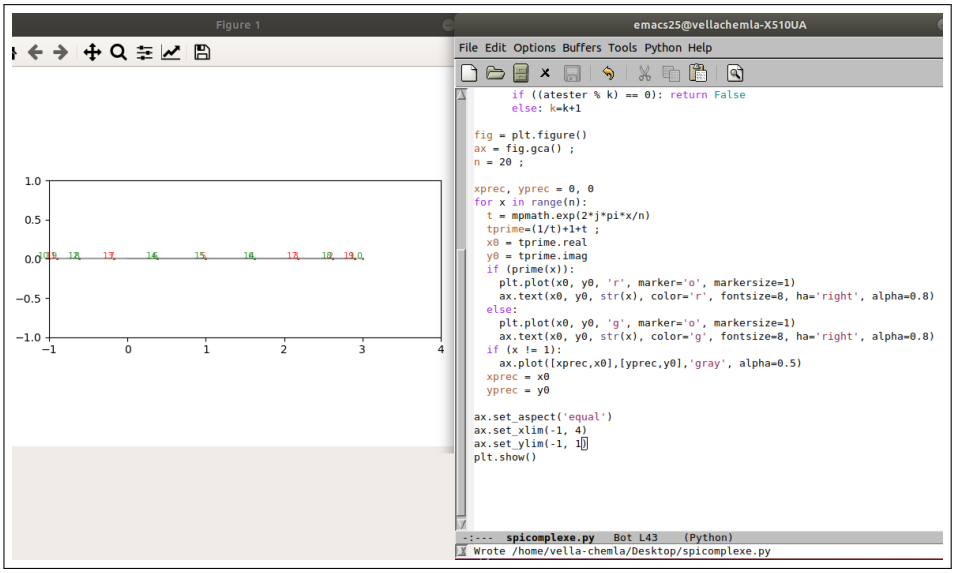


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca() ;
n = 20 ;
xprec, yprec = 0, 0
for x in range(n):
t = mpmath.exp(2*j*pi*x/n)
tprime=(1/t)+1+t ;
x0 = tprime.real
y0 = tprime.imag
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
if (x != 1):
ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
ax.set_xlim(-1, 4)
ax.set_ylim(-1, 1)
plt.show()
-:--- spicomplexe.py Bot L43 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py

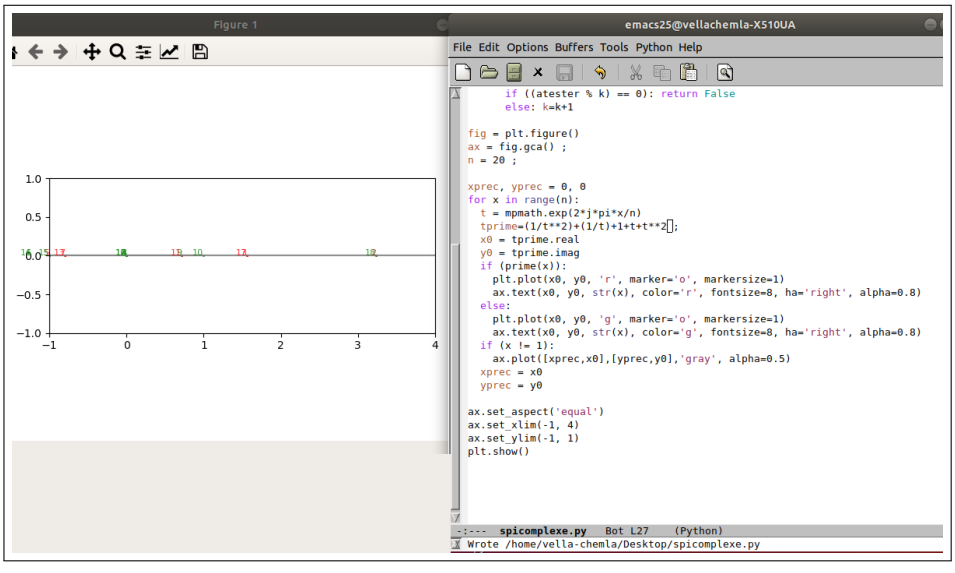


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca() ;
n = 20 ;
xprec, yprec = 0, 0
for x in range(n):
t = mpmath.exp(2*j*pi*x/n)
tprime=(1/t**2)+(1/t)+1+t+t**2;
x0 = tprime.real
y0 = tprime.imag
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
if (x != 1):
ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
ax.set_xlim(-1, 4)
ax.set_ylim(-1, 1)
plt.show()
-:--- spicomplexe.py Bot L27 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py

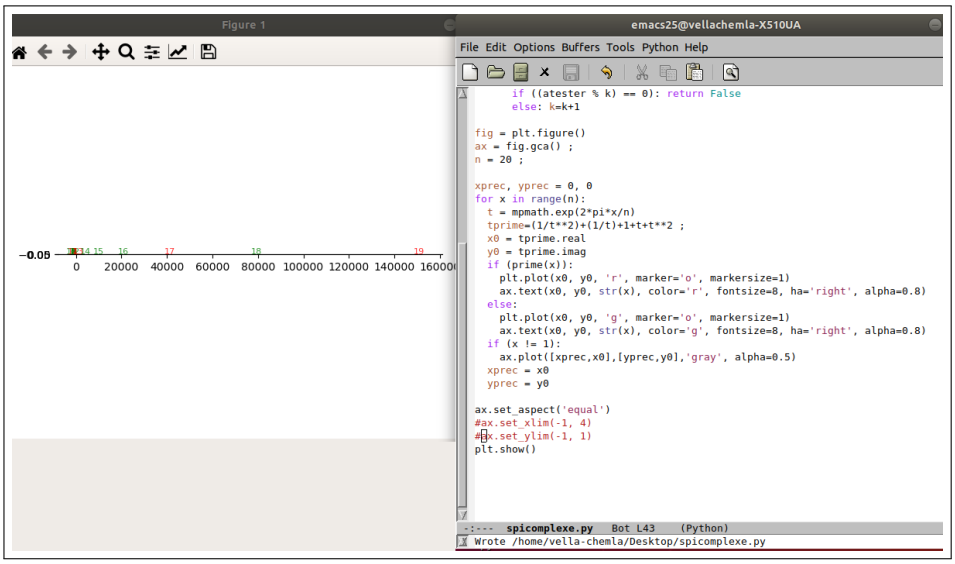


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca() ;
n = 20 ;
xprec, yprec = 0, 0
for x in range(n):
t = mpmath.exp(2*pi*x/n)
tprime=(1/t**2)+(1/t)+1+t+t**2 ;
x0 = tprime.real
y0 = tprime.imag
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
if (x != 1):
ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
#ax.set_xlim(-1, 4)
#ax.set_ylim(-1, 1)
plt.show()
-:--- spicomplexe.py Bot L43 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py

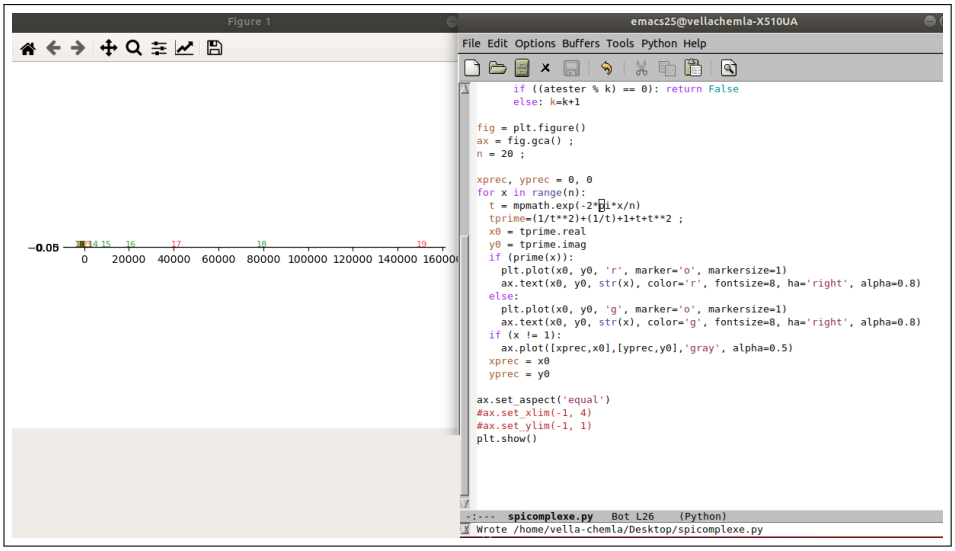


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca() ;
n = 20 ;
xprec, yprec = 0, 0
for x in range(n):
t = mpmath.exp(-2*pi*x/n)
tprime=(1/t**2)+(1/t)+1+t+t**2 ;
x0 = tprime.real
y0 = tprime.imag
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
if (x != 1):
ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
#ax.set_xlim(-1, 4)
#ax.set_ylim(-1, 1)
plt.show()
-:--- spicomplexe.py Bot L26 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py

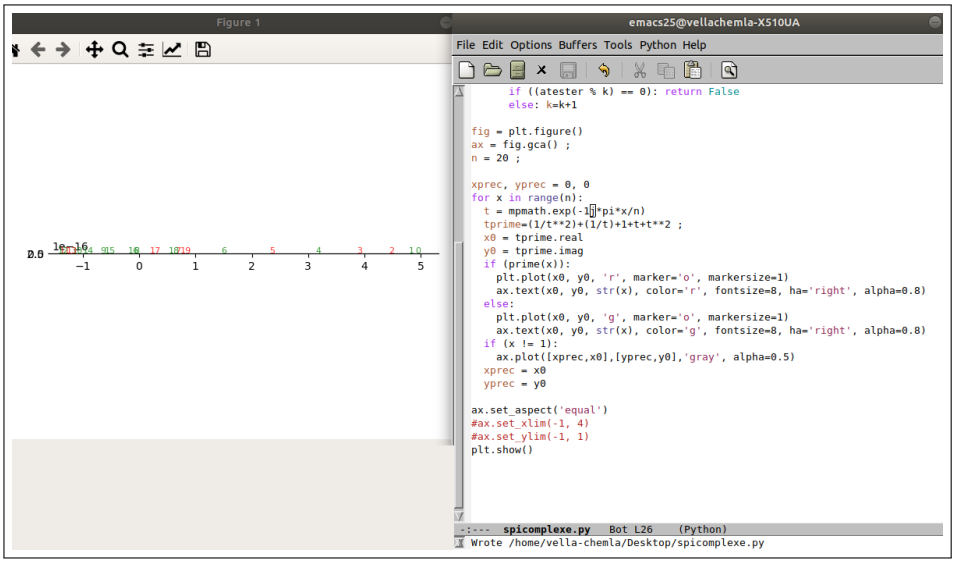


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca() ;
n = 20 ;
xprec, yprec = 0, 0
for x in range(n):
t = mpmath.exp(-1*pi*x/n)
tprime=(1/t**2)+(1/t)+1+t+t**2 ;
x0 = tprime.real
y0 = tprime.imag
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
if (x != 1):
ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
#ax.set_xlim(-1, 4)
#ax.set_ylim(-1, 1)
plt.show()
-:--- spicomplexe.py Bot L26 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py

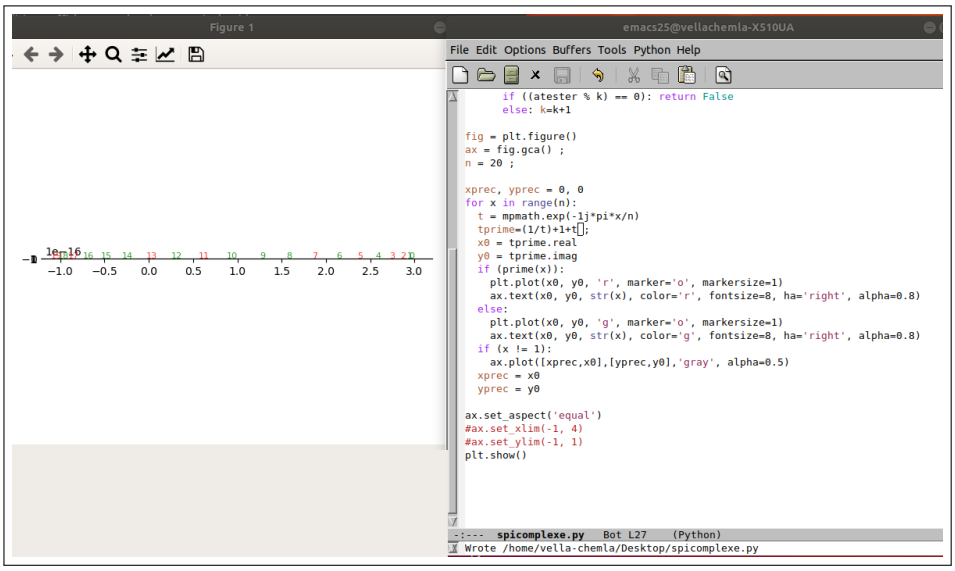
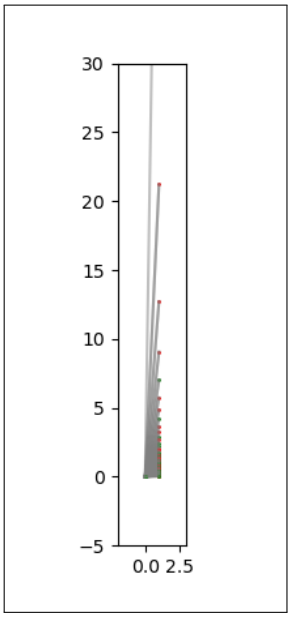


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca() ;
n = 20 ;
xprec, yprec = 0, 0
for x in range(n):
t = mpmath.exp(-1j*pi*x/n)
tprime=(1/t)+1+t;
x0 = tprime.real
y0 = tprime.imag
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
if (x != 1):
ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
#ax.set_xlim(-1, 4)
#ax.set_ylim(-1, 1)
plt.show()
-:--- spicomplexe.py Bot L27 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py


30
25
20
15
10
5
0
−5
0.0 2.5

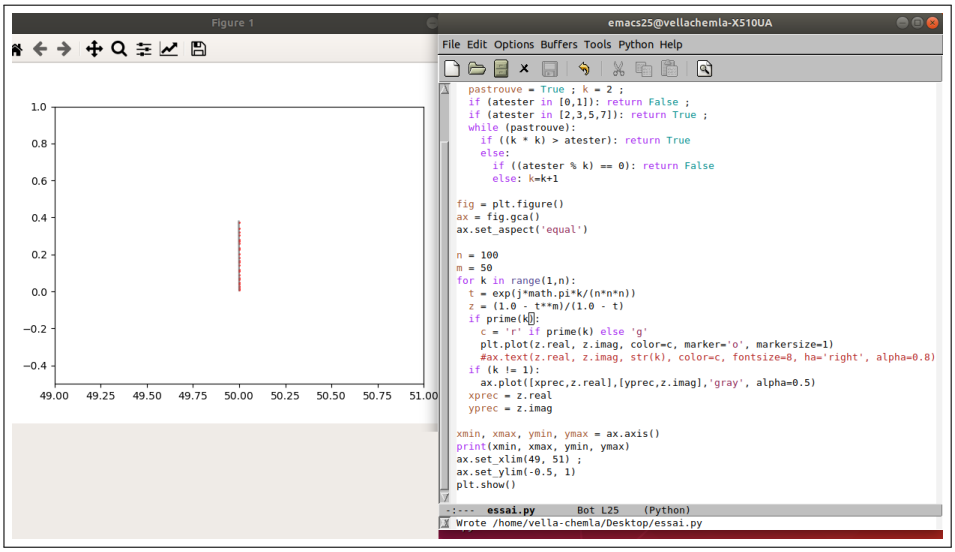


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 50
for k in range(1,n):
t = exp(j*math.pi*k/(n*n*n))
z = (1.0 - t**m)/(1.0 - t)
if prime(k):
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
#ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
ax.set_xlim(49, 51) ;
ax.set_ylim(-0.5, 1)
plt.show()
-:--- essai.py Bot L25 (Python)
Wrote /home/vella-chemla/Desktop/essai.py

Computer Science as an Art

Donald E. Knuth

Award Lecture

[The Turing Award citation read by Bernard A. Galler, chairman of the 1974 Turing Award Committee, on the presentation of this lecture on November 11 at the ACM Annual Conference in San Diego.]

The A.M. Turing Award of the ACM is presented annually by the ACM to an individual selected for his contributions of a technical nature made to the computing community. In particular, these contributions should have had significant influence on major segment of the computer field.

"The 1974 A.M. Turing Award is presented to Professor Donald E. Knuth of Stanford University for a number of major contributions to the analysis of algorithms and the design of programming languages, and in particular for his most significant contributions to the "art of computer programming" through his series of well-known books. The collections of techniques, algorithms and relevant theory in these books have served as a focal point for developing curricula and as an organizing influence on computer science."

Such a formal statement cannot put into proper perspective the role which Don Knuth has been playing in computer science, and in the computer industry as a whole. It has been my experience with respect to the first recipient of the Turing Award, Professor Alan J. Perlis, that at every meeting in which he participates he manages to provide the insight into the problems being discussed that becomes the focal point of discussion for the rest of the meeting. In a very similar way, the vocabulary, the examples, the algorithms, and the insight that Don Knuth has provided in his excellent collection of books and papers have begun to find their way into a great many discussions in almost every area of the field. This does not happen easily. As every author knows, even a single volume requires a great deal of careful organization and hard work. All the more must we appreciate the clear view and the patience and energy which Knuth must have had to plan seven volumes and to set about implementing his plan so carefully and thoroughly.

It is significant that this award and the others that he has been receiving are being given to him after three volumes of his work have been published. We are clearly ready to signal to everyone our appreciation of Don Knuth for his dedication and his contributions to our discipline. I am very pleased to have chaired the Committee that has chosen Don Knuth to receive the 1974 A.M. Turing Award of the ACM.

When Communications of the ACM began publication in 1959, the members of ACM's Editorial Board made the following remark as they described the purposes of ACM's periodicals [2] : "If computer programming is to become an important part of computer research and development, a transition of programming from an art to a disciplined science must be effected." Such a goal has been a continually recurring theme during the ensuing years ; for example, we read in 1970 of the "first steps toward transforming the art of programming into a science" [26]. Meanwhile we have actually succeeded in making our discipline a science, and in a remarkably simple way : merely by

Communications of the ACM, December 1974, Volume 17, number 12.
Author's address, Computer Science Department, Stanford University, Stanford, CA 94305

deciding to call it "computer science."

Implicit in these remarks is the notion that there is something undesirable about an area of human activity that is classified as an "art"; it has to be a Science before it has any real stature. On the other hand, I have been working for more than 12 years on a series of books called "The *Art* of Computer Programming." People frequently ask me why I picked such a title; and in fact some people apparently don't believe that I really did so, since I've seen at least one bibliographic reference to some books called "The *Act* of Computer Programming."

In this talk I shall try to explain why I think "Art" is the appropriate word. I will discuss what it means for something to be an art, in contrast to being a science; I will try to examine whether arts are good things or bad things; and I will try to show that a proper viewpoint of the subject will help us all to improve the quality of what we are now doing.

One of the first times I was ever asked about the title of my books was in 1966, during the last previous ACM national meeting held in Southern California. This was before any of the books were published, and I recall having lunch with a friend at the convention hotel. He knew how conceited I was, already at that time, so he asked if I was going to call my books "An Introduction to Don Knuth." I replied that, on the contrary, I was naming the books after *him*. His name Art Evans, (The Art of Computer Programming, in person.)

From this story we can conclude that the word "art" has more than one meaning. In fact, one of the nicest things about the word is that it is used in many different senses, each of which is quite appropriate in connection with computer programming. While preparing this talk, I went to the library to find out what people have written about the word "art" through the years; and after spending several fascinating days in the stacks, I came to the conclusion that "art" must be one of the most interesting words in the English language.

The Arts of Old

If we go back to Latin roots, we find *ars, artis* meaning "skill." It is perhaps significant that the corresponding Greek word was $\tau\acute{\epsilon}\chi\nu\eta$, the root of both "technology" and "technique."

Nowadays when someone speaks of "art" you probably think first of "fine arts" such as painting and sculpture, but before the twentieth century the word was generally used in quite a different sense. Since this older meaning of "art" still survives in many idioms, especially when we are contrasting art with science, I would like to spend the next few minutes talking about art in its classical sense.

In medieval times, the first universities were established to teach the seven so-called "liberal arts," namely grammar, rhetoric, logic, arithmetic, geometry, music, and astronomy. Note that this is quite different from the curriculum of today's liberal arts colleges, and that at least three of the original seven liberal arts are important components of computer science. At that time, an "art" meant something devised by man's intellect, as opposed to activities derived from nature or instinct; "liberal" arts were liberated or free, in contrast to manual arts such as plowing (cf. [6]). During the middle ages, the word "art" by itself usually meant logic [4], which usually meant the

study of syllogisms.

Science vs. Art

The word "science" seems to have been used for many years in about the same sense as "art" ; for example, people spoke also of the seven liberal sciences, which' were the same as the seven liberal arts [1], Duns Scotus in the thirteenth century called logic "the Science of Sciences, and the Art of Arts" (cf. [12, p. 34f]). As civilization and learning developed, the words took on more and more independent meanings, "science" being used to stand for knowledge, and "art" for the application of knowledge. Thus, the science of astronomy was the basis for the art of navigation. The situation was almost exactly like the way in which we now distinguish between "science" and "engineering."

Many authors wrote about the relationship between art and science in the nineteenth century, and I believe the best discussion was given by John Stuart Mill. He said the following things, among others, in 1843 [28] :

> Several sciences are often necessary to form the groundwork of a single art. Such is the complication of human affairs, that to enable one thing to be *done*, it is often requisite to *know* the nature and properties of many things... Art in general consists of the truths of Science, arranged in the most convenient order for practice, instead of the order which is the mast convenient for thought. Science groups and arranges its truths so as to enable us to take in at one view as much as possible of the general order of the universe. Art... brings together from parts of the field of science most remote from one another, the truths relating to the production of the different and heterogeneous conditions necessary to each effect which the exigencies of practical life require.

As I was looking up these things about the meanings of "art," I found that authors have been calling for a transition from art to science for at least two centuries. For example, the preface to a textbook on mineralogy, written in 1784, said the following [17] : "Previous to the year 1780, mineralogy, though tolerably understood by many as an Art, could scarce be deemed a Science."

According to most dictionaries "science" means knowledge that has been logically arranged and systematized in the form of general "laws." The advantage of science is that it saves us from the need to think things through in each individual case ; we can turn our thoughts to higher-level concepts. As John Ruskin wrote in 1853 [32] : "The work of science is to substitute facts for appearances, and demonstrations for impressions."

It seems to me that if the authors I studied were writing today, they would agree with the following characterization : Science is knowledge which we understand so well that we can teach it to a computer ; and if we don't fully understand something, it is an art to deal with it. Since the notion of an algorithm or a computer program provides us with an extremely useful test for the depth of our knowledge about any given subject, the process of going from an art to a science means that we learn how to automate something.

Artificial intelligence has been making significant progress, yet there is a huge gap between what computers can do in the foreseeable future and what ordinary people can do. The mysterious in-

sights that people have when speaking, listening, creating, and even when they are programming, are still beyond the reach of science ; nearly everything we do is still an art.

From this standpoint it is certainly desirable to make computer programming a science, and we have indeed come a long way in the 15 years since the publication of the remarks I quoted at the beginning of this talk. Fifteen years ago computer programming was so badly understood that hardly anyone even thought about proving programs correct ; we just fiddled with a program until we "knew" it worked. At that time we didn't even know how to express the *concept* that a program was correct, in any rigorous way. It is only in recent years that we have been learning about the processes of abstraction by which programs are written and understood ; and this new knowledge about programming is currently producing great payoffs in practice, even though few programs are actually proved correct with complete rigor, since we are beginning to understand the principles of program structure. The point is that when we write programs today, we know that we could in principle construct formal proofs of their correctness if we really wanted to, now that we understand how such proofs are formulated. This scientific basis is resulting in programs that are significantly more reliable than those we wrote in former days when intuition was the only basis of correctness.

The field of "automatic programming" is one of the major areas of artificial intelligence research today. Its proponents would love to be able to give a lecture entitled "Computer Programming as an Artifact" (meaning that programming has become merely a relic of bygone days), because their aim is to create machines that write programs better than we can, given only the problem specification. Personally I don't think such a goal will ever be completely attained, but I do think that their research is extremely important, because everything we learn about programming helps us to improve our own artistry. In this sense we should continually be striving to transform *every* art into a science : in the process, we advance the art.

Science and Art

Our discussion indicates that computer programming is by now *both* a science and an art, and that the two aspects nicely complement each other. Apparently most authors who examine such a question come to this same conclusion, that their subject is both a science and an art, whatever their subject is (cf. [25]). I found a book about elementary photography, written in 1893, which stated that "the development of the photographic image is both an art and a science" [13]. In fact, when I first picked up dictionary in order to study the words "art" and "science," I happened to glance at the editor's preface, which began by saying, "The making of a dictionary is both a science and an art." The editor of Funk & Wagnall's dictionary [27] observed that the painstaking accumulation and classification of data about words has a scientific character, while a well-chosen phrasing of definitions demands the ability to write with economy and precision : "The science without the art is likely to be ineffective ; the art without the science is certain to be inaccurate."

When preparing this talk I looked through the card catalog at Stanford library to see how other people have been using the words "art" and "science" in the titles of their books. This turned out to be quite interesting.

For example, I found two books entitled *The Art of Playing the Piano* [5, 15], and others called

The Science of Pianoforte Technique [10], *The Science of Pianoforte Practice* [30]. There is also a book called *The Art of Piano Playing : A Scientific Approach* [22].

Then I found a nice little book entitled *The Gentle Art of Mathematics* [31], which made me somewhat sad that I can't honestly describe computer programming as a "gentle art."

I had known for several years about a book called *The Art of Computation*, published in San Francisco, 1879, by a man named C. Frusher Howard [14]. This was a book on practical business arithmetic that had sold over 400,000 copies in various editions by 1890. I was amused to read the preface, since it shows that Howard's philosophy and the intent of his title were quite different from mine ; he wrote : "A knowledge of the Science of Number is of minor importance ; skill in the Art of Reckoning is absolutely indispensible."

Several books mention both science and art in their titles, notably *The Science of Being and Art of Living* by Maharishi Mahesh Yogi [24]. There is also a book called *The Art of Scientific Discovery* [11], which analyzes how some of the great discoveries of science were made.

So much for the word "art" in its classical meaning. Actually when I chose the title of my books, I wasn't thinking primarily of art in this sense, I was thinking more of its current connotations. Probably the most interesting book which turned up in my search was a fairly recent work by Robert E. Mueller called *The Science of Art* [29]. Of all the books I've mentioned, Mueller's comes closest to expressing what I want to make the central theme of my talk today, in terms of real artistry as we now understand the term. He observes : "It was once thought that the imaginative outlook of the artist was death for the scientist. And the logic of science seemed to spell doom to all possible artistic flights of fancy." He goes on to explore the advantages which actually do result from a synthesis of science and art.

A scientific approach is generally characterized by the words logical, systematic, impersonal, calm, rational, while an artistic approach is characterized by the words aesthetic, creative, humanitarian, anxious, irrational. It seems to me that both of these apparently contradictory approaches have great value with respect to computer programming.

Emma Lehmer wrote in 1956 that she had found coding to be "an exacting science as well as an intriguing art" [23]. H.S.M. Coxeter remarked in 1957 that he sometimes felt "more like an artist than a scientist" [7]. This was at the time C.P. Snow was beginning to voice his alarm at the growing polarization between "two cultures" of educated people [34, 35]. He pointed out that we need to combine scientific and artistic values if we are to make real progress.

Works of Art

When I'm sitting in an audience listening to a long lecture, my attention usually starts to wane at about this point in the hour. So I wonder, are you getting a little tired of my harangue about "science" and "art" ? I really hope that you'll be able to listen carefully to the rest of this, anyway, because now comes the part about which I feel most deeply.

When I speak about computer programming as an art, I am thinking primarily of it as an art *form*, in an aesthetic sense. The chief goal of my work as educator and author is to help people learn how to write *beautiful programs*. It is for this reason I was especially pleased to learn recently [32] that my books actually appear in the Fine Arts Library at Cornell University. (However, the three volumes apparently sit there neatly on the shelf, without being used, so I'm afraid the librarians may have made a mistake by interpreting my title literally.)

My feeling is that when we prepare a program, it can be like composing poetry or music ; as Andrei Ershov has said [9], programming can give us both intellectual and emotional satisfaction, because it is a real achievement to master complexity and to establish a system of consistent rules.

Furthermore when we read other people's programs, we can recognize some of them as genuine works of art. I can still remember the great thrill it was for me to read the listing of Stan Poley's SOAP II assembly program in 1958 ; you probably think I'm crazy, and styles have certainly changed greatly since then, but at the time it meant a great deal to me to sec how elegant a system program could be, especially by comparison with the heavy-handed coding found in other listings I had been studying at the same time. The possibility of writing beautiful programs, even in assembly language, is what got me hooked on programming in the first place.

Some programs are elegant, some are exquisite, some are sparkling. My claim is that it is possible to write *grand* programs, *noble* programs, truly *magnificent* ones !

Taste and Style

The idea of *style* in programming is now coming to the forefront at last, and I hope that most of you have seen the excellent little book on *Elements of Programming Style* by Kernighan and Plauger [16]. In this connection it is most important for us all to remember that there is no one "best" style ; everybody has his own preferences, and it is a mistake to try to force people into an unnatural mold. We often hear the saying, "I don't know anything about art, but I know what I like." The important thing is that you really *like* the style you are using ; it should be the best way you prefer to express yourself.

Edsger Dijkstra stressed this point in the preface to his *Short Introduction to the Art of Programming* [8] :

> It is my purpose to transmit the importance of good taste and style in programming, [but] the specific elements of style presented serve only to illustrate what benefits can be derived from "style" in general. In this respect I feel akin to the teacher of composition at a conservatory : he does not teach his pupils how to compose a particular symphony, he must help his pupils to find their own style and must explain to them what is implied by this. (It has been this analogy that made me talk about "The Art of Programming.")

Now we must ask ourselves "What is good style, and what is bad style ?" We should not be too rigid about this in judging other people's work. The early nineteenth-century philosopher Jeremy

Bentham put it this way [3, Bk. 3, Ch. 1] :

> Judges of elegance and taste consider themselves as benefactors to the human race, whilst they are really only the interrupters of their pleasure... There is no taste which deserves the epithet *good*, unless it be the taste for such employments which, to the pleasure actually produced by them, conjoin some contingent or future utility : there is no taste which deserves to be characterized as bad, unless it be a taste for some occupation which has a mischievous tendency.

When we apply our own prejudices to "reform" someone else's taste, we may be unconsciously denying him some entirely legitimate pleasure. That's why I don't condemn a lot of things programmers do, even though I would never enjoy doing them myself. The important thing is that they are creating something *they* feel is beautiful.

In the passage I just quoted, Bentham does give us some advice about certain principles of aesthetics which are better than others, namely the "utility" of the result. We have some freedom in setting up our personal standards of beauty, but it is especially nice when the things we regard as beautiful are also regarded by other people as useful. I must confess that I really enjoy writing computer programs ; and I especially enjoy writing programs which do the greatest good, in some sense.

There are many senses in which a program can be "good," of course. In the first place, it's especially good to have a program that works correctly. Secondly it is often good to have a program that won't be hard to change, when the time for adaptation arises. Both of these goals are achieved when the program is easily readable and understandable to a person who knows the appropriate language.

Another important way for a production program to be good is for it to interact gracefully with its users, especially when recovering from human errors in the input data. It's a real art to compose meaningful error messages or to design flexible input formats which are not error-prone.

Another important aspect of program quality is the efficiency with which the computer's resources are actually being used. I am sorry to say that many people nowadays are condemning program efficiency, telling us that it is in bad taste. The reason for this is that we are now experiencing a reaction from the time when efficiency was the only reputable criterion of goodness, and programmers in the past have tended to be so preoccupied with efficiency that they have produced needlessly complicated code ; the result of this unnecessary complexity has been that net efficiency has gone down, due to difficulties of debugging and maintenance.

The real problem is that programmers have spent far too much time worrying about efficiency in the wrong places and at the wrong times ; premature optimization is the root of all evil (or at least most of it) in programming.

We shouldn't be penny wise and pound foolish, nor should we always think of efficiency in terms of so many percent gained or lost in total running time or space. When we buy a car, many of us are almost oblivious to a difference of $50 or $100 in its price, while we might make a special trip to a particular store in order to buy a 5O¢ item for only 25¢. My point is that there is a time and place for efficiency ; I have discussed its proper role in my paper on structured programming,

which appears in the current issue of *Computing Surveys* [21].

Less Facilities : More Enjoyment

One rather curious thing I've noticed about aesthetic satisfaction is that our pleasure is significantly enhanced when we accomplish something with limited tools. For example, the program of which I personally am most pleased and proud is a compiler I once wrote for a primitive minicomputer which had only 4096 words of memory, 16 bits per word. It makes a person feel like a real virtuoso to achieve something under such severe restrictions.

A similar phenomenon occurs in many other contexts. For example, people often seem to fall in love with their Volkswagens but rarely with their Lincoln Continentals (which presumably run much better). When I learned programming, it was a popular pastime to do as much as possible with programs that fit on only a single punched card. I suppose it's this same phenomenon that makes APL enthusiasts relish their "one-liners." When we teach programming nowadays, it is a curious fact that we rarely capture the heart of a student for computer science until he has taken a course which allows "hands on" experience with a minicomputer. The use of our large-scale machines with their fancy operating systems and languages doesn't really seem to engender any love for programming, at least not at first.

It's not obvious how to apply this principle to increase programmers'enjoyment of their work. Surely programmers would groan if their manager suddenly announced that the new machine will have only half as much memory as the old. And I don't think anybody, even the most dedicated "programming artists," can be expected to welcome such a prospect, since nobody likes to lose facilities unnecessarily. Another example may help to clarify the situation : Film-makers strongly resisted the introduction of talking pictures in the 1920's because they were justly proud of the way they could convey words without sound. Similarly, a true programming artist might well resent the introduction of more powerful equipment ; today's mass storage devices tend to spoil much of the beauty of our old tape sorting methods. But today's film makers don't want to go back to silent films, not because they're lazy but because they know it is quite possible to make beautiful movies using the improved technology. The form of their art has changed, but there is still plenty of room for artistry.

How did they develop their skill ? The best film makers through the years usually seem to have learned their art in comparatively primitive circumstances, often in other countries with a limited movie industry. And in recent years the most important things we have been learning about programming seem to have originated with people who did not have access to very large computers. The moral of this story, it seems to me, is that we should make use of the idea of limited resources in our own education. We can all benefit by doing occasional "toy" programs, when artificial restrictions are set up, so that we are forced to push our abilities to the limit. We shouldn't live in the lap of luxury all the time, since that tends to make us lethargic. The art of tackling miniproblems with all our energy will sharpen our talents for the real problems, and the experience will help us to get more pleasure from our accomplishments on less restricted equipment.

In a similar vein, we shouldn't shy away from "art for art's sake" ; we shouldn't feel guilty about programs that are just for fun. I once got a great kick out of writing a one-statement ALGOL

program that invoked an innerproduct procedure in such an unusual way that it calculated the mth prime number, instead of an innerproduct [19]. Some years ago the students at Stanford were excited about finding the shortest FORTRAN program which prints itself out, in the sense that the program's output is identical to its own source text. The same problem was considered for many other languages. I don't think it was a waste of time for them to work on this; nor would Jeremy Bentham, whom I quoted earlier, deny the "utility" of such pastimes [3, Bk. 3, Ch. 1]. "On the contrary," he wrote, "there is nothing, the utility of which is more incontestable. To what shall the character of utility be ascribed, if not to that which is a source of pleasure?"

Providing Beautiful Tools

Another characteristic of modern art is its emphasis on creativity. It seems that many artists these days couldn't care less about creating beautiful things; only the novelty of an idea is important. I'm not recommending that computer programming should be like modern art in this sense, but it does lead me to an observation that I think is important. Sometimes we are assigned to a programming task which is almost hopelessly dull, giving us no outlet whatsoever for any creativity; and at such times a person might well come to me and say, "So programming is beautiful? It's all very well for you to declaim that I should take pleasure in creating elegant and charming programs, but how am I supposed to make this mess into a work of art?" Well, it's true, not all programming tasks are going to be fun. Consider the "trapped housewife," who has to clean off the same table every day : there's not room for creativity or artistry in every situation. But even in such cases, there is way to make a big improvement : it is still a pleasure to do routine jobs if we have beautiful things to work with. For example, a person will really enjoy wiping off the dining room table, day after day, if it is a beautifully designed table made from some fine quality hardwood.

Therefore I want to address my closing remarks to the system programmers and the machine designers who produce the systems that the rest of us must work with. *Please*, give us tools that are a pleasure to use, especially for our routine assignments, instead of providing something we have to fight with. Please, give us tools that encourage us to write better programs, by enhancing our pleasure when we do so.

It's very hard for me to convince college freshmen that programming is beautiful, when the first thing I have to tell them is how to punch "slash slash JOB equals so-and-so." Even job control languages can be designed so that they are a pleasure to use, instead of being strictly functional.

Computer hardware designers can make their machines much more pleasant to use, for example by providing floating-point arithmetic which satisfies simple mathematical laws. The facilities presently available on most machines make the job of rigorous error analysis hopelessly difficult, but properly designed operations would encourage numerical analysts to provide better subroutines which have certified accuracy (cf. [20, p. 204]).

Let's consider also what software designers can do. One of the best ways to keep up the spirits of a system user is to provide routines that he can interact with. We shouldn't make systems too automatic, so that the action always goes on behind the scenes; we ought to give the programmer-user a chance to direct his creativity into useful channels. One thing all programmers have in com-

mon is that they enjoy working with machines ; so let's keep them in the loop. Some tasks are best done by machine, while others are best done by human insight ; and a properly designed system will find the right balance. (I have been trying to avoid misdirected automation for many years, cf. [18].)

Program measurement tools make a good case in point. For years, programmers have been unaware of how the real costs of computing are distributed in their programs. Experience indicates that nearly everybody has the wrong idea ubout the real bottlenecks in his programs ; it is no wonder that attempts at efficiency go awry so often, when a programmer is never given a breakdown of costs according to the lines of code he has written. His job is something like that of a newly married couple who try to plan a balanced budget without knowing how much the individual items like food, shelter, and clothing will cost. All that we have been giving programmers is an optimizing compiler, which mysteriously does something to the programs it translates but which never explains what it does. Fortunately we are now finally secing the appearance of systems which give the user credit for some intelligence ; they automatically provide instrumentation of programs and appropriate feedback about the real costs. These experimental systems have been a huge success, because they produce measurable improvements, and especially because they are fun to use, so I am confident that it is only a matter of time before the use of such systems is standard operating procedure. My paper in *Computing Surveys* [21] discusses this further, and presents some ideas for other ways in which an appropriate interactive routine can enhance the satisfaction of user programmers.

Language designers also have an obligation to provide languages that encourage good style, since we all know that style is strongly influenced by the language in which it is expressed. The present surge of interest in structured programming has revealed that none of our existing languages is really ideal for dealing with program and data structure, nor is it clear what an ideal language should be. Therefore I look forward to many careful experiments in language design during the next few years.

Summary

To summarize : We have seen that computer programming is an art, because it applies accumulated knowledge to the world, because it requires skill and ingenuity, and especially because it produces objects of beauty. A programmer who subconsciously views himself as an artist will enjoy what he does and will do it better. Therefore we can be glad that people who lecture at computer conferences speak about the *state of the Art*.

References

1. Bailey, Nathan. *The Universal Etymological English Dictionary*, T. Cox, London, 1727, See "Art," "Liberal," and "Science."

2. Bauer, Walter F., Juncosa, Mario L., and Perlis, Alan J. ACM publication policies and plans. *J. ACM* 6 (Apr. 1959), 121-122.

3. Bentham, Jeremy. *The Rationale of Reward*. Trans. from *Théorie des peines et des récompenses*, 1811, by Richard Smith. J. & H. L. Hunt, London, 1825.

4. *The Century Dictionary and Cyclopedia 1.* The Century Co., New York, 1889.

5. Clementi, Muzio. *The Art of Playing the Piano.* Trans. from *L'art de jouer le pianoforte* by Max Vogrich. Schirmer, New York, 1898.

6. Colvin, Sidney. "Art." *Encyclopaedia Britannica*, eds 9, 11, 12, 13, 1875-1926.

7. Coxeter, H. S. M. Convocation address, Proc. 4th Canadian Math. Congress, 1957, pp. 8-10.

8. Dijkstra, Edsger W. EWD316 : *A Short Introduction to the Art of Programming.* T. H. Eindhoven, The Netherlands, Aug. 1971.

9. Ershov, A. P. Aesthetics and the human factor in programming, *Comm. ACM 15* (July 1972), 501-505.

10. Fielden, Thomas. *The Science of Pianoforte Technique.* Macmillan, London, 1927.

11. Gore, George, *The Art of Scientific Discovery.* Longmans, Green, London, 1878.

12, Hamilton, Willian, *Lectures on Logic 1.* Wm. Blackwood, Edinburgh, 1874.

13. Hodges, John A. Elementary Photography : *The "Amateur Photographer" Library 7.* London, 1893. Sixth ed, revised and enlarged, 1907, p. 58.

14. Howard, C. Frusher. Howard's *Art of Computation* and golden rule for equation of payments for schools, business colleges and self-culture.... C.F. Howard, San Francisco, 1879.

15. Hummel, J.N. *The Art of Playing she Piano Forte.* Boosey, London, 1827.

16. Kernighan B.W., and Plauger, P.J. *The Elements of Programming Style.* McGraw-Hill, New York, 1974.

17. Kirwan, Richard. *Elements of Mineralogy.* Elmsly, London, 1784.

18. Knuth, Donald E, Minimizing drum latency time. *J. ACM 8* (Apr. 1961), 119-150.

19. Knuth, Donald E., and Merner, J.N. ALGOL 60 confidential. *Comm. ACM 4* (June 1961), 268-272.

20. Knuth, Donald E. Seminumerical Algorithms : *The Art of Computer Programming 2.* Addison-Wesley, Reading, Mass., 1969.

21. Knuth, Donald E. Structured programming with go to statements, *Computing Surveys 6* (Dec. 1974), pages in makeup.

22. Kochevitsky, George. *The Art of Plano Playing : A Scientific Approach.* Summy-Birchard, Evanston, III, 1967.

23. Lehmer, Emma. Number theory on the SWAC. *Proc. Symp. Applied Math. 6*, Amer. Math. Soc. (1956), 103-108.

24. Mahesh Yogi, Maharishi. *The Science of Being and Art of Living.* Allen & Unwin, London, 1963.

25. Malevinsky, Moses L. *The Science of Playwriting.* Brentano's, New York, 1925.

26. Manna, Zohar, and Pnueli, Amir. Formalization of properties of functional programs. *J. ACM 17* (July 1970), 555-569.

27. Marckwardt, Albert H. Preface to *Funk and Wagnall's Standard College Dictionary*. Harcourt, Brace & World, New York, 1963, vii.

28. Mill, John Stuart. *A System af Logic, Ratiocinative and Inductive*. London, 1843, The quotations are from the introduction, §2, and from Book 6, Chap. 11 (12 in later editions), §5.

29, Mueller, Robert E. *The Science of Art*. John Day, New York, 1967.

30. Parsons, Albert Ross. *The Science of Pianoforte Practice*. Schirmer, New York, 1886.

31. Pedoe, Daniel. *The Gentle Art of Machematics*. English U. Press, London, 1953.

32. Ruskin, John. *The Stones of Venice 3*. London, 1853.

33. Salton, G.A. Personal communication, June 21, 1974.

34. Snow, C.P. *The two cultures. The New Statesman and Nation 52* (Oct. 6, 1956), 413-414.

35. Snow, C.P. *The Two Cultures : and a Second Look*. Cambridge University Press, 1964.

L'informatique comme un art

Donald E. Knuth

Conférence de remise de récompense

[La citation pour la remise de la récompense Turing a été lue par Bernard A. Galler, directeur du Comité de remise de la récompense Turing en 1974, en présentation de cette conférence le 11 novembre à la conférence annuelle de l'ACM à San Diego.]

La récompense A.M. Turing de l'ACM est attribuée chaque année à un individu sélectionné par l'ACM pour ses contributions de nature technique faites à la communauté informatique. En particulier, ces contributions devraient avoir une influence significative sur un segment majeur du domaine de l'informatique.

"La récompense A.M. Turing pour l'année 1974 est attribuée au Professeur Donald E. Knuth de l'Université de Stanford pour ces nombreuses contributions à l'analyse des algorithmes et à la conception des langages de programmation, et en particulier à ses notables contributions à l'"Art de la programmation des ordinateurs" à travers sa série de livres très renommés. Les collections de techniques, d'algorithmes et de théorie dans ces livres a servi de point focal pour développer les curricula informatiques et a influencé l'organisation de l'informatique."

Une telle présentation formelle ne met pas réellement en perspective le rôle que Don Knuth a joué en informatique, ainsi que dans l'industrie des ordinateurs vue comme un tout. Cela a été mon sentiment par rapport au premier récipiendaire de la récompense Turing, le Professeur Alan J. Perlis, qui à chaque meeting auquel il participe réussit à fournir l'éclairage sur les problèmes discutés qui devient le point focal de la discussion pour le reste de la rencontre. D'une façon très similaire, le vocabulaire, les exemples, les algorithmes, et l'éclairage que Don Knuth a fournis dans son excellente collection de livres et articles a commencé à trouver son chemin dans un très grand nombre de discussions dans la plupart des champs du domaine. Cela n'advient pas aisément. Comme tout auteur le sait, même un seul livre nécessite une grande quantité d'une organisation sans faille et un dur travail. Tous peuvent apprécier la vision claire et la patience et l'énergie avec lesquels Knuth a pu planifier sept volumes et comment il a pu mener à bien son plan si précautionneusement et de façon si approfondie.

Il est remarquable que cette récompense et les autres qu'il a déjà reçues lui soient données après que trois volumes de cette œuvre aient été publiés. Nous sommes maintenant prêts à dire à tous combien nous apprécions le dévouement de Don Knuth et ses contributions à notre discipline. Je suis très heureux d'avoir présidé le Comité qui a choisi Don Knuth comme le récipiendaire de la récompense A.M. Turing Award de l'ACM en cette année 1974.

Quand les Communications de l'ACM ont commencé à publier en 1959, les membres du comité éditorial de l'ACM firent la remarque suivante alors qu'ils décrivaient les objectifs des périodiques de l'ACM [2] : "Si la programmation des ordinateurs doit devenir une partie importante de la recherche et du développement des ordinateurs, une transition faisant passer la programmation du statut d'art à celui de discipline scientifique doit avoir lieu." Un tel objectif a été un thème toujours récurrent les années suivantes ; par exemple, nous lisons en 1970 les "premières étapes en vue de transformer l'art de la programmation en une science" [26]. Pendant tout ce temps, nous avons

Communications de l'ACM, Décembre 1974, Volume 17, numéro 12.
Adresse de l'auteur, Département d'informatique, Université de Stanford, Stanford, CA 94305

vraiment réussi à transformer notre discipline en science, et d'une façon remarquablement simple : seulement en décidant de l'appeler la "science des ordinateurs".

Sous-jacente à ces remarques, il y a l'idée qu'il n'est pas très désirable qu'un domaine de l'activité humaine soit classé comme un "art" ; il doit être une science avant d'acquérir tout statut réel. D'un autre côté, j'ai travaillé plus de 12 ans sur une série de livres appelés "L'*Art* de la programmation des ordinateurs." Les gens me demandent souvent pourquoi j'ai choisi un tel titre ; et en fait, quelques personnes apparemment ne croient pas que je l'aie vraiment fait, puisque j'ai vu au moins une référence bibliographique à mes livres sous la forme l'"Acte de la programmation des ordinateurs."

Dans cet exposé, j'essaierai d'expliquer pourquoi je pense que le mot "Art" est le mot approprié. Je discuterai de ce que cela signifie pour quelque chose d'être un art, par opposition au fait d'être une science ; j'essaierai d'examiner si les arts sont de bonnes ou de mauvaises choses ; et j'essaierai de montrer qu'un point de vue approprié sur ce sujet nous aidera à améliorer la qualité de ce que nous faisons.

L'une des premières fois où j'ai été interrogé à propos du titre de mes livres eut lieu en 1966, durant le meeting national précédent de l'ACM qui eut lieu dans le sud de la Californie. C'était avant qu'aucun de ces livres n'ait été publié et je me rappelle avoir déjeuné avec un ami à l'hôtel du colloque. Il savait combien j'étais vaniteux, déjà à cette époque, alors il me demanda si j'allais appeler mes livres "Une introduction à Don Knuth." Je lui répondis que, au contraire, j'avais appelé ces livres comme une introduction à *lui*. Son nom était Art Evans, (L'Art de la programmation des ordinateurs en personne.)

De cette histoire, nous pouvons conclure que le mot "art" a plus d'un sens. En fait, un des plus belles choses au monde est qu'il est utilisé selon de nombreux sens différents, chacune de ces acceptions étant assez appropriée pour la programmation des ordinateurs. Lorsque j'ai préparé cet exposé, je suis allé à la bibliothèque pour trouver ce que les gens avait écrit à propos du mot "art" au fil des ans ; et après avoir passé quelques jours fascinants parmi les piles de livres, j'en vins à la conclusion que le mot "art" doit être l'un des mots les plus intéressants de la langue anglaise.

Les arts des anciens

Si l'on revient aux racines latines, on trouve qu'*ars, artis* signifie "compétence." Cela est sûrement significatif que le mot grec correspondant soit $\tau\acute{\epsilon}\chi\nu\eta$, la racine à la fois du mot "technologie" et du mot "technique."

De nos jours, quand on parle d'"art", on pense probablement d'abord aux Beaux-Arts, comme la peinture et la sculpture, mais avant le vingtième siècle, le mot était généralement utilisé dans un sens plutôt différent. Puisque cet ancien sens de "art" perdure dans de nombreux idiomes, particulièrement lorsque nous opposons l'art et la science, je voudrais prendre quelques minutes pour vous parler de l'art au sens classique du terme.

Au Moyen-Âge, les premières universités furent créées pour enseigner les "arts libéraux", que sont

la grammaire, la réthorique, la logique, l'arithmétique, la géométrie, la musique et l'astronomie. Notez combien cela diffère du curriculum enseigné dans les lycées d'arts libéraux de nos jours, ainsi que la manière dont trois des sept arts libéraux initiaux sont des composants importants de l'informatique. À cette époque, un "art" était une activité conçue par l'intellect humain, par opposition aux activités dérivées de la nature ou de l'instinct ; les arts "libéraux" étaient libérés, ou libres, par opposition aux arts manuels, comme le labour (cf. [6]). Au Moyen-Âge, le mot "art" lui-même désignait habituellement la logique [4], qui était alors l'étude des syllogismes.

Science vs. Art

Le mot "science" semble avoir été utilisé pendant de nombreuses années à peu près dans le même sens que le mot "art" ; par exemple, les gens parlaient également des sept sciences libérales, qui étaient identiques aux sept arts libéraux [1], Duns Scotus au treizième siècle appelait la logique "la Science des Sciences, et l'Art des Arts" (cf. [12, p. 34f]). Alors que la civilisation et les connaissances se développaient, les mots prirent des significations de plus en plus indépendantes, le mot "science" étant utilisé pour désigner les connaissances, et le mot "art" pour l'application des connaissances. Ainsi, la science astronomique était la base de l'art de la navigation. La situation ressemblait à la manière dont aujourd'hui, nous faisons la distinction entre la "science" et l'"ingénierie."

De nombreux auteurs ont écrit à propos des relations entre art et science au dix-neuvième siècle, et je crois que les meilleures arguments ont été donnés par John Stuart Mill. Il a dit les choses suivantes, parmi d'autres, en 1843 [28] :

> Plusieurs sciences sont souvent nécessaires pour former le socle d'un seul art. Telle est la complexité des affaires des hommes, qui permet que pour qu'une seule chose soit *faite*, il est souvent nécessaire de *connaître* la nature et les propriétés de nombreuses choses... L'art en général consiste à assembler les vérités de la science, de la manière la plus efficace pour que cet art soit pratiqué, plutôt que de la manière la plus efficace pour que cet art soit pensé. Les domaines scientifiques groupent et arrangent leurs vérités de manière à nous permettre d'appréhender autant que possible d'un seul regard l'ordre général de l'univers. L'art... assemble les vérités de différents domaines de la science éloignés les uns des autres, vérités liées à la production de conditions différentes et hétérogènes nécessaires à chaque effet que nécessitent les exigences de la vie pratique.

Comme je regardais ces choses à propos de la signification du mot "art," je trouvai que les auteurs parlaient d'une transition de l'art à la science durant au moins deux siècles. Par exemple, on trouve dans la préface d'un livre de minéralogie, écrit en 1784, la phrase suivante [17] : "Avant l'année 1780, la minéralogie, bien qu'elle ait pu vraiment être perçue par beaucoup comme un art, pouvait difficilement être considérée comme une science."

Selon la plupart des dictionnaires, le mot "science" signifie une connaissance qui a été logiquement organisée et systématisée sous la forme de "lois" générales. L'avantage de la science est qu'elle nous préserve de la nécessité de penser aux choses dans chaque cas particulier ; nous pouvons orienter nos pensées vers des concepts de plus haut niveau. Comme l'a écrit John Ruskin en 1853 [32] : "Le travail de la science consiste à remplacer les apparences par des faits, et les impressions par des

démonstrations."

Il me semble que si les auteurs que j'ai étudiés écrivaient de nos jours, ils seraient d'accord avec la caractérisation suivante : la science est une connaissance que nous comprenons si bien que nous pouvons l'inculquer à un ordinateur ; et si nous ne comprenons pas correctement quelque chose, c'est un art que d'y faire face. Puisque la notion d'algorithme et de programme d'ordinateur nous fournit un test extrêmement utile pour connaître la profondeur de notre connaissance de n'importe quel sujet, le processus consistant à aller de l'art vers la science signifie que nous apprenons à automatiser ce processus.

L'intelligence artificielle a fait des progrès significatifs, même s'il reste un écart énorme entre ce que les ordinateurs pourront faire dans un futur proche et ce que les gens ordinaires sont capables de faire. Les processus mystérieux mis en œuvre par les personnes lorsqu'elles parlent, écoutent, créent, et même lorsqu'elles programment, sont hors de portée de la science ; presque tout ce que nous faisons est encore artisanal.

De ce point de vue, il est certainement souhaitable de faire de l'informatique une science, et nous avons en effet suivi un long chemin pendant ces 15 années nous séparant de la publication des remarques que j'ai citées au début de mon intervention. Il y a quinze ans, la programmation des ordinateurs était si mal comprise que presque personne ne pensait à prouver que les programmes étaient corrects ; nous nous dépatouillons avec un programme jusqu'à ce que nous "sachions" qu'il marchait. À ce moment-là, nous ne savions même pas comment exprimer le *concept* de correction d'un programme, d'une façon rigoureuse quelle qu'elle soit. C'est seulement dans les années récentes que nous avons appris le processus d'abstraction par lequel les programmes sont écrits et compris ; et cette nouvelle connaissance à propos de la programmation est en train de produire de gros gains en pratique, même si peu de programmes sont vraiment prouvés comme étant corrects selon une parfaite rigueur, puisque nous commençons à comprendre les principes de la structure des programmes. Le point important est que quand nous écrivons des programmes aujourd'hui, nous savons que nous pourrions en principe construire des preuves formelles de leur correction si nous le voulions vraiment, maintenant que nous comprenons comment de telles preuves doivent être écrites. Cette base scientifique résultent dans des programmes plus robustes que ceux que nous écrivions aux premiers jours de l'informatique quand l'intuition était la seule base étayant la correction des programmes.

Le domaine de la "programmation automatique" est l'un des champs de recherche les plus vastes de l'intelligence artificielle aujourd'hui. Ses partisans rêveraient de donner une conférence intitulée "la programmation des ordinateurs comme artefact" (signifiant que la programmation serait simplement devenue une relique des jours passés), parce que leur objectif est de créer des machines qui écriraient des programmes mieux que nous, en leur donnant seulement les spécifications d'un problème. Personnellement, je ne pense pas qu'un tel but soit jamais atteint, mais je pense que leur recherche est extrêmement importante, parce que tout ce que nous apprenons au sujet de la programmation nous aide à améliorer notre propre art. En ce sens-là, nous devrions sans cesse nous efforcer de transformer *tout* art en une science : dans ce processus, nous faisons avancer l'art.

Science et Art

Notre exposé montre que la programmation des ordinateurs est maintenant *à la fois* une science et un art, et que les deux aspects se complémentent joliment l'un l'autre. Apparemment la plupart des auteurs qui étudient un telle question aboutissent à cette même conclusion, que leur sujet est à la fois une science et un art, quel que soit leur sujet (cf. [25]). J'ai trouvé un livre au sujet de la photographie, écrit en 1893, qui énonçait que "le développement de l'image photographique est à la fois un art et une science" [13]. En fait, quand j'ai ouvert un dictionnaire pour la première fois pour étudier les définitions des mots "art" et "science," j'ai jeté un œil aux préfaces des éditeurs, qui commençaient par dire "L'écriture d'un dictionnaire est à la fois une science et un art." L'éditeur du dictionnaire Funk & Wagnall [27] a observé que l'accumulation méticuleuse et la classification des données à propos des mots a un caractère scientifique, dans la mesure où l'écriture de définitions contenant des mots bien choisis nécessite d'être capable d'écrire avec économie et précision : "La science sans l'art est susceptible d'être inefficace ; l'art sans la science est de façon certaine inapproprié."

En préparant cette conférence, j'ai parcouru le catalogue des cartes à la bibliothèque de Stanford pour voir comment d'autres personnes avaient utilisé les mots "art" et "science" dans les titres de leurs livres. Cela s'est avéré assez intéressant.

Par exemple, j'ai trouvé deux livres intitulés *L'art de jouer du piano* [5, 15], et d'autres appelés *La science de la technique du pianoforte* [10], *La science de la pratique du pianoforte* [30]. Il y a également un livre appelé *L'art de jouer du piano : une approche scientifique* [22].

J'ai aussi trouvé un joli petit livre intitulé *L'art doux des mathématiques* [31], qui m'a rendu assez triste parce que je ne peux honnêtement décrire la programmation des ordinateurs comme un "art doux".

Je connaissais depuis plusieurs années un livre dont le titre est *L'art du calcul*, publié à San Francisco, en 1879, par un homme nommé C. Frusher Howard [14]. C'était un livre sur l'arithmétique pratique pour les affaires qui avait été vendu à 400 000 exemplaires dans plusieurs éditions depuis 1890. J'avais été amusé par la lecture de la préface de ce livre, puisqu'elle montre que la philosophie d'Howard et l'intention de son titre étaient quelque peu différentes des miennes ; il écrit : "La connaissance de la science des nombres est d'une importance mineure ; la compétence dans l'art du calcul est absolument indispensable."

Plusieurs livres mentionnent à la fois la science et l'art dans leur titre, notamment *La science d'être et l'art de vivre* de Maharishi Mahesh Yogi [24]. Il y a aussi un livre dont le titre est *L'art de la découverte scientifique* [11], qui analyse comment ont été faites quelques unes des grandes découvertes de la science.

Voici tout ce qu'on peut trouver à propos du mot "art" dans son acception classique. Vraiment lorsque j'ai choisi ce titre pour mes livres, je ne pensais pas en premier lieu à l'art dans ce sens-là, je pensais davantage à ses acceptions courantes. Le livre probablement le plus intéressant qui se présenta à moi durant ces recherches est un livre relativement récent de Robert E. Mueller intitulé

La science de l'art [29]. De tous les livres que j'ai mentionnés, celui de Mueller est le plus proche de ce que je souhaitais être le thème central de mon allocution aujourd'hui, par rapport à l'art tel que nous entendons ce terme maintenant. Il observe : "On a un jour pensé que les perspectives imaginatives de l'artiste signaient la mort du scientifique. Et que la logique de la science semblait jeter un sort à tous les vols artistiques de la fantaisie." Il continue à explorer les avantages qui résultent effectivement d'une synthèse entre la science et l'art.

Une approche scientifique est en général caractérisée par les mots logique, systématique, impersonnelle, calme, rationnelle, alors qu'une approche artistique est caractérisée par les mots esthétique, créative, humanitaire, anxieuse, irrationnelle. Il me semble que ces deux approches apparemment contradictoires sont très appropriées pour qualifier la programmation des ordinateurs.

Emma Lehmer a écrit en 1956 qu'elle avait trouvé que la programmation était "une science exigeante autant qu'un art intrigant" [23]. H.S.M. Coxeter remarqua en 1957 qu'il se percevait lui-même "plus comme un artiste que comme un scientifique." [7]. C'est à cette époque que C.P. Snow tirait la sonnette d'alarme au sujet de la polarisation grandissante entre "deux cultures" des personnes cultivées [34, 35]. Il insista sur le fait que nous avions besoin de combiner les valeurs scientifiques et artistiques si nous voulions réellement progresser.

Les œuvres d'art

Lorsque je suis auditeur d'une longue conférence, mon attention tend à diminuer à peu près à ce moment dans l'heure. Par conséquent, je me demande si vous commencez à être las de ma harangue à propos de la "science" et de l'"art" ? J'espère vraiment que vous pourrez écouter attentivement ce qui va suivre, parce que c'est maintenant qu'arrive la partie que je ressens la plus profondément.

Quand je parle de programmation des ordinateurs comme un art, je pense d'abord à elle comme à une *forme* d'art, au sens esthétique du terme. Le but principal de mon travail en tant qu'enseignant et auteur est d'aider les gens à apprendre à écrire de *beaux programmes*. C'est pour cette raison que j'étais particulièrement heureux d'apprendre récemment [32] que mes livres étaient présentés dans la bibliothèque des Beaux-Arts à l'Université Cornell. (Pourtant, il semblerait que les trois volumes soient soigneusement restés sur l'étagère, sans être utilisés, et je crains que les bibliothécaires n'aient fait une erreur en interprétant leur titre d'une manière trop littérale.)

Mon sentiment est que quand nous préparons un programme, cela peut être comme lorsque nous composons un poème ou un morceau de musique ; comme l'a dit Andrei Ershov [9], la programmation peut nous donner à la fois une satisfaction intellectuelle et une satisfaction émotionnelle, parce que cela consiste effectivement à maîtriser la complexité et à établir un système de règles consistantes.

En outre, quand nous lisons les programmes des autres, nous pouvons voir certains d'entre eux comme de véritables œuvres d'art d'ingéniosité. J'ai encore le souvenir du grand frisson que j'ai ressenti en lisant le listing du programme en assembleur de Stan Poley SOAP II en 1958 ; vous pensez probablement que je suis fou, et les styles ont probablement beaucoup changé depuis lors, mais à ce moment-là, cela avait beaucoup de signification pour moi que de voir à quel point un programme

système pouvait être élégant, spécialement en le comparant aux autres codes maladroits que j'avais trouvés dans d'autres listings que j'avais étudiés au même moment. La possibilité d'écrire de beaux programmes, même en assembleur, est ce qui m'a accroché à la programmation en premier lieu.

Certains programmes sont élégants, d'autres sont exquis, d'autres sont pétillants. Ce que je prétends, c'est qu'il est possible d'écrire de *grands* programmes, de *nobles* programmes, des programmes vraiment *magnifiques* !

Le goût et le style

L'idée d'un *style* de programmation devient d'avant-garde maintenant, et j'espère que la plupart d'entre vous avez vu l'excellent petit livre sur les *Éléments de style de programmation* de Kernighan et Plauger [16]. Dans cette relation faite ici, il est plus important de rappeler pour nous tous qu'il n'y a pas un style "meilleur" qu'un autre ; tout le monde a ses propres préférences, et c'est une erreur que d'essayer de forcer les gens à entrer dans un moule qui ne leur serait pas naturel. Nous entendons souvent dire "Je ne connais rien à l'art, mais je sais ce que j'aime". La chose importante est que vous *aimez* vraiment le style qui est le vôtre ; cela devrait être la façon par laquelle vous préférez vous exprimer.

Edsger Dijkstra a souligné ce point dans la préface de sa *Courte introduction à l'art de la programmation* [8] :

> Mon but est de transmettre l'important du bon goût et du style dans la programmation, [mais] les éléments spécifiques de style qui seront présentés ne serviront qu'à illustrer quels bénéfices peuvent être tirés du "style" en général. Selon ce point de vue, je me sens comme le professeur de composition au conservatoire : il n'apprend pas à ses élèves à composer une symphonie particulière, il les aide à trouver leur propre style et doit leur expliquer ce que cela implique. (C'est cette analogie qui m'a fait parler de "l'art de la programmation.")

Maintenant nous devons nous demander "Qu'est-ce qu'un bon style et qu'est-ce qu'un mauvais style ?" Nous ne devrions pas être trop rigides par rapport à cela en jugeant le travail d'autrui. Le philosophe du début du dix-neuvième siècle Jeremy Bentham a noté cela ainsi [3, Bk. 3, Ch. 1] :

> Les juges de l'élégance et du bon goût se considèrent comme des bienfaiteurs de l'humanité, alors qu'ils ne sont que les interrupteurs de son plaisir... Il n'y a pas de goût qui mérite l'épithète *bon*, à moins que ça ne soit un goût qui soit utilisé de telle manière que, par le plaisir qu'il procure vraiment, il procure également une utilité simultanée ou future : il n'y a pas de goût qui puisse mériter d'être qualifié de mauvais, à moins qu'il soit le goût pour une activité qui est d'une mauvaise tendance.

Quand nous appliquons nos propres préjugés pour "réformer" le goût de quelqu'un d'autre, nous pouvons inconsciemment lui dénier un plaisir totalement légitime. C'est pour cette raison que je ne condamne pas de nombreuses choses que les programmeurs font, même si je n'apprécierais jamais de les faire moi-même. La chose importante est qu'ils sont en train de créer quelque chose dont *ils*

ont le sentiment que c'est beau.

Dans le passage que je viens juste de citer, Bentham nous donne effectivement quelque avis à propos de certains principes d'esthétique qui sont meilleurs que d'autres, notamment l'"utilité" du résultat. Nous avons quelque liberté pour choisir nos propres standards de beauté, mais ce qui est particulièrement bien, c'est lorsque ce que nous trouvons nous-même comme beau est perçu par d'autres comme étant utile. Je dois concéder que j'apprécie vraiment d'écrire des programmes qui font le plus de bien, dans un certain sens.

Il y a de nombreux sens selon lesquels un programme peut-être "bon", bien sûr. En premier lieu, c'est particulièrement bon d'avoir un programme qui fonctionne correctement. Deuxièmement, c'est souvent bon d'avoir un programme qu'il ne sera pas trop compliqué de modifier, quand il faudra l'adapter à une nouvelle situation. Ces deux objectifs sont atteints lorsque le programme est à la fois facile à lire et compréhensible par une personne qui connaît le langage approprié.

Une autre façon importante pour un programme produit d'être bon, c'est qu'il soit capable d'interagir harmonieusement avec ses utilisateurs, spécialement en se comportant agréablement malgré les erreurs fournies en données par les utilisateurs. C'est vraiment un art de concevoir des messages d'erreur suffisamment significatifs ou d'anticiper des formats d'entrées suffisamment flexibles qui ne sont pas sources d'erreurs de la part de l'utilisateur.

Un autre aspect important de la qualité d'un programme est son efficacité à utiliser les ressources machine. Je suis désolé de dire que de nombreuses personnes de nos jours condamnent l'efficacité des programmes, en nous disant que c'est de mauvais goût de s'en préoccuper. La raison de cela est que nous sommes dans une époque de réaction par rapport à une époque récente dans laquelle l'efficacité était le seul critère d'évaluation des algorithmes, et les programmeurs dans le passé avaient tendance à être si préoccupés par l'efficacité qu'ils produisaient du code inutilement compliqué ; le résultat de cette complexité non-nécessaire a été que les préoccupations à propos de l'efficacité ont chuté, notamment à cause des difficultés de debugging et de maintenance de ces programmes si compliqués.

Le problème effectif est que les programmeurs ont dépensé trop de temps à s'interroger sur l'efficacité des programmes aux mauvais endroits et aux mauvais moments ; une optimisation prématurée est la source de tous les dangers (ou du moins de nombreux d'entre eux) en programmation.

Nous ne devrions pas nous perdre dans les détails et être globalement dispendieux, mais nous ne devrions pas non plus penser à l'efficacité uniquement en termes de tant de pourcent de gain ou perte sur un temps d'exécution, ou un espace utilisé. Quand nous achetons une voiture, beaucoup d'entre nous ne sommes pas trop regardant sur une différence de 50 ou 100 $ sur le prix, alors que nous serions parfois capables d'aller dans un magasin particulier pour acheter quelque chose à seulement 25 ¢ au lieu de 50 ¢. Mon idée par rapport à ce problème est qu'il y a un temps et un lieu pour l'efficacité ; j'ai discuté de ce sujet dans mon article sur la programmation structurée, qui paraît dans le numéro consultable en ce moment des *Computing Surveys* [21].

Moins de facilités : davantage de plaisir

Une chose plutôt curieuse que j'ai noté au sujet de la satisfaction esthétique est que notre plaisir est significativement augmenté quand nous réalisons quelque chose avec des outils limités. Par exemple, le programme dont je suis le plus content est un compilateur que j'ai un jour écrit pour un mini-ordinateur primitif qui avait seulement 4096 mots de mémoire, 16 bits par mot. Cela fait que l'on se sent vraiment être un virtuose lorsqu'on réussit à faire quelque chose dans des conditions si drastiques.

Un phénomène similaire a lieu dans beaucoup d'autres contextes. Par exemple, les gens tombent souvent amoureux de leur Volkswagen mais rarement de leur Lincoln Continental (qui marche vraisemblablement mieux). Quand j'ai appris à programmer, c'était un passe-temps populaire que de faire un maximum de choses avec des programmes qui tenaient sur une seule carte perforée. Je suppose que c'est le même phénomène qui fait savourer aux enthousiastes d'APL leurs programmes "tenant-en-une-ligne". Quand nous enseignons la programmation de nos jours, c'est un fait curieux que nous ne saisissions pas la passion d'un étudiant pour l'informatique jusqu'à ce qu'il ait également eu un cours avec les "mains dans le cambouis" de l'expérience de la programmation sur mini-ordinateur. L'utilisation de grosses machines avec leur système d'exploitation bizarre ne semble pas occasionner une quelconque passion pour la programmation, du moins pas au début.

La façon d'appliquer ce principe pour augmenter le plaisir que les programmeurs ont dans leur travail n'est pas évidente. Les programmeurs gémiraient sûrement si leur chef leur annonçait soudain que leur nouvel ordinateur aura seulement moitié moins de mémoire que l'ancien. Et je ne pense pas que l'on puisse attendre de quiconque, même pas du plus dévoué des "artistes programmeurs", qu'il accueille une telle éventualité de gaieté de cœur, parce que personne n'aime perdre ses acquis sans bonne raison. Un autre exemple peut aider à clarifier la situation : les réalisateurs ont fermement résisté à l'introduction du cinéma parlant dans les années 1920 parce qu'ils étaient fiers à juste titre de la manière dont ils pouvaient faire passer des messages dans leurs films muets. Un véritable artiste programmeur pourrait bien ressentir l'introduction de matériels plus puissants de la même façon ; les matériels de stockage de masse font pâlir nos méthodes de tri à l'ancienne. Mais aujourd'hui, les réalisateurs ne veulent plus revenir au cinéma muet, non pas parce qu'ils sont fainéants, mais parce qu'ils savent qu'il est parfaitement possible de faire de très beaux films en utilisant la technologie améliorée. La forme de leur art a changé, mais il reste toujours beaucoup de place pour l'art.

Comment ont-ils développé leur compétence ? Les meilleurs réalisateurs de films à travers les années semblent en général avoir appris leur art dans des circonstances relativement primitives, souvent dans d'autres pays avec une industrie cinématographique limitée. Et dans les dernières années, les choses les plus importantes que nous ayons apprises à propos de la programmation semblent avoir démarré avec des personnes qui n'avaient pas accès à des ordinateurs trop grands. La morale de cette histoire, il me semble, est que nous devrions utiliser l'idée des ressources limitées dans notre propre enseignement. Nous pouvons tous tirer du bénéfice du fait d'écrire à l'occasion des programmes "jouets", quand les restrictions artificielles sont levées, ainsi nous sommes forcés de pousser nos facultés à leur limite. Nous ne devrions pas vivre dans l'habitude constante du luxe, puisque cela a tendance à nous rendre léthargiques. L'art d'attaquer des mini-problèmes avec toute notre énergie aiguisera nos talents pour les problèmes réels, et l'expérience nous aidera à être plus

heureux encore lorsque nous réussirons sur des équipements moins restreints.

Dans une veine similaire, nous ne devrions pas avoir peur de faire de "l'art pour l'art" ; nous ne devrions pas nous sentir coupable à propos de programmes qu'on écrit juste pour le plaisir. J'ai vraiment pris mon pied en écrivant une fois un programme en ALGOL d'une seule instruction qui faisait appel à une procédure interne d'une telle façon qu'il calculait le m-ième nombre premier, plutôt que le produit intérieur [19]. Il y a quelques années, des étudiants de Stanford étaient excités d'avoir trouvé le programme FORTRAN le plus court qui s'imprimait lui-même, au sens où la sortie du programme était son propre code source. Le même problème a été considéré dans de nombreux autres langages. Je ne pense pas que c'était une perte de temps pour eux de travailler là-dessus ; et Jeremy Bentham, que j'ai déjà cité plus haut, n'aurait pas dénié l'"utilité" de telles utilisations du temps [3, Bk. 3, Ch. 1]. "Au contraire," écrit-il, "il n'y a rien dont l'utilité ne soit plus incontestable. À quoi l'utilité peut-elle être attribuée, si elle ne l'est pas à quelque chose qui est source de plaisir ?"

Fournir de beaux outils

Une autre caractéristique de l'art moderne est l'insistance qu'elle place dans la créativité. Il semblerait que de nombreux artistes de nos jours ne se soucient de rien moins que de créer de belles choses ; c'est seulement la nouveauté d'une idée qui importe. Je ne considère pas que la programmation des ordinateurs pourrait ressembler à l'art moderne selon ce sens-là, mais cela m'amène à une observation qui me semble importante. Parfois nous sommes contraints d'effectuer une tâche de programmation qui est désespérément terne, ne nous permettant aucune échapattoire créative ; et à ce moment-là, quelqu'un pourrait très bien me dire, "alors, c'est beau la programmation ? C'est bien beau de crier que je devrais prendre un grand plaisir à créer des progammes élégants et charmants, mais comment suis-je supposé transformer tout ce bazar en travail d'art ?" Bon, c'est vrai, toutes les tâches de programmation ne vont pas forcément être amusantes. Considérons la "ménagère" qui doit chaque jour nettoyer la même table : il n'y a pas forcément de place pour la créativité ou l'art dans toute situation. Mais même dans de tels cas, il y a moyen de faire une grosse amélioration : c'est toujours un plaisir de faire des travaux de routine si vous avez de beaux outils avec lesquels les faire. Par exemple, une personne aura vraiment plaisir à nettoyer la table, jour après jour, si c'est une table merveilleusement bien ouvragée avec un bois de luxe.

Donc je veux adresser mes remarques terminales aux programmeurs systèmes et aux concepteurs de machines qui produisent les outils avec lesquels le reste d'entre nous travaillons. *S'il vous plaît*, donnez-nous des outils qui sont agréables à utiliser, spécialement pour nos travaux de routine, au lieu de fournir quelque chose avec quoi nous devons nous battre. S'il vous plaît, donnez-nous des outils qui nous encouragent à écrire de meilleurs programmes, en augmentant notre plaisir quand nous le faisons.

C'est très difficile pour moi de convaincre de jeunes gens que la programmation est belle, quand la première chose que je dois leur dire est comment taper "slash slash JOB égal etc, etc." Même les langages pour le travail de contrôle peuvent être conçus de telle façon que c'en est un plaisir que de les utiliser, au lieu d'être strictement fonctionnel.

Les concepteurs de matériels informatiques peuvent faire que leurs machines soient beaucoup plus

agréables à utiliser, par exemple en fournissant une arithmétique des réels qui satisfasse des lois mathématiques simples. Les possibilités offertes par la plupart des machines actuellement en vente rendent le travail d'analyse rigoureuse des erreurs difficile et sans espoir, mais les opérations proprement conçues devraient encourager les analystes numériques à fournir de meilleures sous-procédures dont la précision a été certifiée (cf. [20, p. 204]).

Considérons aussi ce que peuvent faire les concepteurs de logiciels. La meilleure manière de rester dans l'esprit d'un utilisateur système est de lui fournir des procédures avec lesquelles il peut interagir. Nous ne devrions pas rendre les systèmes trop automatiques, de telle façon que l'action part toujours derrière la scène ; nous devons offrir à l'utilisateur programmeur une possibilité de diriger sa créativité vers des canaux utiles. Une chose que tous les programmeurs ont en commun est qu'ils apprécient de travailler avec des machines ; alors gardons-les dans la boucle. Certaines tâches sont mieux faites par la machine alors que d'autres sont mieux faites par les humains ; et un système convenablement spécifié trouvera l'équilibre correct. (J'ai essayé d'éviter une automatisation maldirigée pendant de nombreuses années, cf. [18].)

Les outils d'évaluation des programmes sont un bon cas d'école. Pendant des années, les programmeurs ne savaient pas comment étaient effectivement distribués les coûts de calcul dans leurs programmes. L'expérience montre que presque tous les programmeurs ont une mauvaise idée de l'endroit effectif où se situent les nœuds d'engorgement de leurs programmes ; il n'est pas étonnant que les tentatives pour tendre vers plus d'efficacité se trompent, si on ne donne jamais au programmeur une estimation de la baisse des coûts selon les lignes de code qu'il a écrites. Sa tâche ressemble à celle d'un couple de jeunes mariés qui essaient de programmer un budget équilibré sans connaître le coût des items individuels comme la nourriture, le logement, l'habillement. Tout ce que nous avons fourni aux programmeurs, c'est un compilateur optimisé, qui fait mystérieusement quelque chose aux programmes qu'il traduit, mais qui n'explique jamais ce qu'il fait. Heureusement, nous voyons apparaître des systèmes qui accordent à l'utilisateur un peu de crédit pour son intelligence ; ils fournissent automatiquement une instrumentation des programmes et un retour à propos des coûts effectifs. Ces systèmes expérimentaux ont rencontré un franc succès, parce qu'ils produisent des améliorations mesurables, et spécialement aussi parce qu'ils sont marrants à utiliser, donc j'ai confiance sur le fait que ce n'est qu'une affaire de temps avant que l'utilisation de tels systèmes devienne une manière standard de procéder. Mon article dans *Computing Surveys* [21] discute de cela plus avant, et présente quelques idées d'autres manières selon lesquelles une procédure interactive peut améliorer la satisfaction des programmeurs utilisateurs.

Les concepteurs de langage aussi ont une obligation de fournir des langages qui encouragent un bon style, dans la mesure où nous savons tous combien le style est fortement influencé par le langage dans lequel il s'exprime. Le sursaut d'intérêt actuel pour la programmation structurée a révélé qu'aucun de nos langages existant n'est vraiment idéal pour gérer la structure des programmes et des données, et on ne sait pas encore de façon claire comment devrait être un tel langage idéal. Par conséquent, je regarderai attentivement les expériences de conception des langages qui seront faites dans les quelques prochaines années.

Résumé

Pour résumer : nous avons vu que la programmation des ordinateurs est un art, parce qu'elle applique des connaissances accumulées au monde, parce qu'elle requiert des compétences et de l'ingéniosité, et particulièrement parce qu'elle produit des objets de beauté. Un programmeur qui se voit inconsciemment lui-même comme un artiste appréciera davantage ce qu'il fait et le fera mieux. Ainsi, nous pouvons nous réjouir du fait que les personnes qui donnent des conférences d'informatique parlent de l'*état de l'Art*.

Références

1. Bailey, Nathan. *The Universal Etymological English Dictionary*, T. Cox, London, 1727, See "Art," "Liberal," and "Science."

2. Bauer, Walter F., Juncosa, Mario L., and Perlis, Alan J. ACM publication policies and plans. *J. ACM* 6 (Apr. 1959), 121-122.

3. Bentham, Jeremy. *The Rationale of Reward*. Trans. from *Théorie des peines et des récompenses*, 1811, by Richard Smith. J. & H. L. Hunt, London, 1825.

4. *The Century Dictionary and Cyclopedia 1.* The Century Co., New York, 1889.

5. Clementi, Muzio. *The Art of Playing the Piano.* Trans. from *L'art de jouer le pianoforte* by Max Vogrich. Schirmer, New York, 1898.

6. Colvin, Sidney. "Art." *Encyclopaedia Britannica*, eds 9, 11, 12, 13, 1875-1926.

7. Coxeter, H. S. M. Convocation address, Proc. 4th Canadian Math. Congress, 1957, pp. 8-10.

8. Dijkstra, Edsger W. EWD316 : *A Short Introduction to the Art of Programming.* T. H. Eindhoven, The Netherlands, Aug. 1971.

9. Ershov, A. P. Aesthetics and the human factor in programming, *Comm. ACM 15* (July 1972), 501-505.

10. Fielden, Thomas. *The Science of Pianoforte Technique*. Macmillan, London, 1927.

11. Gore, George, *The Art of Scientific Discovery*. Longmans, Green, London, 1878.

12, Hamilton, Willian, *Lectures on Logic 1.* Wm. Blackwood, Edinburgh, 1874.

13. Hodges, John A. Elementary Photography : *The "Amateur Photographer" Library 7.* London, 1893. Sixth ed, revised and enlarged, 1907, p. 58.

14. Howard, C. Frusher. Howard's *Art of Computation* and golden rule for equation of payments for schools, business colleges and self-culture.... C.F. Howard, San Francisco, 1879.

15. Hummel, J.N. *The Art of Playing she Piano Forte*. Boosey, London, 1827.

16. Kernighan B.W., and Plauger, P.J. *The Elements of Programming Style*. McGraw-Hill, New York, 1974.

17. Kirwan, Richard. *Elements of Mineralogy*. Elmsly, London, 1784.

18. Knuth, Donald E, Minimizing drum latency time. *J. ACM 8* (Apr. 1961), 119-150.

19. Knuth, Donald E., and Merner, J.N. ALGOL 60 confidential. *Comm. ACM 4* (June 1961), 268-272.

20. Knuth, Donald E. Seminumerical Algorithms : *The Art of Computer Programming 2*. Addison-Wesley, Reading, Mass., 1969.

21. Knuth, Donald E. Structured programming with go to statements, *Computing Surveys 6* (Dec. 1974), pages in makeup.

22. Kochevitsky, George. *The Art of Plano Playing : A Scientific Approach*. Summy-Birchard, Evanston, III, 1967.

23. Lehmer, Emma. Number theory on the SWAC. *Proc. Symp. Applied Math. 6*, Amer. Math. Soc. (1956), 103-108.

24. Mahesh Yogi, Maharishi. *The Science of Being and Art of Living*. Allen & Unwin, London, 1963.

25. Malevinsky, Moses L. *The Science of Playwriting*. Brentano's, New York, 1925.

26. Manna, Zohar, and Pnueli, Amir. Formalization of properties of functional programs. *J. ACM 17* (July 1970), 555-569.

27. Marckwardt, Albert H. Preface to *Funk and Wagnall's Standard College Dictionary*. Harcourt, Brace & World, New York, 1963, vii.

28. Mill, John Stuart. *A System af Logic, Ratiocinative and Inductive*. London, 1843, The quotations are from the introduction, §2, and from Book 6, Chap. 11 (12 in later editions), §5.

29, Mueller, Robert E. *The Science of Art*. John Day, New York, 1967.

30. Parsons, Albert Ross. *The Science of Pianoforte Practice*. Schirmer, New York, 1886.

31. Pedoe, Daniel. *The Gentle Art of Machematics*. English U. Press, London, 1953.

32. Ruskin, John. *The Stones of Venice 3*. London, 1853.

33. Salton, G.A. Personal communication, June 21, 1974.

34. Snow, C.P. *The two cultures. The New Statesman and Nation 52* (Oct. 6, 1956), 413-414.

35. Snow, C.P. *The Two Cultures : and a Second Look*. Cambridge University Press, 1964.

Virtual Heidelberg Laureate Forum 2020 - Dialogue : Tarjan / Knuth

Donald E. Knuth, Robert E. Tarjan

Introduction : Coming up next is another highlight of this virtual Heidelberg Laureate Forum. The two touring laureates, Donald Knuth and Robert Tarjan, will discuss topics based on questions previously submitted by the young researchers. Donald Knuth received the 1974 ACM A.M. Turing Award for his major contribution to the analysis of algorithms and the design of programming languages, and in particular, for his contribution to the art of computer programming, through his well-known book series. Robert Tarjan was awarded the 1983 Nevanlinna Prize and the nineteen-eighty-six ACM A.M. Turing Award, along with John Hopcroft, for fundamental achievements in the design and analysis of algorithms and data structures.

Bob and Don, take it away.

Robert Tarjan : Thank you, Peter. Let me just welcome everybody who is online, all the young researchers and everyone participating. You know, both of us are getting kind of long in the tooth now, but Don, you've always been one of my personal heroes, so it's a bit intimidating to engage with you in this. And it's a great honor and a great pleasure for me to have this opportunity to converse with you. You know, you've been an idol of mine ever since I got to Stanford as a first year grad student.

I think what we'll try to do here is maybe I'll ask Don several questions and then he can answer, ask me several questions and we'll try to get an interesting dialogue going. I have lots of questions from the young researchers. I apologize in advance that we probably won't be able to get to all your questions : so many questions, so little time. But let's get started here. Don, my first question to you is, can you tell us about your T-shirt ?

Donald Knuth : How clever of you to ask that question, though OK, I don't know if you can see it very well, but it says *Concrete*. But this is a special T-shirt. I don't know, maybe only two or three were ever made. It came out in 1989 when our book *Concrete*[1] was new. And I'm wearing it in honor of Ron Graham, my co-author, who died in July. And it was his daughter, Cheryl, who made these T-shirts for us at that time.

R. T. : It was a terrible loss to the field, his passing. Can you tell us...

D. K. : You mentioned Princeton and of course, he was using the text at Princeton that year and I was using it at Stanford.

R. T. : That was a beautiful book, *Concrete mathematics*. I actually had to teach an undergraduate course out of it. And I have to say it was a challenge because it's beautiful, but there's lots of advanced material in it.

Transcription of a video here https://www.youtube.com/watch?v=O5g4Zl8ppQA.
by Denise Vella-Chemla, 4.10.2020, work in progress.

1. *Concrete mathematics*, Ronald Graham, Donald Knuth, Oren Patashnik, Addison-Wesley, 1988.

D. K. : And we never could figure out what was difficult about it, but we knew that something.

R. T. : My second question, can you tell us a little bit about your personal journey and how you got into computer science and maybe why computer science, not mathematics, given that I first knew you as a professor of mathematics at Caltech ?

D. K. : Oh, yeah, well, that's true. When I got started, of course, there was no such thing as computer science and computer science... I think the first computer science department came out about 1965, Stanford was one of the very first.

And so, you know, I got my... I started in college in 1956. That's nine years before the computer science existed. And I started out majoring in physics. And then, I found out that the labs were too high for me. I couldn't do weldings, for example. It was terrible, all that voltage. And I had, well, I shouldn't want to go into a too long story, but it scared me that I couldn't see what I was doing and I couldn't do that.

So for a year, I took a math class that convinced me that I should really switch over. And so I became a math major. And there were five of us at that case at the time. And I got my bachelor's in math in nineteen sixty. Then I went to Caltech and studied mostly combinatorial mathematics with Marshall. So, and some reason decided Caltech to keep me on as a professor, so I stayed. I had a great time at Caltech and including meetings I don't know, maybe you were a freshman by the time I left, my last year was 1968 or somewhat.

I mean, the summer of 68 is when I took a year off to work on national service. But anyway, by that time, I realized that computer science was really my version of my career path because I had seen that I was editing several journals at the time from math department at Caltech, but I was sitting in lots and lots of lectures about mathematics, where I was sitting in the back row saying, so what ?

So about the lectures that I was hearing and all the correspondence I was having about computer science, which was very exciting for me. So I decided that I should make one move in my life, namely to be a full professor somewhere. And I had... There were four major places I was deciding between. One was with Stanford, or one with Berkeley, and one was Harvard, all in computer science. And the fourth was Caltech, where I would stay in mathematics.

And so anyway, I came in Stanford. And then shortly after, *you* came as a first year student. And one of the reasons I went to Stanford was I thought at Caltech, we couldn't really... we do much in computer science. And you came in my office and, you know, and I was your advisor for classes. And the next thing I took, Stanford got this great curriculum. So I listed all the courses that we have and you said, oh, you've already taken those courses from some visitor who came to Caltech. So that's how I got into the field. And basically, I believe it's because I realized even when I was an undergraduate, that there was something about the way my brain had evolved by that time, that computers were really resonating with me. And I really loved everything, all these connections. I just, I was born to be a geek and maybe at least by the time I was 16, I was...

R. T. : So, I have several questions, but I'll just ask one now and then turn it over to you. When did the idea for the book come to you ? When did you get started on that project ?

D. K. : Yeah, I was a second year grad student at Caltech and it was January 1962. I entered in the fall of 1960 and my favorite textbooks had been published by Addison-Wesley, when I was an undergraduate, my calculus books, my physics books, some books, a number of various things that I liked to read and an editor took me after lunch and he said "Don, we'd like you to write a book about how to write a compiler." and gosh, I had always enjoyed writing when I had been working for campus publications for example. And so I was thrilled by this idea and I came home and on a sheet of yellow paper that I still have somewhere, I wrote down the title of twelve chapters I thought ought to be in this book. Chapter 12 was about compilers and the other eleven chapters were preparing for compilers and I'm so far I'm up to chapter 7 now from that list that I started in 1962.

R. T. : Well I'm sure we'll return to the topic of the book but maybe now I'll turn over to you and let you ask a few questions.

D. K. : Yeah, okay, well, did you learn you were a geek every time ? How do you get the bug to come in ?

R. T. : I'll go back a little farther in my personal history. Way back. My father ran a state mental hospital in California and he was interested in doing research on reasons for the developmental disablement and he was fairly well known. Linus Pauling who was at the Caltech at the time came over to our house and they were doing a joint research project and Dr Pauling left the Caltech catalog with me so I had that scientific interest. When I got into public school, this was back in the days when the California public schools were still really strong, I had an amazing math teacher. So I got bug to study mathematics, I read Scientific American columns by Martin Gardner. I had an opportunity when I was in high school to do a little bit of programming and I worked when I was in college, doing computer programming. So my plan was always to do mathematics. I went to Caltech as an undergraduate but unfortunately I never had a chance to take a class from you. Caltech at the time was all about physics, the math department was a little bit strange but wonderful. Brilliant combinatorialists, Herbert, Ryser, Marshall Hall, so I had a great time. I have to say being an undergraduate at Caltech was probably one of the biggest challenges of my life. It was extremely intense. Then deciding on graduate school, I applied to a couple of math departments and a couple of computer science departments and ended up going to Stanford in computer science and never regretting it because it gave me an opportunity to use mathematics in something where you could actually see, in a concrete way, the effects of algorithms and algorithm analysis and so on. So that's the short version of the story.

D. K. : Good and we're lucky that John Hopkins had a sabbatical so he came to visit...

R. T. : That was an extremely fortunate event, yes, that is certainly true. It was early days, I mean, Alan Kay was there. Stanford in those days was a very special place. Alan Kay gave a talk already in the HLF [2] and he said something like "Find a great research community.". Stanford in those

2. HLF : Heidelberg Laureate Forum.

days in computer science was a great research community. And of course, there was involvement of people from Berkeley too : Dick [3] Karp and Waller, for grad students, it was quite an amazing place.

D. K. : We might remind to the people that although it was 1970, it was really the 60s. *(R. T. laughs)*. I mean as I recall, you lived in Berkeley, in the commune and you commuted every day, and as I sort of remember that I was very worried about you because you were proving theorems in your head as you were driving the freeways and I was afraid you'd get into a wreck. *(R. T. laughs)*.

R. T. : Thank you for that concern. That was when I got to be an assistant professor at the Stanford. That was after my grad student days.

D. K. : All right, well, I got the time mixed up and then a few years ago but... So but anyway, there was a lot going on in those days outside, yeah, outside of college as well. And I guess, there always is, but those days were pretty. I mean students were setting fire to campus buildings.

R. T. : It was the height of the Vietnam war and there were protests against the war and there was the whole hippie movement and it was a very interesting time. Many of us were very idealistic somehow. Perhaps a little naive, or a little *too* idealistic.

D. K. : Well thank godness people ideas are realistic I see. So now the thing I remember the most is that basically in the 70s you revolutionized computer science at least from my standpoint which is you know trying to write a textbook. You and Shannon in the 70s were the two people who most upset my table of contents *(R. T. laughs)* for what I would have to do and the reason was that it was, you know, you were doing things that I had absolutely no conception of in 1962 when I set up this on the plan for the books and especially, this was the first time, this now seems, maybe, it was obvious but boy, it was so unusual to find something like a deep algorithm, some property of a brand new way to organize information in computer that what... it's way out there, everything hung together, blown away by the fact that somebody could actually prove theorems about complicated data structures and before that, there were just data structures you could easily explain to somebody in ten minutes or maybe an half an hour but all of a sudden, you came up with these methods that really take a lot of proof and it's almost amazing that they work at all. I remember my secretary Phyllis, she said that you visit here every once in a while, so you were also doing research while you were sitting in her house, I guess, also thinking about these things but I mean, can you say something about the origin of those ideas that you would even conceive of having a deep data structure ?

R. T. : Well, that's a hard question to answer : how does one figure out how to do research. One tries... one has to be intensely curious, one has to explore the design space, and the thing is computers kept getting faster and faster. Moore's law is a gift to theoreticians as well as to practitioners, it creates the opportunity to think about things on a higher level and the idea of analyzing algorithms from a bit of a fuzzy point of view that is ignoring constant factors really opened up the field and thinking about actually proving time bounds which really wasn't done until we started doing it I think that was important because it kind of drove. If you have a goal to make an algorithm faster then you're forced to strip away the irrelevance and figure out what are the critical parts that are involved in the computation and use those in exactly the right way so...

3. Richard M. Karp

D. K. : So that was your initial work but because of this goal that you had about you know, trying to get linear time or were you trying to just get cubic time or something I mean, I'm thinking, it was the linear time challenge the thing that did it first for you ?

R. T. : Well, how did I come to start really working hard on algorithms ? I took a Lisp programming course from John McCarthy who was another A.M. Turing award winner, unfortunately, and again, great Stanford environment, but his final project we were supposed to write a Lisp program. One of the options was to write a Lisp program to do test graphs for planarity. I've been interested in planar graphs because I was interested in the four colors, the now theorem problem. I was trying to prove this myself in high school using computational methods but I didn't have enough information and enough computational power I can app and solve this problem in 1976 using exactly this techniques. So the mathematical criterion for planarity is the famous Kuratowski criterion (a graph is non-planar if it contains a complete graph on five vertices or a complete bipartite graph on two sets of three vertices). So everybody else trying to implement the Kuratowski test but I managed to find a paper by Shimon Even and Lempel and Cederbaum actually which gave an algorithm which essentially constructed an abstract representation of embedding. So I implemented that algorithm, it turns out to be quadratic time, they didn't state that but that's what it is and I managed to do really well on the project. So then I thought if quadratic time, why not faster ? And John Hopcroft came and we started talk about graph algorithms and exploring the potentials of depth first search and things led to things. When you have a tool that works effectively, you try it on any problem you can think of. So let me turn it back to you, I'll go ahead, back to case western. I seem to remember a famous story about you being on the basketball team. Can you say something about that ?

D. K. : Well I was the manager, I was a scorekeeper and if somebody wants to look this up there's a one minute video so just google youtube for, it's called the electronic coach [4] and I worked out a way to keep the basketball statistics more that had been done before and I came up with a formula for each player as to how much they contributed to that game that took into account the shots they missed and the fumbles and the rebounds that they got and things like this all into one glorious formula I don't believe the formula anymore but at least I had this formula and so it wasn't just the points that you scored but everything that you did went into this and I would punch cards after every game and compute these numbers and then the coach seemed to like them.

R. T. : That was back in the days when we were programming on IBM machines and you had to input things on punch cards on in great massive boxes of punch cards...

D. K. : And you could see it in that video and also, I wrote a book on, actually, I'm glad that I lived long enough to write this book, it's called *"Selected papers on fun and games"* and one of the chapters in there gives the whole story about this including the reviews in the newspapers and like how it was carried on, showing everything like this, Newsweek magazine. *(D. K. laughs.)*

R. T. : So it seems like you were doing quantitative sports analysis thirty years ahead of its time. Did you have ever open an opportunity or interaction with some of these people who are doing it now for real with professional sports teams ?

4. https://www.youtube.com/watch?v=dhh8Ao4yweQ.

D. K. : No, no, that was when it was easy and fun but I was a junior. But anyway, it just shows again that computers were totally involved, you know, with my life all the way through...

R. T. : Computer science. It's been said that any field that has science in its name is not a science. So I might ask you "is computer science a science, a branch of engineering, a branch of mathematics, an art ?", but let me ask it in a more personal way, especially since you've titled one of your famous books *The Art of Computer Programming*. You see yourself as an artist, a scientist, a mathematician, an engineer, a philosopher, some combination ?

D. K. : Yeah, Okay, that was the subject of my turn talk in 1974 [5]. So I can't do any better than ask people to plug back that button but to shorten it, I looked up about art and science and I found, you know, lots of books had there in there in the title, I quote a whole bunch of them in that at the time and I realized that the word art stands for not only fine arts but also things that are artificial and basically it comes from the greek word $\tau\varepsilon\chi\nu\varepsilon$, in german *Kunst* and so on now. It meens something that's made by human beings as opposed to being present in nature. And science is the study of knowledge and organization of things and so, in a sense, I can tell you a short definition that science is what we understand well enough to explain to a computer and art is everything else. And as science advances, we learn more and more about whatever field we're studying, but then, as we learn more and more about it, our brains figure out and keep a few jumps ahead, and that's the art.

R. T. : So that raises a very interesting question because artificial intelligence has been reborn now as deep learning neural nets and some people are worried that the singularity is coming where computers will evolve beyond us in intelligence. What role do humans have in this and should we be worried about this, what do you think about this new version of artificial intelligence ?

D. K. : Yeah, well I don't know how long it'll be before they rename our department as the department of machine learning but I'd rather talk about almost everything else but I'm seriously worried about the potential for weapons especially. This video, well, what's the name... anyway, it's such an important video but I put it out of my mind I guess, Steve Russell put that important thing to show that it's not that far out that we could have little drones targeting our enemies and nobody being able to do it stop and it wouldn't cause that much for terrorists to do this, not to mention people who don't like other people in their own government or in some other country so...

R. T. : Boy, it seems like technology. There is always the fact that they are good uses and bad uses and it's up to us as human beings to take use of technologies in the right way, it's a big challenge.

D. K. : Yeah we have to understand it, that's for sure, right.

R. T. : Well, do you think that we, as theoretical computer scientists or as mathematicians have something to contribute to the new field of artificial intelligence ? Should we ? What's our role here ? I know some of my colleagues have jumped in trying to... and one big challenge seems to be neural nets solve problems, do things, accomplish tasks but nobody has a clue to how they do it. There's no explanation, which is kind of anti-scientific, is that right ?

5. https://amturing.acm.org/award$_w inners/knuth_1$013846.cfm.

D. K. : That's right, I apologize for saying Steve Russell, I meant Stuart Russell, of course, and he's given the most deep thought to this problem of how to prepare early on for whatever. But the unfortunate thing is that his solutions are always assuming that human beings are rational. And I'm beginning to wonder more and more about that every day because people are not being rational. And that is so we have to figure out a way to save the world with the irrationality.

So, but anyway, as you say, people like you and me have things that we're good at and we aren't necessarily good at everything. And so I figured I've got a few more years to live I'm gonna spend my time about the things that I can do more uniquely than things that I did, I'm not sure anybody can do.

R. T. : That's the conclusion I came to also. There's little time and let's use our abilities that we have in the best way we can.

This is perhaps a related question, computer science, unlike mathematics, is still a very young field among many. You got into it at the very beginning and I was only more or less a decade later. What does this mean ? I mean, how do you see computer science going forward ? How has it changed in your lifetime ? Do you have directions you would like to see it go ?

D. K. : Maybe these questions, but I'd like to turn back to you. It's not clear whether computer science is a subset of mathematics or mathematics is a subset of computer science and different days, I could argue either way, but actually I believe that they're different. And although I've been part of both worlds at different times, I consider myself a lapsed mathematician now. But there are days when I say, oh, today, I'm going to think like a mathematician. And sometimes I can solve a problem by taking it by having my mathematician's cloak on for a day and then I put on my computer scientist cap and I work on it from another point of view. And I can strongly feel the difference when I'm operating in one mode or the other.

I argued this with Bill Thurston, though, and he couldn't see any difference. So I don't know. But to me, I think that computer science and mathematics are the two sciences known, so far, bodies of knowledge known so far that are not based on nature.

R. T. : They are created by human beings.

D. K. : We get to make up our own ground rules. You know, we design a universe that we're going to study and physicists don't have that, but string theorists maybe. But, you know, chemists, biologists, they deal with nature. But mathematical designs are the parts where we're studying things that are detached. But created. These idea that they were really out there all the time or not ? But anyway, that separates it from the other field.

And I see that, as years go by, I think mathematicians are starting to get it, the way I understand computer science. And computer scientists are a lot of times more interested in Wall Street than in science. As you know, a lot of our staff students have gone off and become hedge fund people or something instead of advancing, you know. So when we were in the 70s and there was a question

whether universities would survive and so on, it was clear to me that if Stanford would be burned down, I would still gather a bunch of students somewhere and we would go somewhere and we would want to talk about computer science.

And I've never been I never said, oh, what's a good way to have an easy life for you or to make a lot of money or something like this, start a company, have all these stock options and then by the end of summer. No, the last thing on my list of motivation. On the other hand, I know other people that did their best work because they were excited by different things that excite me. I'm not saying my way is the only way, I guess.

R. T. : But let me give it back to you and give you the opportunity to ask me a few questions.

D. K. : Yes. What do you think about the motivation that keeps you going ? Like, I think you're finishing a book now ?

R. T. : starting a book ! Well, the long delayed book project, I make no commitments. But yeah, in this process, I discovered the beauty of TeX. I have to say, I waited too long to try to learn this. We'll see how it goes. *(D. K. laughs.)* But, you know, I had the same feeling as you as I was developing these algorithms in the 70s and the 80s. The field is not yet stable enough to try to capture information in the book. But at this point now, finally, this is one of the positive things about what has happened with the epidemic, the pandemic.

I've got a lot more time because I'm not on an airplane. So I actually started doing some writing and it requires large blocks of time, obviously, and I think there's enough stability that it's actually maybe possible to do it.

D. K. : So let me... Isn't it true that you started writing a monograph and you left the whole manuscript in a metro station once ?

R. T. : there was a briefcase that I left in a metro station in Paris, yes. Containing lots of useful stuff I used to write on pencil and paper. Now it's on the computer and most of it's stored in the cloud, fortunately. So I guess I'm trusting to a particular platform as opposed to you who are interested in your local environment. I hope your house is secure, Don, because we wouldn't want to lose that treasure trove of information.

D. K. : Well, I think it's OK. Yeah. When you spoke about plans for our dialogue, you mentioned that you had some concerns about publications and conference culture and things like that. Can you...

R. T. : Yes, yes. Yes. This is a difference, I think, between most scientific fields. And I would include mathematics there as well and computer science as it has evolved. It seems now like we're putting all our effort into conference publications fast turnaround. And if we are a mathematical field, maybe it works for the experimentalists, but for the theoreticians, it's kind of amazing to me. I'm sort of old. I mean, I'm sort of a lapsed mathematician in the same way that you are.

And it's kind of amazing to me that we make progress in spite of the fact that there are plenty of mistakes, there's plenty of kind of half baked publications with great results, but not fully worked out. And it kind of breaks my heart that people don't take the time to go back and fill in the details and investigate further. So it seems to be an inevitable trend in our field, and it's getting even a little bit worse with the advent of platforms such as arxiv, where people can post almost anything, some half-refereed at best.

So I am concerned about it. I think it's very important to try to get everything into a refereed journal, if possible. I try to encourage my students to do that, but it is a challenge. It's gotten much more about publish or perish as it was, back when I was a student.

D. K. : I imagine that's because there are thousands of people now when there were only dozens when you and I were young but boy, well, I know enough about the history that there's always been problems with quality control in publication. But I remember last year, I got really angry when I heard about what, you know, I looked at a paper and I thought it would solve the problem that I thought I would never see the solution of in my lifetime and I thought it was a grad solution but the problem involved is a generalization of knights tours and the guy got the most insulting referee reports when he submitted the article to a journal and I showed this, you know, and essentially, in those high pure academics for saying, nobody would ever stoop to write a paper about something that the generalization of a nice tool and so, there is no consideration for taste and I can't say that my taste is better than another. I can look at papers and say this is trash and I can look at something else and I say this is beautiful. I have no algorithm although, I don't take machine learning let to solve their problem.

R. T. : That's the flip side of the coin : I mean, great papers were rejected by narrow-minded reviewers and the greatness only emerges later on perhaps.

D. K. : Right. Plus the... So I finally found a good referee for the guy, but you know, he worked for a journal that... you know... screws the universities by charging too much for this journal. So he didn't want to publish in that journal. So he left it on the arxiv for the moment. You know, in the journal as the editor of the journal of algorithms, I researched what was being done by the patient publishers who are in most cases using our services for free to do all the refereeing and editorial work, but they are not interested in anything but the bottom line and they were making huge profit margins they import, the academic press had been bought out by somebody who had been bought out by somebody else who had been bought out by elsewhere and so on and finally the publishers were making arrangements with universities, and swearing the universities to secrecy so that we couldn't even find out what libraries were being forced to pay to the journals, and that was the last draw for me and I reported all those things to our editorial board of our journal on algorithms and we resigned our percent to 100 and we started the ACM transaction on algorithms...

R. T. : Is that the solution then : more support for professional societies, non profit organizations ?

D. K. : Yeah, it seems to me that in the professional societies, we have some control over that. I don't know what your opinion is... ?

R. T. : I concur, I concur...

D. K. : Tim Gowers has...

R. T. : I'm going to change the subject, sorry, go ahead... I'm going to change the subject a little bit. Not only are you a serious computer scientist. But you're a serious organist player and composer also. Can you tell us a little bit about what you've done and what you're up to in this domain ?

D. K. : Okay, so anyway, you were at my 80th birthday party, one of the great friends who came there.

R. T. : And I'd been in a T-shirt it would have been that one.

D. K. : *(D. K. laughs.)* Okay so I'm 82 now but on my 80th birthday, we had a celebration in Northern Sweden where there is a particularly wonderful pipe organ and when I heared about that organ, I figured that... I had also planned, since the 60's, I wanted to write some kind of a major composition for pipe organ. It was a dream that I had all these years until finally I got to be, I don't know, when I was seventy five years old, I thought if I'm ever going to do this, I should better start now. So on a day I was visiting Vienna, and I went to the best music store in Vienna and I bought some blank music papers from the same store that Beethoven had bought stocks, and Brahms, and so on, so I bought from and I figured that wouldn't hurt until I brought that with me and I started writing my piece. And I worked hard off and on for five years, and came up with a very strange piece that I have to say I do like... I am glad that it came up the way it did. So we had this glorious Premiere. You can see it on youtube google sent by a team of people who captured it, they had the best state of the art, several dozens of cameras. So we have a 360° with two different cameras and lots of audios, the best mic team for surround sound and so on. So we have many terabytes of data from that Premiere performance. And I'm hoping that computer scientists are looking for a thesis project is going to make a beautiful virtual reality, things of this so that people can watch it and select as they're watching it what they want to see because we have all the bits captured and Stanford arxiv has it all in the digital form and it's available to any researcher who wants it.

R. T. : I have to say that watching it in person was an overwhelming experience and it would be good to relive.

D. K. : Yeah so, it's divided in twenty-two parts each of which is about five minutes long and you can watch one part at a time if you go to my website you can find the playlist it's on youtube [6]. But then the canadian premiere took place and it was a wonderful performance in Waterloo, two years ago now, and the people there made a great video where you can watch you know in a different way. And then last year in Czechia in Brno [7], and again they have made a tremendous video of the performance. So now there are three excellent sources like if anybody wants to see if they like. I call it high bandwidth music because I was reacting to the lot of modern music... I don't know. They start out with some idea and then they hold it until the audience gets it and then they move on to the next idea and so on. But I wanted to go back to the way Mozart would do it and come

6. https://www.youtube.com/playlist?list=PLvixIGKr5sJffdfwecygYqhXsgz-EBCC8.
7. https://www.youtube.com/watch?v=wk7dEKMPP68

up with ideas faster than then we can take in so we have to listen to it twice and three times and we get more each time and... So they made these really nice, they captured in three beautiful ways and I'm so glad that ought to be music of the world and maybe it'll be popular and maybe it'll be a dud but at least, it satisfied me that, you know, I had this as one of my life's goals, it's hard to explain but if you have talked to me two years before I finished it and somebody would told me I only had two years to live I had to choose between finishing that piece or finishing out a computer program like... I somewhat would have chosen that piece although I should have chosen the other computer part because I have no right to write music.

R. T. : I don't see why not. So let me ask you for the young researchers benefit. Do you have any advice to people wanting to go into computer science as it is now. Do you have any advice for students or for mentors of students ?

D. K. : Yesterday, when the question was asked to Leslie Lamport and to Tony Hoare, Lamport's answer was so wonderful, he said "Writing.", what to do ? "Writing.". And I guess like monopolizing this too much away from you here but it's the way for me to tell you "Finish this book we're working on." *(R. T. laughs)* because all my life, I found that what I was doing was a sort of convex combination of mathematics and writing. But writing was always very important and keep trying at all that's why I write so many computer programs now.

R. T. : I certainly have to agree with that, right right right, rewrite, but it's a hard challenge. Ideas are worthless if you cannot communicate them with other people and the best way to do that still is to get them down in old-fashioned paper, I would say.

D. K. : And the other thing that came through from that dialogue yesterday which I would say in different way is that don't be too much influenced by trendy stuff. Don't write a paper because you have to write a paper or because you think that you have to impress people about something that you aren't personally interested in but you're trying to make an impression. That's the worst reason to write a paper ; the model that I like to think of this is Euler whose papers were always very impressive but because his attitude was he wanted to tell the people what was impressing him.

R. T. : Yes, I think Alan Kay who also gave a talk at this forum, he had this notion of outlaw ideas getting squashed but... You have to follow your own path somehow, you have to figure out what your own path is and follow it. The best students I've had came in with or ended up with their own ideas that they developed and actually, Pat Hanrahan earlier in the week said I'd never give my PhD students a thesis topic. I require them find a topic for themselves, now, that's kind of a rigorous approach but that's part of what it is to do a PhD, it's to learn how to ask the right questions, it's much more about the right questions than the answers to other people's questions.

D. K. : Here here here and in fact asking questions is something that you've just demonstrated that you know how to do well and *(R. T. laughs.)*

R. T. : Let me ask one more, we are almost out of time here but I noticed that according to wikipedia and this is a question from one of the young researchers, you're known for your scientific jokes, I wonder if you can tell us one of your favourites ?

D. K. : Okay, so *(D. K. shows one of his books open at the right page).* Take a look at *Concrete mathematics* bibliography, reference n° 44, T. Brown, *Multivariable subpolynomial waffles which do not satisfy the lower regular q-property piffles in a collection of 250 papers on waffle theory dedicated to R.S. Green on his 23rt birthday...* And then I have a marginal note. In the margin, it says *Such papers are not cited in this book. (R. T. laughs.)* And that's why I wrote *Concrete mathematics* and you look in the index and you see who T. Brown was and he was Trivial Brown, it's fun to look at the different translations of *Concrete mathematics*.

R. T. : I seem to recall also that your first publication was in an unusual magazine. Can you remind us what it was ?

D. K. : Just a second. *(D. K. stands up and go to look for the magazine in question.)* Okay, someday, see if you can get a hold of this book *Selected papers on fun and games*. It not only talks about basketball but it talks about my first publication which was in Mad magazine, there I am *(D. K. zooms in for us to see well the Mad magazine.)*.

R. T. : So if you want to be done, Knuth, young researchers find a way to get your first article published in Mad magazine. So many questions, so little time, let me thank you Don for a wonderful dialogue and I'm thrilled to have participated in it and thanks to HLF and all the organizers.

D. K. : Okay, I guess we ought to cut, bye-bye.

P. : Thank you bless, very much for the insight and I'm sure your discussion was very inspirational for the young researcher. We really appreciate you both taking the time. Take care and thank you very much.

D. K. AND R. T. : Thank you Peter.

Forum virtuel des lauréats d'Heidelberg 2020 - Dialogue : Tarjan / Knuth

Donald E. Knuth, Robert E. Tarjan

Introduction : Nous allons entendre un autre moment fort de ce forum virtuel des lauréats d'Heidelberg. Les deux lauréats de cet échange, Donald Knuth et Robert Tarjan, discuteront de sujets basés sur des questions soumises auparavant par de jeunes chercheurs. Donald Knuth a reçu la récompense de l'ACM A.M. Turing en 1974 pour ses contributions majeures à l'analyse des algorithmes et sa conception de langages de programmation, et en particulier pour sa contribution à L'art de la programmation des ordinateurs, à travers sa série de livres très célèbre. Robert Tarjan a été recompensé par le prix Nevanlinna en 1983 et a reçu la récompense de l'ACM A.M. Turing en 1998, avec John Hopcroft, pour ses réalisations fondamentales dans la conception et l'analyse des algorithmes et des structures de données.

Bob et Don, c'est à vous.

Robert Tarjan : Merci, Peter. Permettez-moi de souhaiter la bienvenue à tous ceux qui nous suivent en ligne, tous les jeunes chercheurs et tous ceux qui participent. Vous savez, tous deux nous commençons à avoir de la bouteille, mais Don, vous avez toujours été l'un des mes principaux héros, et c'est un peu intimidant de m'engager là-dedans avec vous. Et c'est un grand honneur pour moi d'avoir cette opportunité de converser avec vous. Vous savez, vous avez été un idole pour moi depuis que je suis entré à Stanford pour ma première année d'études.

Je pense que ce que nous allons essayer de faire ici est peut-être que je poserai à Don quelques questions et alors, il peut répondre, me poser quelques questions et nous allons essayer d'avoir ainsi une conversation intéressante qui va s'établir. J'ai beaucoup de questions des jeunes chercheurs. Veuillez excuser par avance le fait que nous ne pourrons vraisemblablement pas prendre toutes les questions : tant de questions, si peu de temps. Mais commençons ici. Don, ma première question est "Pouvez-vous nous en dire plus au sujet de votre T-shirt ?"

Donald Knuth : Comme c'est intelligent à vous de poser une telle question, bien que, ok, je ne sache pas si vous pouvez bien le voir, mais ça dit *Concretes*. Mais c'est un T-shirt spécial. Je ne sais pas, peut-être seulement deux ou trois ont été produits. C'était en 1989 quand notre livre *Concretes*[1] était tout nouveau. Et je le porte en l'honneur de Ron Graham, mon co-auteur, qui est décédé en juillet. Et c'est sa fille, Cheryl, qui avait confectionné ces T-shirts pour nous à ce moment-là.

R. T. : Cela a été une terrible perte pour notre communauté que son décès. Pouvez-vous nous dire...

D. K. : Vous avez mentionné Princeton et bien sûr, il utilisait le texte à Princeton cette année-là et je l'utilisais à Stanford.

R. T. : C'était un beau livre, *Mathématiques concrètes*. J'ai eu à enseigner un cours de premier cycle à partir de ce livre. Et je dois dire que c'était un défi parce qu'il est beau, mais il y a beaucoup

Traduction d'une video ici https://www.youtube.com/watch?v=O5g4Zl8ppQA.
par Denise Vella-Chemla, 4.10.2020, travail en cours.

1. *Mathématiques concrètes*, Ronald Graham, Donald Knuth, Oren Patashnik, Addison-Wesley, 1988.

de contenu avancé dans ce livre.

D. K. : Et nous n'avions jamais imaginé à quel point ce serait difficile, mais on s'en doutait un peu.

R. T. : Ma seconde question, pouvez-vous nous en dire un peu plus à propos de votre parcours personnel et de la manière dont vous êtes venu à l'informatique et peut-être pourquoi la science des ordinateurs, et non pas les mathématiques, étant donné que je vous ai d'abord connu comme professeur de mathématiques à Caltech ?

D. K. : Oh, oui, eh bien, c'est vrai. Quand j'ai commencé, bien sûr, il n'y avait pas d'informatique puisque l'informatique... Je pense que le premier département d'informatique a démarré en 1965, Stanford était l'un des tout premiers.

Et donc, vous savez, j'ai obtenu mon... J'ai commencé à la faculté en 1956. C'est neuf ans avant que l'informatique n'existe. Et j'ai commencé avec une majeure en physique. Et alors, j'ai trouvé que les travaux dirigés étaient trop durs pour moi, je ne pouvais pas faire des soudures, par exemple. C'était terrible, tous ces voltages. Et j'avais, bon, je ne voudrais pas me lancer dans une histoire trop longue, mais ça m'effrayait de ne pas pouvoir voir ce que je faisais et je ne pouvais pas faire ça.

Alors pendant une année, j'ai suivi un cours qui m'a convaincu que je devrais vraiment changer de voie. Et alors, j'ai pris une majeure en mathématiques. Et nous étions cinq dans ce cas à ce moment-là. Et j'ai obtenu mon diplôme en maths en 1960. Ensuite, je suis allé à Caltech et j'ai étudié principalement les mathématiques combinatoires avec Marshall. Et alors, quelques raisons ont fait que Caltech a pris la décision de me garder comme professeur, du coup, je suis resté. J'ai eu de belles années à Caltech et même des rencontres, je ne sais pas, peut-être étiez-vous étudiant aux alentours du moment où je suis parti, ma dernière année là a été 1968 ou à peu près.

Je veux dire, l'été de 68, c'est l'année où j'ai dû arrêter pour faire mon service national. Mais en tout cas, à ce moment-là, j'ai réalisé que l'informatique était vraiment l'idée que je me faisais d'une carrière parce que j'avais vu que j'effectuais quelques services à ce moment-là pour le département de math de Caltech, mais j'assistais aussi à de très nombreux exposés à propos de mathématiques, où je m'asseyais au fond de l'amphi et où je me disais, et alors quoi ?

Donc j'allais suivre des exposés et j'échangeais beaucoup de correspondance à propos de l'informatique, ce qui était très excitant pour moi. Aussi je décidai que je ne me déplacerai qu'une seule fois dans ma vie, notamment pour être un professeur complet quelque part. Et j'ai obtenu... Il y avait quatre postes principaux entre lesquels je devais me décider. L'un était Stanford, l'autre avec Berkeley, et un autre était Harvard, tous en informatique. Et le quatrième c'était Caltech, où je resterai dans le domaine des mathématiques.

Et donc en tout cas, je suis venu à Stanford. Et ensuite, peu de temps après, *vous* êtes venu comme étudiant de première année. Et l'une des raisons pour lesquelles je suis allé à Stanford était que je pensais, lorsque j'étais à Caltech, que nous ne pourrions pas vraiment... nous faisons davantage en informatique. Et vous êtes venu dans mon bureau, et j'ai été votre professeur pour certains cours.

Et la chose suivante que j'ai prise, Stanford avait cet extraordinaire curriculum. Alors j'ai fait la liste de tous les cours que nous avions et vous avez dit, oh, vous avez déjà reçu ces cours d'un des visiteurs qui est venu à Caltech. Et c'est ainsi que je suis entré dans le domaine. Et de façon basique, je crois que c'est parce que j'ai réalisé, même lorsque j'étais en premier cycle, qu'il y avait quelque chose dans la manière dont mon cerveau avait évolué à ce moment-là, qui faisait que les ordinateurs résonnaient vraiment en moi. Et j'aimais vraiment tout, toutes ces connexions. Je veux juste... Je suis né pour être un geek et peut-être qu'à ce moment-là, j'avais à peu près 16 ans, j'étais...

R. T. : Alors, j'ai plusieurs questions, mais je vais juste vous en poser une et ensuite, ça sera votre tour. Quand l'idée du livre vous est-elle venue ? Quand avez-vous démarré ce projet ?

D. K. : Oui, j'étais en seconde année à la faculté à Caltech et c'était en janvier 1962. J'ai commencé à l'automne 1960 et mon livre préféré avait été publié par Addison-Wesley, quand j'étais en premier cycle, mon livre de calcul, mon livre de physique, d'autres livres, un certain nombre de choses diverses que j'aimais lire et un éditeur me prit à parti après le repas et me dit "Don, nous aimerions que vous écriviez un livre à propos de la façon dont on écrit un compilateur." et mon dieu, j'aurais toujours aimé en écrire un quand j'avais travaillé sur des publications pour le campus par exemple. Et alors, j'étais content à cette idée et je rentrais à la maison et sur une feuille de papier jaune que j'avais quelque part, j'écrivis les titres des douze chapitres que je pensais devoir être dans ce livre. Le chapitre 12 était à propos des compilateurs et les onze autres chapitres étaient une préparation pour les compilateurs et au jour d'aujourd'hui, j'en suis au chapitre 7 de cette liste que j'ai commencée en 1962.

R. T. : Bien, je suis sûr que nous reviendrons à ce sujet du livre mais peut-être maintenant, puis-je vous laisser poser quelques questions.

D. K. : Oui, très bien, avez-vous su un jour que vous étiez définitivement un geek, comment avez-vous attrapé le virus ?

R. T. : Je reviens un peu en arrière dans mon histoire personnelle. Flash back. Mon père dirigeait un hôpital psychiatrique en Californie et il s'intéressait à la recherche des raisons pour lesquelles il y a parfois des failles dans le développement d'un individu et il était assez connu. Linus Pauling qui était à Caltech à ce moment-là vint chez nous et ils menaient une recherche en commun et le Dr Pauling me laissa l'annuaire de Caltech et alors, j'avais cet intérêt scientifique déjà. Quand j'allai à l'école publique, c'était à cette époque où les écoles publiques étaient encore vraiment d'un niveau élevé, j'avais un professeur de math incroyable. C'est là que j'ai attrapé le virus pour étudier les mathématiques, je lisais les colonnes du Scientific American par Martin Gardner. J'ai eu aussi l'opportunité quand j'étais au lycée de faire des travaux de programmation. Ainsi, mes projets ont toujours été de faire des mathématiques. Je suis allé à Caltech en premier cycle universitaire mais malheureusement, je n'eus jamais cette chance de vous avoir comme professeur. Caltech à ce moment-là était plutôt orientée vers la physique, le département de math était un peu étrange mais merveilleux. De brillants théoriciens de la combinatoire, Herbert, Ryser, Marshall Hall, et donc je vécus une belle période. Je dois dire qu'être en premier cycle à Caltech a été probablement l'un des plus grands défis de ma vie. C'était extrêmement intense. Alors je me décidai pour la suite, je candidatai pour deux départements de math et deux départements d'informatique et je finis par

venir à Stanford en informatique, et je ne l'ai jamais regretté parce que cela m'a donné l'opportunité d'utiliser les mathématiques dans un sujet dans lequel on ne les trouvait pas habituellement, de façon concrète, sur les effets des algorithmes, et l'analyse des algorithmes et tout ça. Voilà la version courte de l'histoire.

D. K. : Bien et nous avons eu cette chance que John Hopkins bénéficie d'une année sabbatique, ce qui fait qu'il est venu en visite...

R. T. : C'était une excellente coïncidence oui, effectivement. C'étaient le début, je veux dire, Alan Kay était là. À cette époque, Stanford était un endroit très spécial. Alan Kay donna une conférence, déjà dans le cadre des HLF [2] et il dit quelque chose du style "Trouvez une grande communauté de recherche.". Stanford en ces temps-là, en informatique, était une grande communauté de recherche. Et bien sûr, il y avait l'investissement de personnes de Berkeley également : Dick [3] Karp et Waller, pour le premier cycle, c'était un endroit assez incroyable.

D. K. : Nous devrions rappeler à nos auditeurs que bien qu'on soit alors en 1970, c'étaient vraiment les années 60. *(R. T. rit)*. Je veux dire par là comme je le rappelle que vous viviez à Berkeley, en centre-ville et vous faisiez le chemin chaque jour, et comme je m'en souviens un peu, je m'inquiétais beaucoup parce que vous prouviez des théorèmes en conduisant sur l'autoroute et j'étais effrayé à l'idée que vous soyez retrouvé dans une épave. *(R. T. rit)*.

R. T. : Merci de votre sollicitude. Ça, c'est quand j'ai obtenu un poste de professeur assistant à Stanford. C'était après que j'aie obtenu mes diplômes.

D. K. : Oui, c'est ça, je me mélange un peu les pinceaux au niveau du temps depuis quelques années mais... Bon mais de toute façon, il se passait beaucoup de choses à cette époque, à l'extérieur des lycées aussi. Et je me rappelle que c'était une chouette époque. Je veux dire que les étudiants mettaient le feu aux immeubles dans les campus.

R. T. : Il y avait la guerre au Vietnam et il y avait des protestations contre la guerre et il y avait tout le mouvement hippie et c'était une période très intéressante. Beaucoup d'entre nous étions très idéalistes en quelque sorte. Peut-être un peu naïfs, ou un peu *trop* idéalistes.

D. K. : Oui, grâce à Dieu, je constate que les idées des gens sont aussi réalistes. Bien, maintenant, la chose dont je me rappelle le plus est que dans les années 70, vous avez révolutionné l'informatique, du moins de mon point de vue qui était d'essayer d'écrire un livre. Shannon et vous dans les années 70 étiez les deux personnes qui ont le plus contribué à la mise à jour de ma table des matières *(R. T. rit)* par rapport à ce que j'avais à faire, et la raison de cela était que, vous savez, vous faisiez des choses dont je n'avais absolument aucune idée en 1962 quand je mis au point ce plan pour les livres, et en particulier, c'était la première fois, ça semble évident aujourd'hui, peut-être, mais c'était si inhabituel de trouver quelque chose comme un algorithme profond, une propriété d'une toute nouvelle manière d'organiser l'information en informatique qui était... c'est loin maintenant, tout s'est mélangé, balayé par le fait qu'on pouvait vraiment prouver des théorèmes à propos de ces

2. HLF : Forum des lauréats d'Heidelberg.
3. Richard M. Karp

structures de données compliquées et avant ça, il y avait juste les structures de données que vous pouviez facilement expliquer à quelqu'un en dix minutes ou peut-être en une demi-heure mais tout à coup, vous êtes arrivés avec ces méthodes qui nécessitaient beaucoup de preuves et c'est même presque incroyable qu'elles marchaient. Je me rappelle de ma secrétaire Phyllis, elle disait que vous lui rendiez souvent visite et donc vous faisiez aussi de la recherche assis chez elle, j'imagine, encore en train de penser à ces trucs, mais ce que je veux dire, c'est pourriez-vous nous dire quelque chose à propos de l'origine de ces idées que vous avez conçues comme le fait d'avoir une structure profonde de données ?

R. T. : Eh bien, c'est une question à laquelle il est difficile de répondre : comment quelqu'un se rend-il compte de la façon dont il fait de la recherche. On essaie... on doit être intensivement curieux, on doit explorer un espace conceptuel, et le fait est que les ordinateurs deviennent de plus en plus rapides. La loi de Moore est un cadeau pour les théoriciens aussi bien que pour les expérimentateurs, elle offre la possibilité de penser aux choses à un haut niveau conceptuel et l'idée d'analyser les algorithmes d'un point de vue en quelque sorte un peu flou, en oubliant les facteurs constants, a vraiment ouvert le domaine et penser à la façon de vraiment démontrer la manière dont les temps peuvent être bornés qui n'avait vraiment pas été faite jusqu'à ce qu'on commence à le faire, je pense que c'était important parce que c'est en quelque sorte conducteur (de la recherche). Si vous avez comme but de faire un algorithme plus rapide alors vous êtes obligé de le déshabiller de tout ce qui n'est pas pertinent et donc d'identifier quelles sont les parties critiques qui sont impliquées lors de l'exécution et de les utiliser exactement comme il faut de telle façon que...

D. K. : Oui, ça c'était votre travail initial mais à cause de ce but que vous aviez d'obtenir vous savez, des temps linéaires ou juste cubiques ou quelque chose de ce genre, je pense, c'est surtout le défi du temps linéaire que vous avez réalisé qui vous a amené aux algorithmes au début, non ?

R. T. : Eh bien, comment ai-je vraiment commencé à travailler sur les algorithmes ? J'ai suivi un cours de programmation en Lisp donné par John McCarthy qui a lui aussi reçu la récompense A.M. Turing, malheureusement, et à nouveau, c'était dans ce grand environnement de Stanford, mais comme projet final, nous étions supposés écrire un programme Lisp. Une des possibilités consistait à écrire un programme Lisp pour tester la planarité des graphes. Je m'étais intéressé aux graphes planaires parce que j'étais intéressé par le problème des quatre couleurs, le problème qui est maintenant un théorème. J'essayais de le prouver moi-même alors que j'étais au lycée, en utilisant des méthodes de calcul mais je n'avais pas assez d'information et pas assez de puissance de calcul à utiliser en 1976 en utilisant exactement les techniques en question. Alors, le critère pour la planarité était le fameux critère de Kuratowski (un graphe est non planaire s'il contient un graphe complet sur cinq sommets ou un graphe complet biparti sur deux ensembles de trois sommets). Alors tous les autres essayaient d'implémenter le critère de Kuratowski mais j'ai trouvé un article de Shimon Even et Lempel et Cederbaum à ce moment-là qui donnait un algorithme qui construisait une représentation abstraite de l'intégration. Alors j'ai implémenté cet algorithme, et il s'est avéré en temps quadratique, ils ne l'avaient pas établi quant à eux mais le fait est là et j'ai réussi à vraiment bien l'utiliser pour notre projet. Alors j'ai pensé si on a pu le faire en temps quadratique, pourquoi pas plus vite ? Et John Hopcroft est venu et nous avons commencé à parler d'algorithmes de graphes et à explorer les possibilités de la recherche en profondeur d'abord et les choses amènent à d'autres choses. Quand vous avez un outil qui marche effectivement, vous l'essayez sur tous les problèmes

auxquels vous pensez. Mais revenons à toi, j'avance, revenons au cas ouest. Il me semble me rappeler d'une histoire célèbre où vous participiez à l'équipe de basket. Pouvez-vous nous en dire plus ?

D. K. : Eh bien, j'étais le capitaine et le teneur de marque et si quelqu'un veut voir ça, il y a une video d'une minute, il vous suffit de la google-youtuber, ça s'appelle *Le coach électronique*[4] et j'ai réfléchi pour que les statistiques au basket soient améliorées et j'ai mis au point une formule pour chaque joueur, qui tenait compte de la manière dont ils avaient contribué au match en cours, qui prenait en compte les essais de buts ratés, et les tentatives et les rebonds qu'ils avaient reçus, et des trucs comme ça, tout ça dans une grosse formule que j'ai oubliée mais en tout cas, je l'avais, ce n'était pas juste les points qui étaient comptés mais tout ce qui était fait et j'entrais ça sur cartes perforées à chaque fin de match et je calculais ces nombres et le coach semblait les apprécier.

R. T. : C'était en ces temps anciens où on programmait sur des machines IBM et où on devait entrer les trucs sur des cartes perforées dans des grandes boîtes énormes contenant des tas de cartes perforées...

D. K. : Et vous pourriez voir ça dans cette video et aussi, j'ai écrit un livre qui en parle, vraiment, je suis heureux d'avoir vécu assez lontemps pour pouvoir écrire un tel livre, il est intitulé *"Articles choisis sur le plaisir et les jeux"* et un des chapitres dans ce livre raconte toute l'histoire en incluant les articles de presse et la manière dont ça avait été mené, en montrant tout ça, dans le Newsweek magazine. *(D. K. rit.)*

R. T. : Donc on dirait que vous faisiez de l'analyse quantitative des sports trente ans avant que tous en fassent. Avez-vous eu la possibilité d'interagir avec des personnes qui en font aujourd'hui pour des équipes de sport professionnel ?

D. K. : Non, non, j'ai fait ça quand c'était facile et marrant mais j'étais un jeune-homme. Mais en tout cas, cela montre à nouveau que les ordinateurs ont toujours été partie prenante de ma vie, tout au long de toutes ces années...

R. T. : L'informatique (en anglais "Computer science") : on dit que tout domaine qui a le mot science dans son nom n'est en général pas une science. Alors je pourrais vous demander "l'informatique est-elle une science, une branche de l'ingénierie, une branche des mathématiques, un art ?", mais laissez-moi vous le demander de manière plus personnelle, spécialement parce que vous avez donné comme titre à l'un de vos célèbres livres *L'Art de la programmation des ordinateurs*. Vous vous voyez comme un artiste, un scientifique, un mathématicien, un ingénieur, un philosophe, une combinaison... ?

D. K. : Oui, ok, c'était le sujet de mon discours en 1974[5]. Et donc je ne peux faire mieux que de demander aux gens d'aller à cette source, mais pour résumer, j'ai regardé autour des termes d'art et de science, et j'ai trouvé, vous savez, beaucoup de livres qui avaient ces mots dans leur titre, j'en ai cité de nombreux ce jour-là et j'ai réalisé que le mot art est utilisé non seulement pour les Beaux-Arts mais également pour des choses qui sont artificielles et de façon basique, c'est un mot

4. https://www.youtube.com/watch?v=dhh8Ao4yweQ.
5. https://amturing.acm.org/award$_winners/knuth_1$013846.cfm.

qui vient du mot grec $\tau\varepsilon\chi\nu\varepsilon$, en allemand *Kunst* et etc. Il signifie quelque chose qui est fait par des êtres humains par opposition à quelque chose déjà présent dans la nature. Et la science est l'étude de la connaissance et de l'organisation des choses, et donc, en un certain sens, je peux vous fournir une courte définition qui est que la science, c'est ce que l'on comprend suffisamment pour l'expliquer à un ordinateur tandis que l'art, c'est tout le reste. Et comme la science avance, nous en apprenons de plus en plus à propos d'un sujet quelconque que nous étudions, mais alors, comme nous en apprenons de plus en plus à son propos, notre cerveau conçoit et effectue quelques sauts en avant et ça, c'est l'art.

R. T. : Ainsi cela amène une question très intéressante parce que l'intelligence artificielle est née à nouveau récemment en tant qu'apprentissage profond par réseaux de neurones et quelques personnes sont ennuyées de ce qu'une singularité risque d'advenir lorsque les ordinateurs deviendront plus intelligents que nous. Quel rôle les humains ont-ils là dedans et devraient-ils vraiment s'inquiéter, que pensez-vous de cette nouvelle version de l'intelligence artificielle ?

D. K. : Oui, bien, je ne sais pas combien de temps il faudra pour qu'ils renomment notre département en département de l'apprentissage machine mais je parlerai plutôt de toutes les autres choses possibles que celle-là, mais je suis sérieusement inquiet du potentiel en termes d'armes "artificiellement intelligentes" spécialement. Cette video, bon, comment appelle-t-on cela, déjà... tant pis, c'est une video si importante mais elle est sortie de ma tête, Steve Russell l'a postée pour montrer que le temps n'est pas si éloigné dans le futur où nous pourrions avoir des petits drones qui viseraient nos ennemis et personne ne pourrait les arrêter et ça ne serait pas trop difficile que des terroristes s'en emparent, sans compter les gens qui n'aiment pas d'autres personnes dans leur propre gouvernement ou bien dans d'autres pays et donc...

R. T. : Et Don, ça ressemble aux problèmes qu'on a déjà avec la technologie. Persiste toujours le fait qu'il y a des bons usages et des mauvais usages et que c'est à nous en tant qu'êtres humains d'utiliser les technologies d'une bonne façon, c'est un grand défi.

D. K. : Oui nous devons comprendre que c'est vrai, ça c'est sûr.

R. T. : Bon, pensez-vous que nous, en tant qu'informaticiens théoriciens ou en tant que mathématiciens, contribuons à ce nouveau domaine de l'intelligence artificielle ? Devrions-nous y contribuer ? Quel est notre rôle ici ? Je sais que certains de mes collègues ont franchi le pas d'essayer... et un grand défi semble être que les réseaux de neurones résolvent des problèmes, font des choses, accomplissent des tâches, mais que personne n'a d'indice pour savoir comment ils procèdent. Il n'y a pas d'explication, ce qui est un peu anti-scientifique, n'est-ce pas ?

D. K. : C'est vrai, excusez-moi d'avoir dit Steve Russell, je voulais dire Stuart Russell, bien sûr, et il a exprimé les pensées les plus profondes sur la manière d'anticiper tout ce qui pourrait se produire. Mais la chose malheureuse est que ses solutions supposent toujours que les êtres humains sont rationnels. Et je commence à m'interroger de plus en plus à ce sujet chaque jour parce que les être humains ne sont pas des êtres rationnels. Et nous devons du coup trouver un moyen de sauver le monde avec son irrationalité.

Mais de toute façon, comme vous dites, les personnes comme vous et moi avons des choses que nous savons bien faire et nous ne sommes pas nécessairement doués pour tout faire. Et donc j'imagine que lors des années de vie qui me restent, j'utiliserai mon temps à faire ce que je peux faire plutôt que ce que j'ai fait, et dont je ne suis pas sûre que quiconque pourrait le faire.

R. T. : C'est aussi la conclusion à laquelle j'ai abouti. Nous avons peu de temps et utilisons donc les capacités qui sont les nôtres du mieux que nous le pouvons.

Voici une question liée : l'informatique, contrairement aux mathématiques, est encore un domaine très jeune parmi de nombreux domaines. Vous êtes entré dans ce domaine à son tout début et j'y suis venu plus ou moins une dizaine d'années après. Qu'est-ce que je veux dire par là ? Je veux dire, comment voyez-vous l'informatique évoluer dans les prochaines années ? Comment a-t-elle évolué durant votre vie ? Y-a-t-il des directions dans lesquelles vous aimeriez la voir aller ?

D. K. : Mais peut-être qu'avant de répondre à ces questions, j'aimerais bien revenir à vous. Le fait que l'informatique soit un sous-ensemble des mathématiques ou que les mathématiques soient un sous-ensemble de l'informatique n'est pas très clair, et certains jours, je pourrais dire l'un ou l'autre, mais vraiment, je crois qu'elles sont différentes. Et bien que j'ai travaillé dans ces deux domaines, je ne me considère plus comme un mathématicien maintenant. Mais il y a des jours où je pense, oh, aujourd'hui, je vais penser comme un mathématicien. Et parfois je peux résoudre un problème en y pensant avec mon costume de mathématicien pendant une journée, et puis après, je mets ma casquette d'informaticien et je travaille à ce problème selon un autre point de vue. Et je sens vraiment une différence selon que je suis dans l'un des modes de réflexion ou l'autre.

J'ai discuté de ça avec Bill Thurston, pourtant, et il ne pouvait pas voir de différence. Du coup, je ne sais pas. Mais selon moi, je pense que l'informatique et les mathématiques sont les deux sciences pour lesquelles les corpus de connaissances établis jusque-là ne sont pas basés sur la nature.

R. T. : Ils sont créés par des êtres humains.

D. K. : Nous avons réussi à imposer nos propres règles. Vous savez, nous concevons l'univers que nous allons étudier et les physiciens n'ont pas cette possibilité, à part les théoriciens des cordes peut-être [6]. Mais vous savez, les chimistes, les biologistes, se confrontent à la nature. Mais la conception mathématique est un domaine où nous étudions des concepts détachés de la nature, que nous créons. C'est vraiment cette idée : les choses sont-elles tout le temps là juste dehors ou pas ? Mais en tout cas, cela sépare l'informatique des autres domaines.

Et je vois cela, comme les années passent, je pense que les mathématiciens sont en train de la saisir, cette manière dont je comprends l'informatique. Et les informaticiens sont souvent plus intéressés par Wall Street que par la science. Comme vous le savez, beaucoup de nos étudiants sont partis vers des fonds de pensions ou des choses du même style plutôt que de faire progresser le domaine. Et donc lorsque dans les années 70, on se demandait si les universités survivraient et etc., il était clair pour moi que si Stanford devait être brûlée, on réussirait toujours à rassembler un petit groupe d'entre eux et nous irions quelque part, et nous voudrions parler d'informatique.

6. léger petit tacle.

Et je n'ai jamais pensé quelque chose comme oh, quel bon moyen d'avoir une vie facile ou de gagner un maximum d'argent, ou quelque chose d'avoisinant comme démarrer une boîte, avoir toutes ces stock-options en un rien de temps. Non, c'était la dernière de mes motivations. D'un autre côté, je connais d'autres personnes qui fournissaient leur meilleur travail parce qu'elles étaient excitées par des choses différentes des choses qui m'excitent moi. Je ne dis pas que mon chemin est le seul chemin, je pense.

R. T. : Mais laissez-moi revenir à vous et vous donner l'opportunité de me poser quelques questions.

D. K. : Oui. Que pensez-vous de la motivation qui vous anime ? Comme par exemple, je pense le fait que vous êtes en train de terminer un livre ?

R. T. : De commencer un livre ! Bien, ce projet très long et longtemps reporté, je ne prends pas d'engagement. Mais oui, dans ce processus, j'ai découvert la beauté de TeX. Je dois dire que j'ai attendu trop longtemps pour apprendre TeX. Nous verrons comme ça évolue. *(D. K. rit).* Mais, vous savez, j'ai eu le même sentiment que vous décrivez lorsque j'ai développé ces algorithmes dans les années 70 et 80. Le domaine n'est pas encore suffisamment stable pour essayer d'inclure toute l'information dans le livre. Mais au point où nous en sommes, là, finalement, c'est une des choses positives qui sont advenues à cause de l'épidémie, la pandémie.

J'ai beaucoup plus de temps parce que je ne suis pas sans arrêt dans un avion. De ce fait, j'ai vraiment commencé à écrire quelque chose et cela nécessite de longues plages de temps, évidemment, et je pense qu'il y a maintenant suffisamment de stabilité pour que cela devienne possible d'écrire ce livre.

D. K. : Alors laissez-moi vous demander... Est-il vrai que vous aviez commencé à écrire une monographie et que vous avez perdu tout le manuscrit dans une station de métro ?

R. T. : Il était dans une mallette que j'ai laissée dans une station de métro à Paris, oui, contenant beaucoup de matériau que j'avais écrit au stylo sur papier. Maintenant c'est sur ordinateur et la plupart de ce que nous écrivons est stocké dans le cloud, heureusement. Mais cela m'oblige à faire confiance à une plateforme particulière, par opposition à vous qui ne faites confiance qu'en votre environnement local. J'espère que votre maison est sûre, Don, parce que nous ne voudrions pas perdre tout ce trésor d'information.

D. K. : Bien, je pense qu'elle est sûre. Oui, je... Quand vous parliez lors de la préparation de dialogue, vous avez mentionné que vous aviez des avis à émettre sur les publications et la culture des conférences, et des choses relatives à ça. Pouvez-vous développer...

R. T. : Oui, oui, oui. C'est une différence, je pense, entre les différents champs disciplinaires. Et je voudrais inclure les mathématiques ici, aussi bien que l'informatique telle qu'elle a évolué. Il semblerait que maintenant, nous mettions tous nos efforts dans un rapide flot de publications pour des conférences. Et si l'on se place dans le domaine des mathématiques, peut-être que ça fonctionne pour les maths expérimentales, mais pour les théoriciens, cela m'étonne. Je suis de la vieille école

en quelque sorte. Je veux dire que je suis un mathématicien caduque, comme vous d'ailleurs.

Cela me surprend que nous fassions des progrès malgré le fait qu'il y a plein d'erreurs, il y a plein de publications à moitié cuites avec de grands résultats, mais non approfondies. Et cela me fend le cœur que les gens ne prennent pas le temps de revenir en arrière et de remplir les détails, et d'aller plus avant. Du coup, cela me semble une tendance inévitable de notre domaine, et ça devient encore pire avec l'émergence de plateformes comme arxiv, où les gens peuvent poster à peu près n'importe quoi, au mieux à moitié relues par des pairs (referees).

Donc, je me sens concerné par cela. Je pense qu'il est très important d'essayer d'obtenir de tout article qu'il paraisse dans un journal à comité de lecture (refereed), si possible. J'essaie d'encourager mes étudiants à faire cela, mais c'est un défi. L'époque est de plus en plus au "publish or perish" qu'elle ne l'était lorsque j'étais étudiant.

D. K. : J'imagine que c'est parce qu'il y a des milliers de chercheurs alors qu'il n'y en avait que des douzaines quand vous et moi étions jeunes mais, Bob, bon, j'en connais assez sur cette histoire pour savoir qu'il y a toujours eu des problèmes concernant le contrôle de la qualité des publications. Mais je me rappelle l'année passée, j'ai vraiment été en colère quand j'en ai entendu parler, vous savez, j'avais regardé un papier et je pense qu'il pourrait résoudre un problème dont j'avais pensé pendant des années que je n'en verrais pas la solution durant toute ma vie et je pensais que ce serait une solution brillante mais le problème nécessitait une généralisation du problème de parcours du cavalier (sur un échiquier) et le gars a reçu les rapports de referee les plus insultants possibles quand il a soumis l'article à un journal et je les ai montrés, vous savez... et surtout dans ces comités académiques de haut niveau de pairs pour le dire vite, personne ne se serait abaissé à écrire quelque chose comme la généralisation d'un bel outil, et donc, on n'a pas à prendre en compte les goûts, et je ne peux pas dire que mes goûts soient meilleurs que d'autres et je peux voir un article et penser ça c'est poubelle et puis en voir un autre et le trouver beau. Je n'ai pas d'algorithme, mais je n'utilise pas le machine learning [7], pour résoudre leurs problèmes.

R. T. : C'est le revers de la médaille : je veux dire, de grands articles ont été rejetés par des reviewers au front bas et la grandeur n'émergera que plus tard, peut-être.

D. K. : Bien. Plus le... J'ai finalement trouvé un bon referee pour le type, mais vous savez, il travaillait pour un journal qui... vous savez... vissait les universités qui le chargaient trop. Et alors il ne voulait pas publier dans ce journal. Il l'a donc laissé sur arxiv pour le moment. Vous savez, dans un journal comme le journal des algorithmes dont j'ai fait partie du comité de rédaction, j'ai cherché ce qui est fait par les éditeurs patients qui dans la plupart des cas utilisent gratuitement nos services pour faire tout le travail éditorial et de refereeing, mais qui ne s'intéressent à rien de plus qu'au titre et aux profits énormes qu'elles réalisent, dont le bureau académique a été acheté par quelqu'un qui a été acheté par quelqu'un qui lui-même etc. et finalement, les éditeurs font des arrangements avec les universités et astreignent les universités au secret de façon à ce que nous ne pourrions même pas retrouver quelle est la bibliothèque qui a finalement effectivement payé pour le journal et c'était mon dernier tirage et j'ai rapporté toutes ces choses à notre comité éditorial de notre journal sur les algorithmes et nous avons fait remonter notre participation à 100 % et nous

7. léger petit tacle.

avons démarré les transactions de l'ACM sur les algorithmes...

R. T. : Est-ce la solution alors : davantage de concours financier de la part de société professionnelles et pas de sociétés lucratives (privées) ?

D. K. : Oui, il me semble que nous avons un certain contrôle là-dessus à travers les sociétés professionnelles. Je ne sais pas quelle est votre opinion... ?

R. T. : Je suis d'accord, je suis d'accord...

D. K. : Tim Gowers a...

R. T. : Je vais changer de sujet, désolé, poursuivez... Je vais un peu changer de sujet. Vous êtes non seulement un informaticien mais vous êtes également un organiste de talent ainsi qu'un compositeur. Pouvez-vous nous en dire un peu plus sur les projets que vous avez déjà menés et ceux que vous menez en ce moment dans ce domaine ?

D. K. : D'accord, donc de toute façon, vous étiez à la fête d'anniversaire de mes 80 ans, l'un des grands amis qui vinrent là.

R. T. : Et j'avais un tee-shirt qui aurait pu être celui-là.

D. K. : *(D. K. rit.)* Ok, j'ai maintenant 82 ans mais lors de cette célébration, nous sommes allés dans le nord de la Suède où se trouve un merveilleux orgue à tuyaux et quand j'avais entendu parler de cet orgue, je m'étais dit... J'avais souhaité depuis mes 60 ans écrire une sorte de composition majeure pour orgue à tuyaux. C'était un rêve que j'avais toutes ces années et que j'ai finalement réussi à réaliser, je ne sais pas, quand j'avais soixante-quinze ans, je m'étais dit, si tu veux réaliser ce rêve, il faudrait voir à peut-être commencer maintenant. Alors un jour où je visitais Vienne, je suis entré dans le meilleur magasin de musique de Vienne et j'ai acheté du papier à musique vierge dans le même magasin que celui dans lequel Beethoven achetait ses stocks de papier musique, et Brahms, et etc, et je me disais ça ne peut pas faire de mal et j'ai emporté ce papier avec moi et j'ai commencé à écrire ma pièce. Et j'ai travaillé dur à cette pièce pendant cinq ans et j'ai abouti à une œuvre très étrange dont je dois dire que je l'aime beaucoup... Je suis heureux qu'elle soit née comme ça. Et donc nous avons eu cette Première glorieuse. Vous pouvez la voir sur Google-Youtube postée par un groupe de gens qui ont fait la captation video, ils avaient du matériel dernier cri, quelques douzaines de caméras. Du coup, nous avons un enregistrement à 360° par deux caméras et de nombreux enregistrements audio, la meilleure équipe de mixage et tout ça. Nous avons donc plusieurs teraoctets de données de cette performance pour la Première. Et j'espère que des informaticiens pourront mener un projet de thèse sur la manière dont on pourrait mettre les choses en beauté grâce à la réalité virtuelle, ce qui permettrait aux auditeurs de sélectionner ce qu'ils veulent regarder pendant l'écoute parce que nous avons aussi tous ces bits de données sur arxiv à Stanford sous forme digitale et c'est accessible à tout chercheur qui le souhaite.

R. T. : Je dois dire que le voir en personne a été une expérience grandiose et ce serait bien d'avoir la possibilité de revivre cela.

Oui donc la prestation est divisée en vingt-deux parties chacune d'entre elles durant environ cinq minutes et vous pouvez les voir une par une si vous allez sur mon site web, vous pouvez trouver la playlist sur youtube [8]. Mais alors a aussi eu lieu une première canadienne et il y eut une très belle représentation à Waterloo, il y a deux ans maintenant, et des personnes filmèrent là une belle video qui vous permet de voir l'œuvre d'une autre manière. Et l'année passée à Brno [9] en République tchèque, un merveilleux enregistrement de la performance a été fait à nouveau. Donc maintenant il y a trois excellentes sources que quiconque le souhaite peut visionner. J'appelle ça de la musique à large bande parce que je réagissais à l'excès de musique moderne... Je ne sais pas. Ils commencent avec une idée de thème et ils tiennent ce thème jusqu'à ce que l'audience l'ait aussi et alors, ils passent une autre idée et ainsi de suite. Mais je voulais revenir à la manière dont Mozart réussit cela, qui fait que les idées viennent plus vite que nous ne réussissons à les saisir et nous sommes forcés d'écouter deux ou trois fois et nous saisissons davantage d'éléments à chaque écoute... Ainsi ce qu'ils ont fait est vraiment beau et je suis si heureux que cela devienne une sorte de musique du monde et peut-être que mon œuvre deviendra populaire et peut-être que ça sera un bide mais au moins, cela me satisfait d'avoir eu ce but tout au long de ma vie, c'est difficile à expliquer mais si vous m'en aviez parlé il y a deux ans avant que j'aie fini et si quelqu'un m'avait dit qu'il ne me restait plus que deux ans à vivre et que j'aie eu à choisir entre terminer cette œuvre musicale ou bien finir d'écrire un programme comme... J'aurais sûrement choisi la musique même si j'aurais plutôt dû choisir l'informatique parce que je n'ai pas vraiment le droit d'écrire de la musique.

R. T. : Je ne vois pas pourquoi vous n'en auriez pas le droit. Laissez-moi vous interroger dans l'intérêt des jeunes chercheurs. Avez-vous un conseil pour les personnes qui veulent venir à l'informatique telle qu'elle est aujourd'hui. Avez-vous des conseils pour les étudiants ou pour les mentors des étudiants ?

D. K. : Hier, quand la question a été posée à Leslie Lamport et à Tony Hoare, la réponse de Lamport a été si merveilleuse, il a dit "Écrire.", que faire ? "Écrire.". Et je crois que j'ai monopolisé la parole plus que vous ici mais j'en profite pour vous dire "Finissez ce livre sur lequel vous êtes en train de travailler." *(R. T. rit)* parce que toute ma vie, j'ai trouvé que je réalisais une sorte de combinaison convexe entre les mathématiques et l'écriture. Mais écrire était toujours très important et garder cette activité est important, c'est pour cette raison que j'écrit tant de programmes informatiques aujourd'hui encore.

R. T. : Je dois certainement être d'accord avec ça, bien, bien, bien, réécrire. Mais c'est un défi difficile. Les idées sont sans valeur si vous ne pouvez pas les communiquer à d'autres personnes et le meilleur moyen de le faire et de les coucher sur du bon vieux papier, je dirais.

D. K. : Et l'autre chose qui m'est venue à l'esprit à travers ce dialogue écouté hier que je dirais d'une façon différente est ne soyez pas trop influencé par les trucs à la mode. N'écrivez pas un papier juste parce que vous devez écrire un papier ou parce que vous pensez que vous devez impressionner les gens à propos de quelque chose qui ne vous intéresse pas directement mais parce que vous cherchez à donner le change. C'est la pire raison pour écrire un papier ; le modèle auquel

8. https://www.youtube.com/playlist?list=PLvixIGKr5sJffdfwecygYqhXsgz-EBCC8.
9. https://www.youtube.com/watch?v=wk7dEKMPP68

j'aime penser à ce propos est celui d'Euler dont les articles étaient toujours très impressionnants mais parce que ses actes provenaient du fait qu'il voulait dire aux gens ce qui l'impressionnait.

R. T. : Oui, je pense à Alan Kay qui a aussi présenté un exposé à ce forum, il avait aussi cette notion d'idées hors-la-loi qui sont écrasées... Vous devez suivre votre propre chemin en quelque sorte, vous devez trouver quel est votre propre chemin et le suivre. Les meilleurs étudiants que j'ai eus arrivaient ou repartaient avec leurs propres idées qu'ils avaient développées et vraiment, Pat Hanrahan plus tôt cette semaine l'a aussi dit, je ne donne jamais à mes étudiants de sujet de thèse, j'exige d'eux qu'ils trouvent leur sujet eux-mêmes, maintenant, c'est une sorte d'approche rigoureuse mais ça fait partie de ce dont il faut être capable pour faire une thèse, il s'agit d'apprendre à se poser les bonnes questions, c'est davantage le fait de se poser les bonnes questions plutôt que de trouver des réponses à des questions que d'autres ont posées.

D. K. : C'est ça, exactement ça, et en fait donc, il s'agit de poser les questions, c'est juste ce dont vous devez montrer que vous savez le faire comme il faut *(R. T. rit.)*.

R. T. : Laissez-moi vous poser une question de plus, nous arrivons au bout de notre temps d'échange mais j'ai noté que selon wikipedia, et c'est une question de l'un des jeunes chercheurs, vous êtes renommé pour vos blagues scientifiques, je me demande si vous pouvez nous dire quelle est votre blague favorite ?

D. K. : Ok, alors *(D. K. montre l'un de ses livres ouvert à la bonne page.)*. Jetez un œil à la bibliographie de *Mathématiques concrètes*, la référence 44, *T. Brown, Multivariable subpolynomial waffles which do not satisfy the lower regular q-property* pointe vers une collection de 250 articles sur la théorie des gaufres dédiée à R.S. Green pour son anniversaire, ses 23 ans... Et alors, j'ai une note en marge. Dans la marge, ça dit "De tels articles ne sont pas cités dans ce livre". *(R. T. rit.)* Et c'est pour cette raison que j'ai écrit *Mathématiques concrètes* et vous regardez l'index et vous voyez qui était T. Brown et il est nommé Trivial Brown, c'est rigolo de regarder les différentes traductions de *Mathématiques concrètes*.

R. T. : Il me semble me rappeler que votre première publication était dans un magazine inhabituel. Pouvez-vous nous rappeler quel magazine c'était ?

D. K. : Juste une seconde. *(D. K. se lève et va chercher le magazine en question.)* Ok, un de ces jours, essayez de mettre la main sur ce livre *Recueil d'articles sur le plaisir et les jeux*. Il parle non seulement de basket-ball mais il parle également de mes premières publications dans le magazine Mad, me voilà *(D. K. approche le magazine de la webcam pour qu'on le voit bien.)*.

R. T. : Donc voici votre conclusion, Knuth, les jeunes chercheurs devraient retrouver votre premier article publié dans le magazine Mad. Tant de questions, si peu de temps, laissez-moi vous remercier, Don, pour un merveilleux dialogue et je suis ravi d'y avoir pris part et merci au HLF et à tous les organisateurs.

D. K. : Ok, j'en déduis que nous devons arrêter là, bye-bye.

Conclusion : *Merci du fond du cœur, merci beaucoup pour votre perspicacité et je suis sûr que votre discussion sera très inspirante pour les jeunes chercheuses et les jeunes chercheurs. Nous avons vraiment apprécié que tous deux, vous preniez tout votre temps. Prenez soin de vous et merci beaucoup.*

D. K. et R. T. : Merci Peter.

L'informaticien qui ne peut pas s'arrêter de raconter des histoires

Donald E. Knuth

Pour le pionnier de l'informatique Donald Knuth, une bonne programmation est synonyme d'une belle expression.

Susan D'Agostino, 16 avril 2020

Donald Knuth est un informaticien majeur de sa discipline. Pendant les années du démarrage de l'informatique, dans le milieu du siècle dernier, une compagnie fabriquant du sucre lança un concours qui consacra les talents de Donald Knuth alors qu'il avait 13 ans. Le concours demandait aux enfants de déterminer le nombre de mots qu'on pourrait faire avec les lettres de la société : Ziegler's Giant Bar. C'était un problème bien défini avec différentes parties, juste le genre de problème qu'il adorait.

"J'ai eu une période obsessionnelle-compulsive qui m'a dirigé vers les problèmes digitaux, discrets. Et j'adore me pencher sur de grandes quantités d'information", dit Knuth.

Knuth feuilleta méthodiquement le dictionnaire intégral Funk & Wagnalls de 2000 pages de sa famille. Il réussit même à convaincre ses parents qu'il était malade, gagnant deux semaines loin de l'école pour travailler sur ce problème. Après avoir étiqueté les index avec des entêtes comme "Aa," "Ab" et "Ba" basés sur les commencements possibles des mots utilisant les lettres de la société sucrière, il éplucha les colonnes du dictionnaire en notant les mots qui devaient être conservés. Il découvrit qu'il pouvait sauter des sections entières du dictionnaire, comme les pages commençant par la lettre "C," ou bien les sections des mots commençant par "B" et dont la deuxième lettre était "U."

Les instances officielles du concours avaient identifié environ 2 000 mots qui pouvaient leur être fournis, mais Knuth en trouva plus de 4 700. Son gain au concours fut un spot publicitaire à la télévision et du chocolat pour toute sa classe. Il continuerait à gagner de nombreuses autres récompenses, incluant la première récompense Grace Murray Hopper de l'ACM Award, la médaille nationale de la Science et la récompense A.M. Turing.

Knuth finit par agréger ses deux amours des problèmes discrets digitaux et des larges collections d'informations dans son œuvre majeure, *L'Art de la programmation des ordinateurs* - une série de livres qu'il commença à écrire pendant ses études en 1962 et qui n'est toujours pas achevée. Il publia le volume 1 en 1968, et la version actuelle en est sa 42$^{\text{ème}}$ réimpression. Le volume 2 suivit en 1969 et le volume 3 en 1973. Donald Knuth était alors professeur d'informatique à l'Université Stanford, mais le fait que son travail puisse l'empêcher de terminer ses livres l'inquiétait. Alors il prit un congé sabbatique en 1990 et il prit sa retraite en 1993

traduction (Denise Paule Vella-Chemla, 11.10.2020) d'un article de Quanta magazine, lisible ici https://www.quantamagazine.org/computer-scientist-donald-knuth-cant-stop-telling-stories-20200416/.

pour passer le reste de sa vie à terminer l'ensemble des sept volumes. Maintenant âgé de 82 ans, il travaille dur sur la partie B du volume 4, et il anticipe que ce volume aura au moins six parties de A à F.

L'Art de la programmation des ordinateurs est plus qu'un manuel d'utilisation. Comme Isaac Asimov et Eric Temple Bell ont tissé dans leurs romans et autour de leurs personnages des histoires de science et de math, Knuth se régale à raconter des histoires d'informatique.

"Le meilleur moyen de communiquer d'un être humain à un autre consiste à raconter une histoire," dit-il.

Cette passion pour la communication l'a aidé à jouer un rôle mettant en vedette son œuvre majeure dans l'histoire de l'informatique. Lorsque son éditeur lui envoya les épreuves de la seconde édition du volume 2 dans les années 70, Knuth fut perturbé par l'agencement et l'apparence des nombres, des symboles et des mots sur les pages. Il prit un vol pour Los Angeles pour voir une machine qui imprimait des magazines sur papier brillant, en espérant que cela lui offrirait un gain esthétique, mais c'était trop cher. Toutefois, pendant ce voyage, il commença à développer un langage informatique qui lui permettrait de composer des mathématiques de manière digitale.

De retour à Stanford, Knuth mit en sommeil *L'Art de la programmation des ordinateurs* pendant presque une dizaine d'années pour développer TeX (prononcé "tech"), un langage sophistiqué et qui changea la donne, en pourvoyant les ordinateurs de la typographie digitale. Il le rendit open source, surtout pour le bénéfice des mathématiciens professionnels, des informaticiens, des économistes, des ingénieurs, des linguistes, des statisticiens et de toute autre personne pour qui des symboles techniques manquaient à son clavier mais qui maîtrisait le placement de formules complexes mieux que ses éditeurs. Dans le monde des programmes éphémères d'ordinateurs, TeX a perduré comme le standard en or pour écrire des articles scientifiques plus beaux et plus faciles à lire et à comprendre par des experts.

L'intérêt de Knuth pour la narration d'histoire l'a également amené à développer une philosophie de la programmation littéraire - une méthode pour écrire des programmes d'ordinateurs comme des essais littéraires. Un programme littéraire intercale au sein du source une élégante prose en une langue naturelle, comme l'anglais. Le code source a pour caractéristique d'être fonctionnel et efficace, alors que l'exposé en langage naturel cible le lecteur humain plutôt que le compilateur de l'ordinateur. Quiconque devra un jour mettre à jour ou debugger un programme littéraire évitera le problème chronophage et coûteux consistant à essayer de comprendre les algorithmes qu'a mis en place le programmeur initial, les décisions conceptuelles et les stratégies d'implémentation qu'il a choisis. Knuth est un informaticien qui comprend que les mots comptent.

Quanta Magazine s'est entretenu avec Knuth en Févrirerchez lui sur le campus de Stanford. L'interview a été condensée et éditée pour être plus claire.

Avez-vous toujours été intéressé par l'écriture ?

J'ai compris tôt que le monde réel était trop difficile pour moi. Je ne m'attendais pas à découvrir quoi que ce soit de nouveau, mais j'aimais transmettre mon enthousiasme pour les idées en écrivant.

En classe de sixième, deux amis et moi commençâmes un journal de deux pages sur une machine ditto. Nous avions des blagues. Au collège, chaque nuit de lundi en tant qu'éditeur de journal, je faisais une nuit blanche pour que le journal puisse sortir. J'ai vu ma première ligne tapée au lycée, en tant qu'éditeur du journal des étudiants. Quand j'étais junior et senior, nous avons commencé l'ingéniorat et les revues scientifiques. Par exemple, j'ai éérit, "Th_5E_4 $C_3EmIC_2Al_2$ Ca_3P_4Er." Chaque mot était une formule chimique.

Et cela vous a amené à votre œuvre majeure ? Y pensez-vous comme à une autre histoire ?

L'Art de la programmation des ordinateurs est un manifeste. Il décrit la façon dont j'aime les maths et la manière dont j'aurais aimé qu'on me les apprenne. Dès la page 1, je racontais l'histoire des algorithmes. La plupart des livres à ce moment-là n'explorait pas le côté humain des découvertes. Ils disaient juste, "Voici comment marche la chimie," ou "Voici comment marche la physique."

Je raconte aussi une histoire technique. Je dis, "Voici quelque chose qui ne marche pas, et voici la manière de résoudre ce problème." Plutôt que de présenter seulement des faits, j'ajoute du drame. La science est beaucoup plus facile à apprendre si vous connaissez l'ordre dans lequel ont eu lieu les découvertes. En outre, je suis incapable de résister à une bonne histoire. Je ne me voyais pas comme un pionnier, mais comme un journaliste.

Au-delà de l'histoire, alors, de quoi parle *L'Art de la programmation des ordinateurs* ?

Après deux années passées à écrire mon livre, j'ai réalisé qu'un programme était quantitativement déterminé comme bon quand il était nouveau. Je ne voulais pas juste dire qu'un programme était meilleur qu'un autre. Je voulais dire celui-ci est de 13.8 % meilleur que celui-là, et expliquer comment les comparer.

Admettons que l'auteur A parle de son algorithme A, et que l'auteur B parle de son algorithme concurrent B. Et l'auteur A n'a jamais écrit à propos de l'algorithme B, et l'auteur B n'a jamais écrit à propos de l'algorithme A. En plus, les auteurs A et B ont utilisé des ordinateurs différents. En tant que journaliste neutre, j'expliquais les deux d'un seul point de vue. Demander "Dans quelle proportion cet algorithme est-il bon ?" est un problème marrant. C'est cela l'analyse des algorithmes.

Est-ce que "l'analyse des algorithmes" est juste une manière différente de dire "l'art de la programmation des ordinateurs" ?

J'assistais à une conférence de la Société de mathématiques industrielles et appliquées en 1967 quand quelqu'un me demanda ce que je faisais. À ce moment-là, l'informatique était découpée en analyse numérique, intelligence artificielle et langages de programmation. C'était ça. J'ai réalisé que j'avais besoin d'un nom pour ce que je faisais.

La nouveauté dans mon livre, c'étaient les études rigoureuses de l'efficacité des algorithmes. Je décidai que la prochaine fois qu'on me poserait la question, je répondrais "l'analyse des algorithmes". Ma définition était : si ça m'intéresse, c'est de l'analyse des algorithmes. Ça n'était pas une très bonne définition.

Plus tard, je décidai de le justifier. Je décidai que c'était l'étude quantitative, i.e. la mesure de la valeur d'un algorithme, que je divisais en deux parties. Une partie considérait tous les algorithmes possibles pour résoudre un problème donné. L'autre partie considérait un algorithme particulier pour un certain problème.

L'Analyse des algorithmes allait devenir le travail de ma vie. Je dis à mon éditeur de changer le titre de mon livre en *L'analyse des algorithmes*. Mon éditeur dit, "ça ne se vendra jamais". Ils ont pris la bonne décision. Et j'ai été content quand, 40 ans plus tard, cinq ou six livres sont sortis avec le titre *Analyse des algorithmes*.

Mais pour vous, la programmation est davantage que simplement fonctionnelle. Quand vous avez conçu TeX, par exemple, vous vouliez trouver "la courbe la plus plaisante" reliant certains points. Étiez-vous en train d'essayer de programmer la beauté ?

Mon programme devait relier des points d'une manière qui correspondrait à l'ingénierie-inverse de ce que ferait un expert calligraphe. Il y a un point dans la lettre "S" où la courbure change et de positive devient négative. Alors peut-être peut-elle rester constante un peu de temps. Les concepteurs de lettres suivent une certaine logique pour transformer des lignes en formes de lettres. Je voulais capturer non seulement le résultat de cette conception, mais également l'intelligence qui se cache derrière cette conception. C'est comme écrire un programme d'ordinateur.

J'ai parlé à des designers pour comprendre comment ils parvenaient à procéder de la sorte. Les maths étaient là pour capturer le design de manière quantitative. Avec les mathématiques, j'ai mis des cadrans autour de chaque partie. Je pouvais dire la lettre "A" a ce point, cette épaisseur, ces angles ici, elle diminue là, une bosse en haut et une certaine longueur d'empattement.

Je n'ai jamais essayé de remplacer les concepteurs. Je voulais seulement capturer exactement

pour les générations futures ce que nous étions alors en train de faire. Avec TeX, le design est reproductible.

Aviez-vous anticipé l'acceptation globale de TeX ou sa capacité à perdurer longtemps ?

TeX était seulement conçu pour être utilisé par ma secrétaire et moi-même. Phyllis [Astrid Benson Winkler] était une merveilleuse secrétaire. Elle pouvait lire mes manuscrits et les rendre beaux. La technologie de l'impression était en train de passer de mode parce que les méthodes par essais-erreurs devenaient trop chères. Presque tous les articles publiés dans les années 70 étaient atroces à regarder. Dans les *American Mathematical Monthly*, les indices étaient écrits dans une fonte différente des lignes principales de texte. Je savais que l'informatique rendrait à nouveau les livres beaux.

Je finis de débugger une version temporaire de TeX en avril 1978. En mai, j'avais 10 utilisateurs. En juin, j'avais 100 utilisateurs. En juillet, j'en avais 1000. Chaque nouveau groupe dirait "Est-ce que vous avez cette fonctionnalité dont j'ai besoin ?". Cinq ans après, je produisais une mise à jour qui est essentiellement le TeX que nous avons aujourd'hui. Il était destiné à des américains. Alors les européens ont commencé à l'utiliser. De ce fait, dans les années 80, je l'ai fait fonctionner pour les langages du monde entier.

À vous entendre, on a le sentiment que la découverte a toujours fait partie de votre cheminement. Cela reste-t-il vrai aujourd'hui ?

J'écris en moyenne cinq nouveaux programmes par semaine. Les poètes doivent écrire des poèmes. Je dois écrire des programmes d'ordinateur.

Le test ultime pour savoir si je comprends quelque chose est de voir si je peux l'expliquer à un ordinateur. Je peux vous dire quelque chose et vous opinerez de la tête, mais je ne suis pas sûr que je vous l'ai bien expliqué. Mais l'ordinateur n'acquiesce pas de la tête. Il me répète exactement ce que je lui ai appris. Dans la plupart des circonstances dans la vie, vous pouvez bluffer, mais pas avec les ordinateurs.

Vous passez vos journées à écrire, mais vous avez aussi d'autres centres d'intérêt. Comment abordez-vous chaque journée ?

Jack London écrivait 1000 mots chaque jour avant de s'adresser à tout le monde. Il était assez totalitaire par rapport à ça, "Laissez-moi seul jusqu'à ce que j'aie écrit mon millier de mots !" Alors il pouvait boire ou relire le reste de la journée. Non, mon principe en terme d'emploi du temps est de faire la chose que je déteste le plus qui est sur ma liste de choses à faire. Quand approche le week-end, je suis très heureux.

Vraiment ? Comment faire des choses que vous détestez peut-il vous rendre heureux ?

Ce serait très facile pour moi de dire "Oh, laissez-moi être un génie et ne jamais nettoyer les toilettes." Mais même laver les toilettes est une tâche faisable. [Mon épouse] Jill et moi avons des uniformes dotés d'une fente dans laquelle on peut glisser un flacon de nettoyant 409. Vous y allez, faites gicler le produit et vous vous sentez bien d'avoir nettoyé les toilettes !

Le succès d'une personne dans la vie est déterminé par le fait qu'elle ait un potentiel minimum élevé, pas un potentiel maximum élevé. Si vous pouvez vraiment bien faire quelque chose mais qu'il y a d'autres choses que vous ne savez pas bien faire, ces dernières choses vous retarderont. Mais si pour presque tout ce que vous faites, vous êtes au top, alors vous aurez une bonne vie. Et c'est pour ça que j'essaie d'apprendre comment passer à travers les choses que les autres trouvent déplaisantes.

Vous avez aussi de nombreux projets qui n'ont rien à voir avec l'informatique, comme la composition musicale, *Fantasia Apocalyptica*. Vous avez même construit votre maison autour d'un orgue à tuyaux à deux étages. Est-ce que cette variété fait aussi partie de ce qui vous rend heureux ?

J'ai écrit deux livres, incluant *Les choses dont un informaticien parle rarement*, qui parlent de théologie - des choses que vous ne pouvez pas prouver - plutôt que de mathématiques ou d'informatique. Ma vie ne serait pas complète si elle était entièrement consacrée à des choses déconnectées et arides. Les choses mystiques que je ne comprends pas me donne de l'humilité. Il y a des choses au-delà de ma compréhension.

En mathématiques, je sais quand un théorème est correct. J'aime ça. Mais la vie ne vaudrait pas le coup si tout était faisable. Cette connaissance ne m'affaiblit pas. Plutôt, elle m'assure de ne pas rester coincé dans une ornière.

Est-ce que cela importe que vous terminiez *L'Art de la programmation des ordinateurs* ?

Oh, je réalise que l'informatique continuera de vivre et de se développer. L'un des scenari est que tout le monde cessera de travailler sur les ordinateurs que nous avons actuellement. On ira tous vers l'apprentissage machine et les ordinateurs quantiques. Alors je pourrai assister à la fin de l'histoire des ordinateurs non quantiques. Je suis toujours plus content quand je peux dire "Voici la fin de l'histoire". C'est la façon la plus facile d'imaginer que moi aussi, je finirai. Mais je ne réponds pas à votre question.

Légendes des photos

Donald Knuth a travaillé sa vie entière pour raconter des histoires avec et à propos des programmes d'ordinateur. (Vivian Cromwell)

Knuth travaille en ce moment sur la partie B du volume 4 de L'Art de la programmation des ordinateurs. Il envisage que ce volume ait quatre parties de plus, et que l'ensemble de la série ait trois volumes de plus. (Jill Knuth)

Knuth a toujours raconté des histoires, en leur donnant une tournure scientifique. Au lycée, il écrivit une courte histoire dans laquelle chaque mot était une formule chimique. (Jill Knuth)

Knuth dans son bureau à Stanford avec le concepteur de fontes Herman Zapf en 1980, juste quelques années après la version du programme de typographie de Knuth, TeX. (Chuck Painter / Stanford News Service)

"Le meilleur moyen de communiquer d'un être humain à un autre consiste à raconter une histoire," dit Knuth. Cette approche lui a permis d'analyser les algorithmes plus rigoureusement et de façon neutre. (Vivian Cromwell)

Knuth chez lui en 2020. Il a pris sa retraite en 1993 pour terminer son œuvre majeure, L'Art de la programmation des ordinateurs. (Jill Knuth)

Textes des "accroches" ou "coiffes" [1]

Je suis incapable de résister à une bonne histoire. Je ne me voyais pas comme un pionnier, mais comme un journaliste.

Je savais que la programmation des ordinateurs rendraient à nouveau les livres beaux à regarder.

1. terme journalistique exact ?

Les arts culinaires et l'informatique

La pensée dépend absolument de l'estomac.
— DIDEROT, lettre à d'Alembert (1770)

Bien sûr nous savons que nos navigateurs utilisent des "cookies" et des "menus". Et il y a une organisation appelée CodeChef, qui traite chaque mois les concours de Codage. Mais la relation entre l'informatique et la nourriture s'étend maintenant au cœur profond de notre discipline, depuis que nous sommes fondamentalement concernés par l'étude systématique des processus. Bien longtemps avant que l'informatique n'ait été inventée, des chefs anonymes ont découvert un grand nombre des principes de base qui nous guident aujourd'hui : les séquences pas à pas d'instructions ; l'exécution conditionnelle ; les répétitions aléatoires ; le parallélisme ; l'évaluation paresseuse ; la liaison tardive ; le codage participatif ; etc.

Merci à Dieu, pour cette faculté que nous avons de griller, rôtir, pocher, badigeonner, braiser, malaxer, faire cuire, pétrir, et pour la faculté que nous avons de communiquer ces techniques de génération en génération !

Des siècles d'expérimentation ont montré quelles recettes si goûteuses on pouvait tirer d'ingrédients de la nature qui ne sont habituellement pas délicieux ou nutritifs. Mais il est également clair que de nombreux plats restent à découvrir, parce que seule une minuscule fraction des possibilités a déjà été testée. Le nombre d'ingrédients est limité, mais le nombre de façons de les combiner et les préparer est exponentiel. Même si nous nous limitons à un temps polynomial, nous serons capables de n'explorer qu'une minuscule part de ce vaste territoire gustatif.

Par exemple, saviez-vous que les myrtilles fraîches et les avocats, enveloppés de pâte sont assez délicieux ? En combinant simplement trois ingrédients, choisis au hasard dans différents rayons du magasin d'alimentation près de chez vous, il y a de grandes chances que vous soyez le premier à découvrir quelque chose qui mérite d'être partagé.

Quand je cuisine pour moi seul, j'aime bien utiliser la "méthode du diagramme de Venn", selon laquelle je peux tester tous les sous-ensembles d'un ensemble donné de saveurs. Par exemple, je peux mettre des œufs frais, de l'ail, de la menthe, et des petits-pois dans un poêlon, de telle manière que chacune des 15 combinaisons non vides occupe environ 1/15ème de la place totale dans l'ustensile. Alors, si j'étais capable de me retenir de remuer le contenu, je pourrais manger mon déjeuner à même la poêle, savourant chacune des possibilités isolément.

Les biochimistes ont introduit de nouvelles expériences gustatives révolutionnaires connues sous le nom de *gastronomie moléculaire*. Les informaticiens ont-ils quelque chose de similaire à offrir au monde, quelque chose qui mériterait le nom de *cuisine à l'informatique* ? Je pense que nous pouvons le faire : nous pouvons potentiellement enrichir la gastronomie en y incorporant ce que nous avons appris à propos des *paramètres*. En effet, j'ai souhaité une application que mon épouse et moi utiliserions quotidiennement, qui pourrait peut-être s'appeler METAFOOD. Cette appli saurait exactement quels ingrédients sont disponibles dans notre garde-manger et notre réfrigérateur, ainsi que le genre de matériels de cuisine dont nous disposons. Nous demanderions à METAFOOD de nous suggérer un menu (potentiellement aléatoirement) basé sur ce qu'on pourrait faire et le temps nécessaire pour le faire. Quand nous nous serions mis d'accord sur le but, METAFOOD nous dirait quoi faire en premier, quoi faire ensuite, etc. Il regarderait nos progrès - en changeant en cours de route, si nécessaire, comme le fait le système de navigation dans une automobile. Naturellement, nous communiquerions avec METAFOOD par micro et caméra, pas en touchant l'écran avec nos doigts, parce que nos doigts seraient couverts de trucs gluants. Après, METAFOOD nous aiderait à planifier une liste d'achats, etc.

Bon appétit !

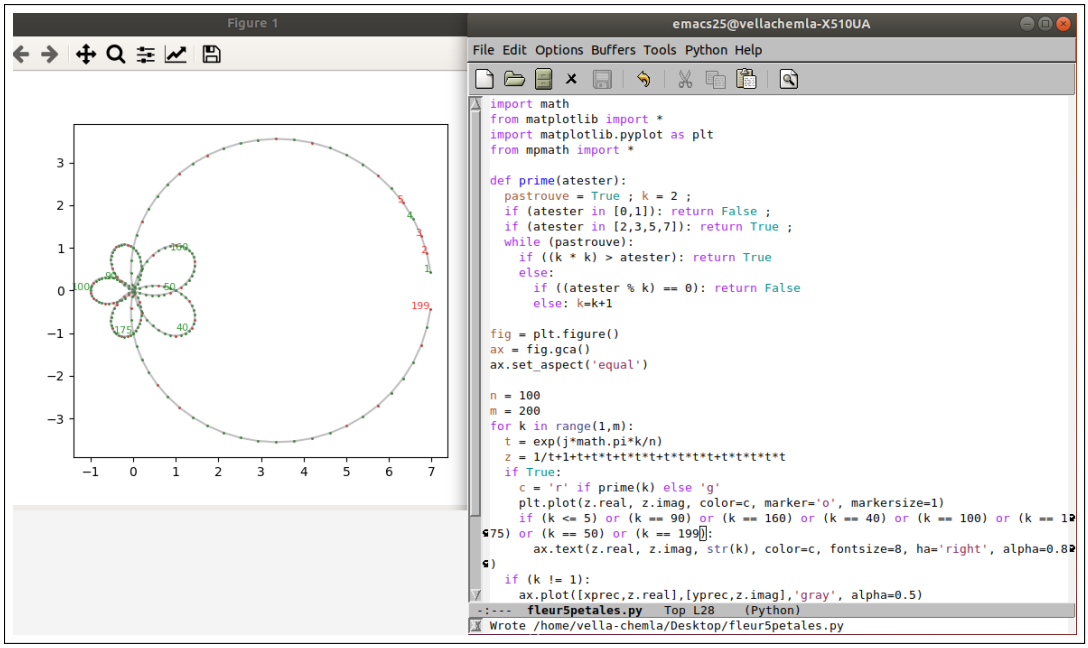
Donald B. Knuth
Stanford, California
Autumn 2019

Petite étape (Denise Vella-Chemla, 10.10.2020)

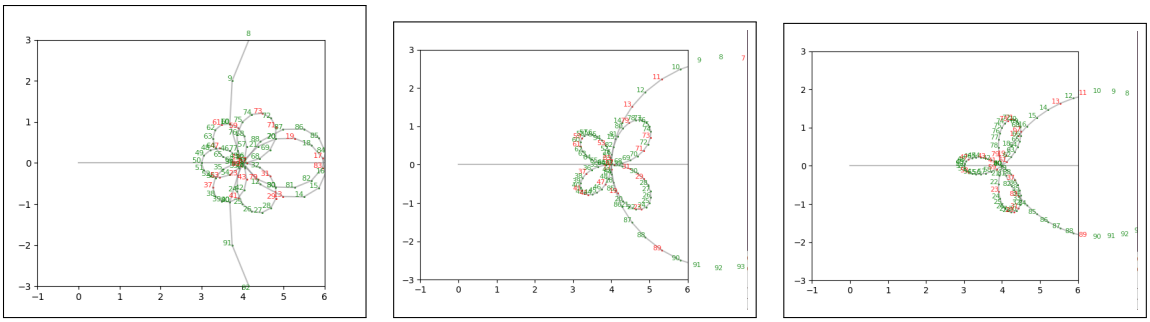
On fait toujours varier le même programme fourni dans une note précédente, pour essayer de comprendre pourquoi les images des entiers par nos fonctions sont ce qu'elles sont : on a trois paramètres sur lesquels on joue pour faire varier les graphiques :

- k, qui varie de 1 à 100, avec quelques variantes ;
- t, qui est en général de la forme $e^{i\pi k/n}$;
- z, le complexe dessiné, dont on "suit le parcours" (on relie les images successives, pour dessiner un lacet), et qui est en général, avec quelques variantes, égal à $\frac{1-t^m}{1-t} = \sum_{k=0}^{k=m-1} t^k$.

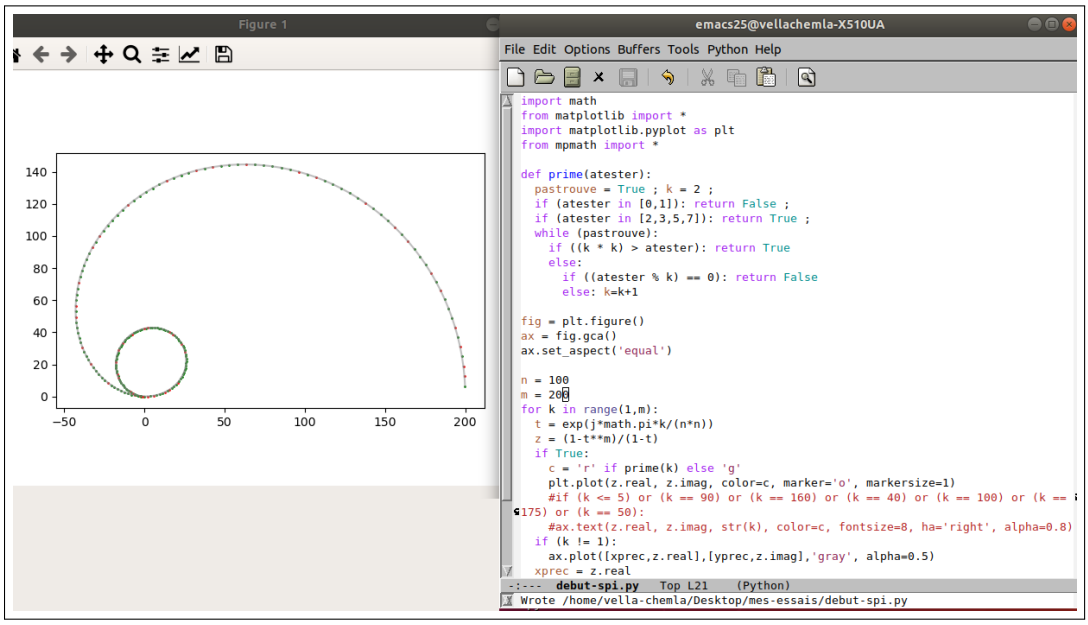
Voyons d'abord le bracelet avec fleur : on l'obtient en prenant la somme de $1/t$ (qu'on dit "à gauche") à t^5 (à droite).



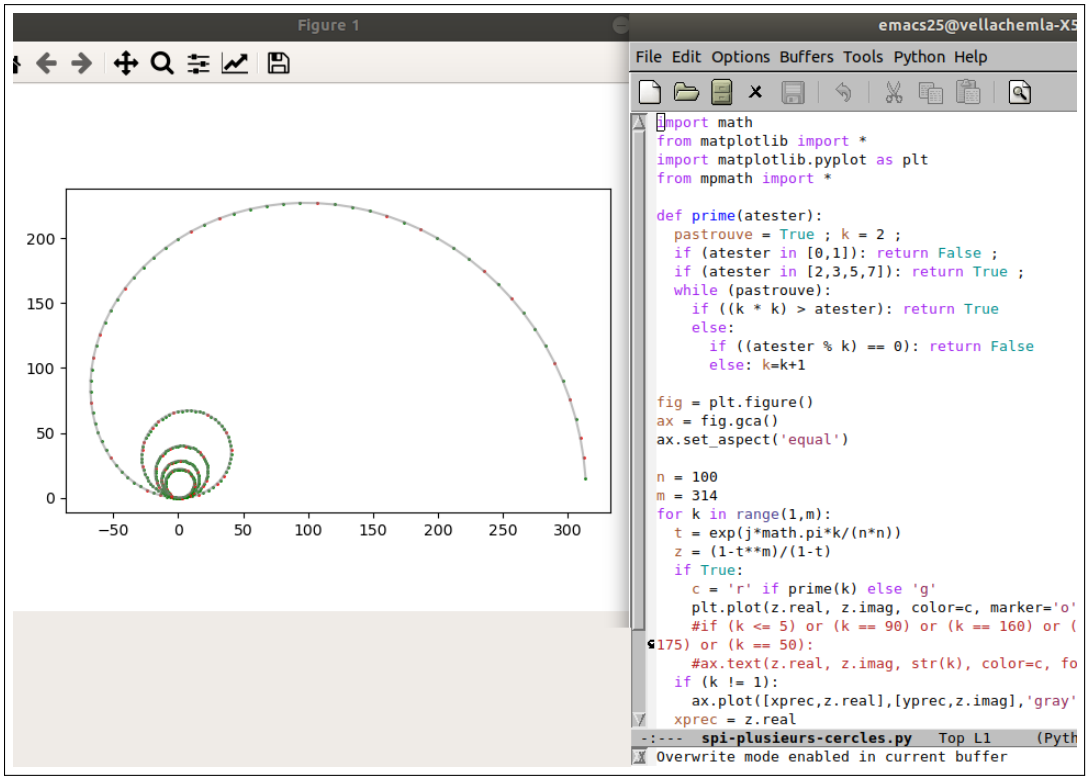
On peut jouer sur le nombre de pétales en changeant la puissance la plus grande.



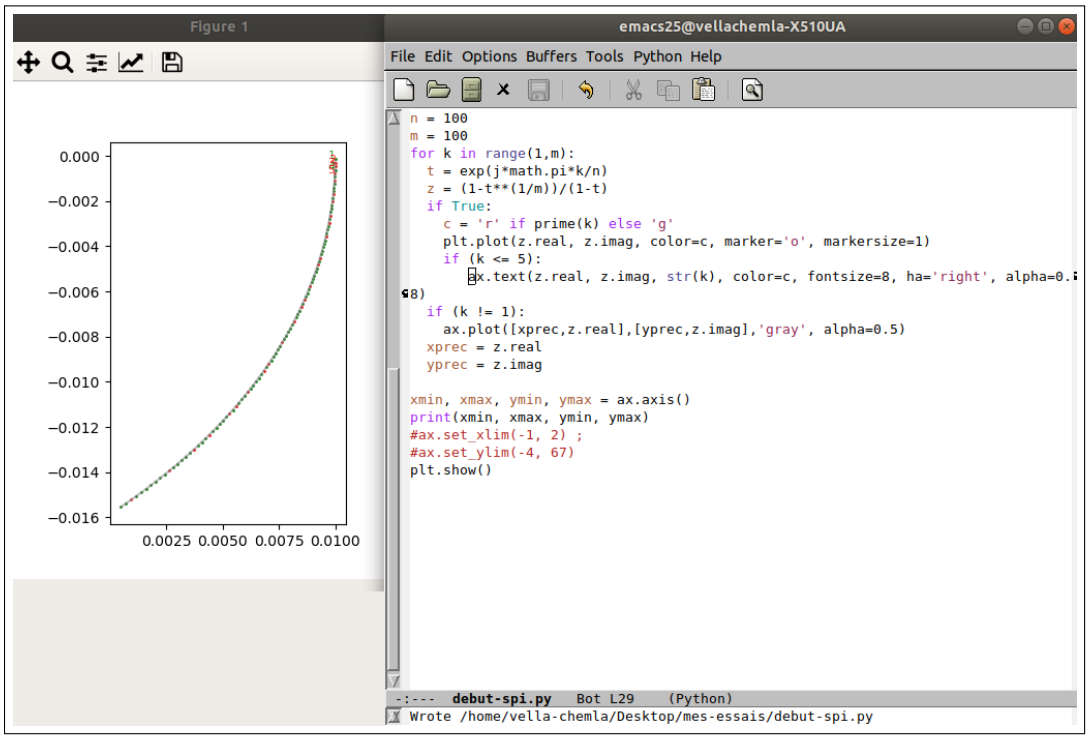
On veut comprendre pourquoi la fleur est au milieu du bracelet, pourquoi le bracelet n'est pas lui-même une seule grosse fleur, pourquoi il y a le haut du cercle, le bas du cercle, et la fleur au milieu à gauche. Alors, on réussit à obtenir un début de spirale, voire même un premier cercle en bas.



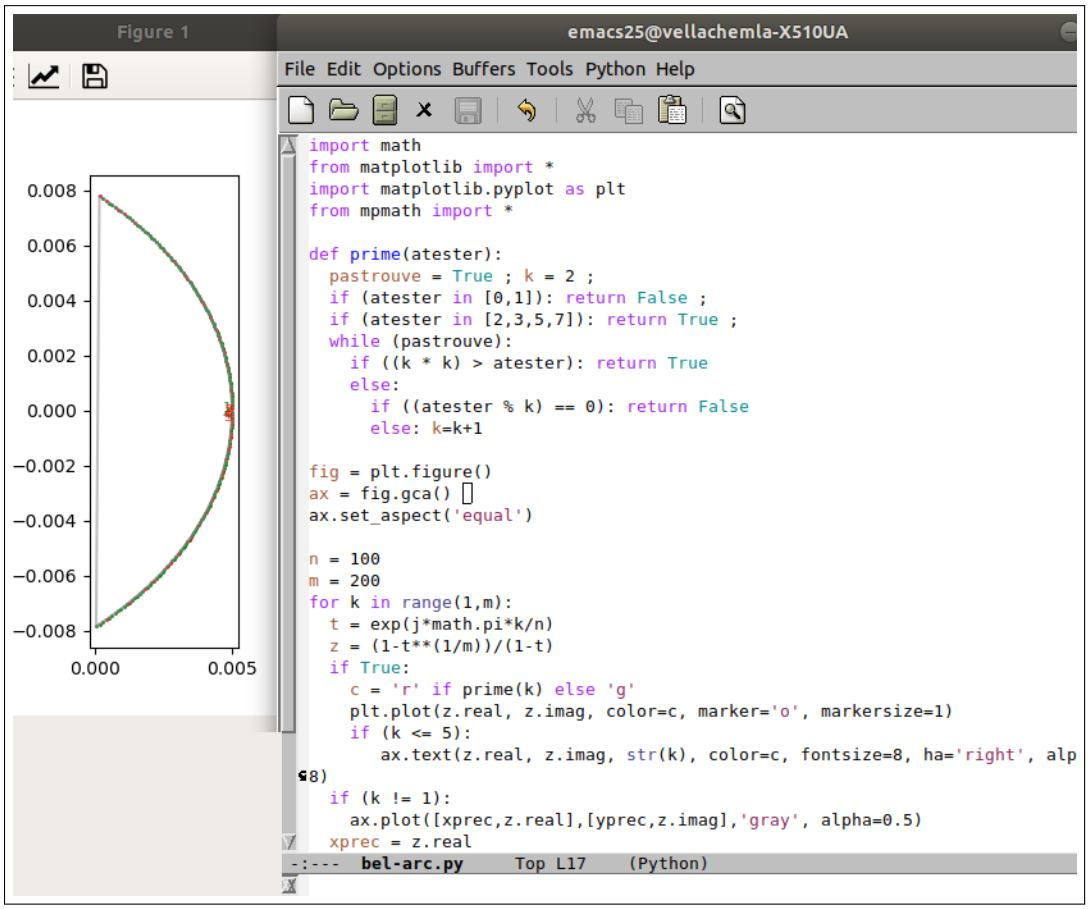
Puis on obtient en allant plus loin des cercles supplémentaires.



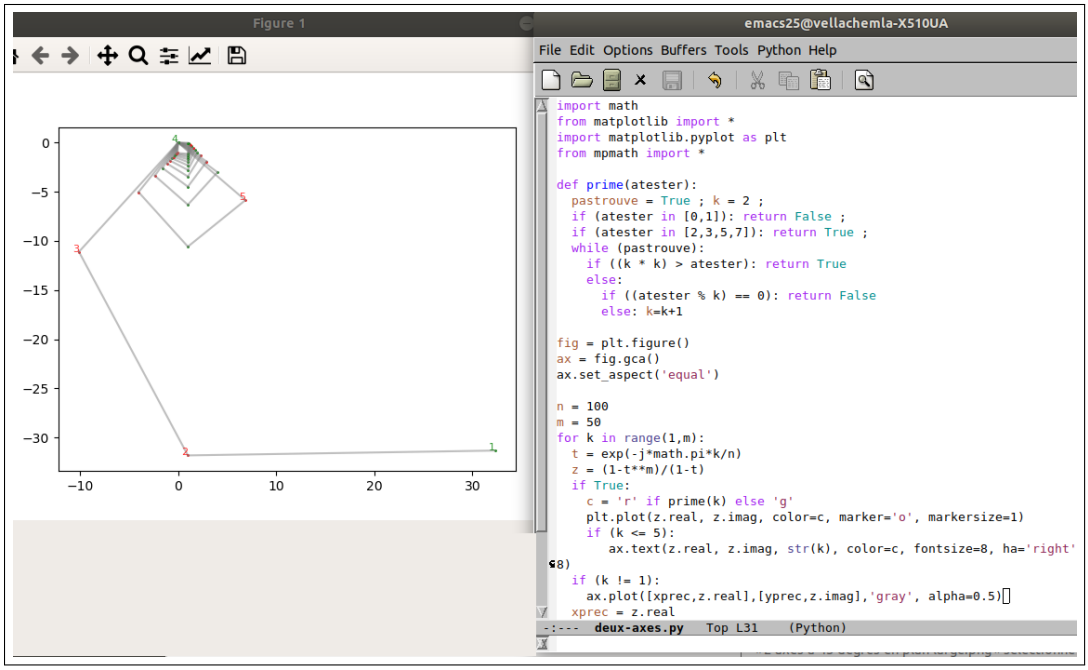
De façon surprenante, quand on met l'inverse de l'exposant, dans le calcul de la somme, la spirale part "à l'envers" (les images des nombres de 1 à 5 sont étiquetées en haut à droite du graphique).



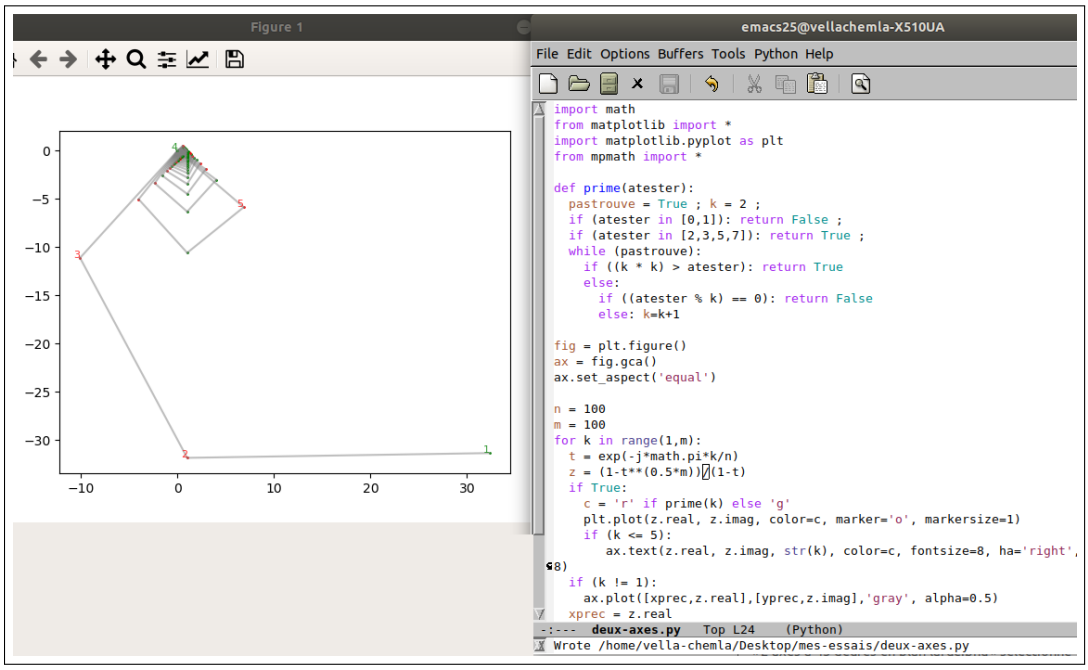
On pense “ok, toi, je vais te faire faire tout le tour”, et là, gros couac : ça dessine un arc, la corde de l’arc est dessinée entre 101 et 102, le fait d’avoir inversé l’exposant a bien sûr (sic !) rendu les valeurs minuscules.

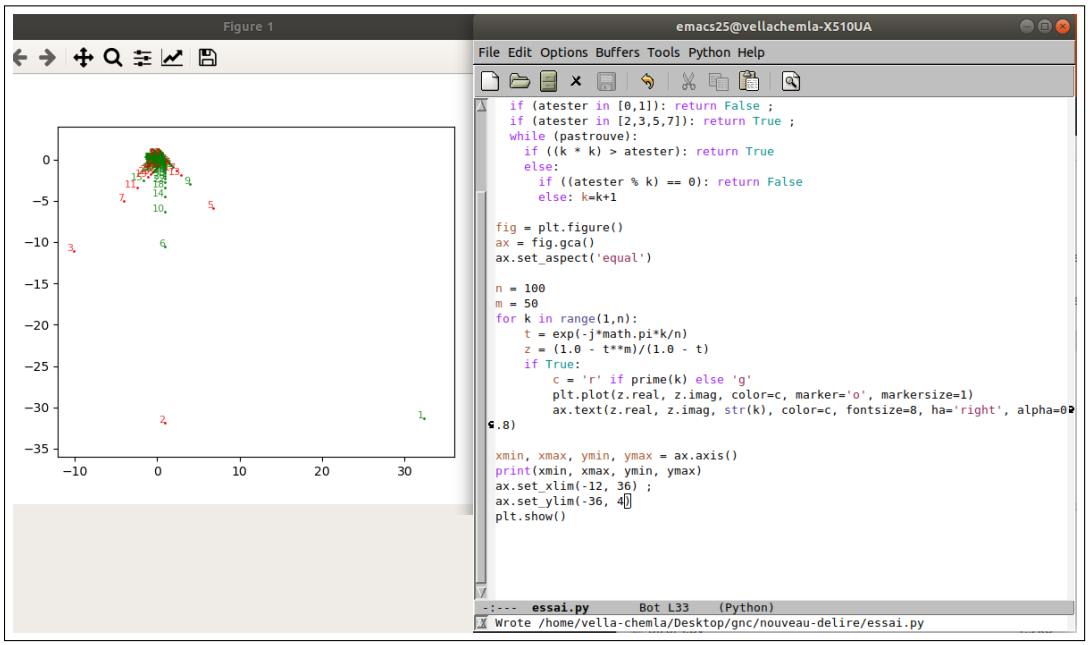


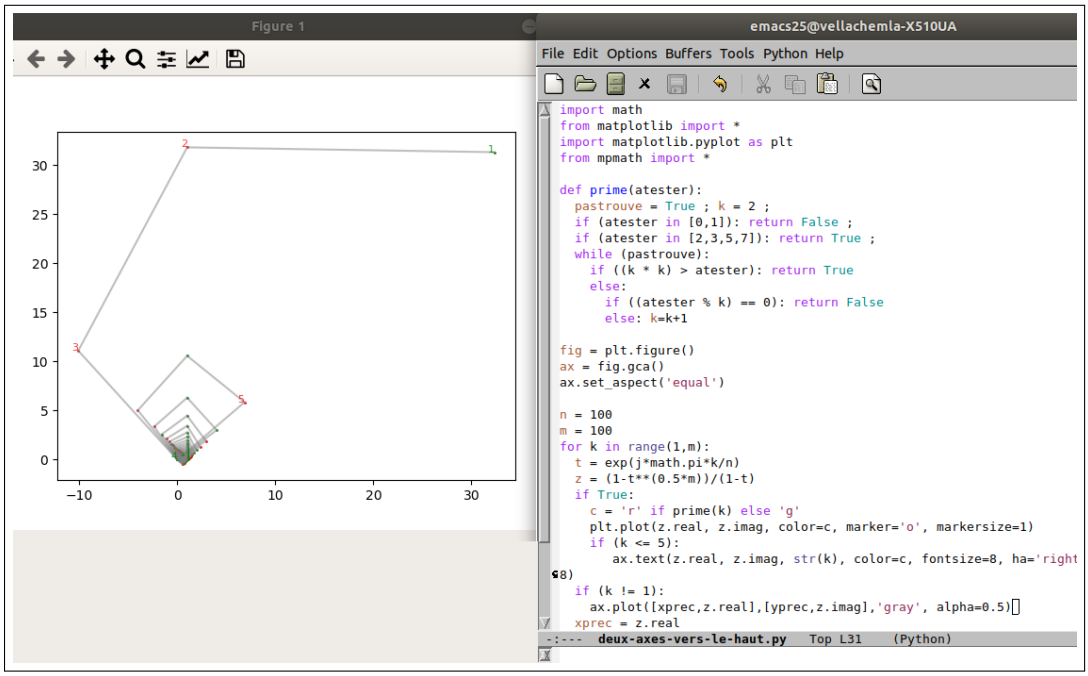
On a réussi par ci, par là, à mettre les nombres des formes $4x+1$ et $4x+3$ sur deux axes perpendiculaires à 45 degrés.



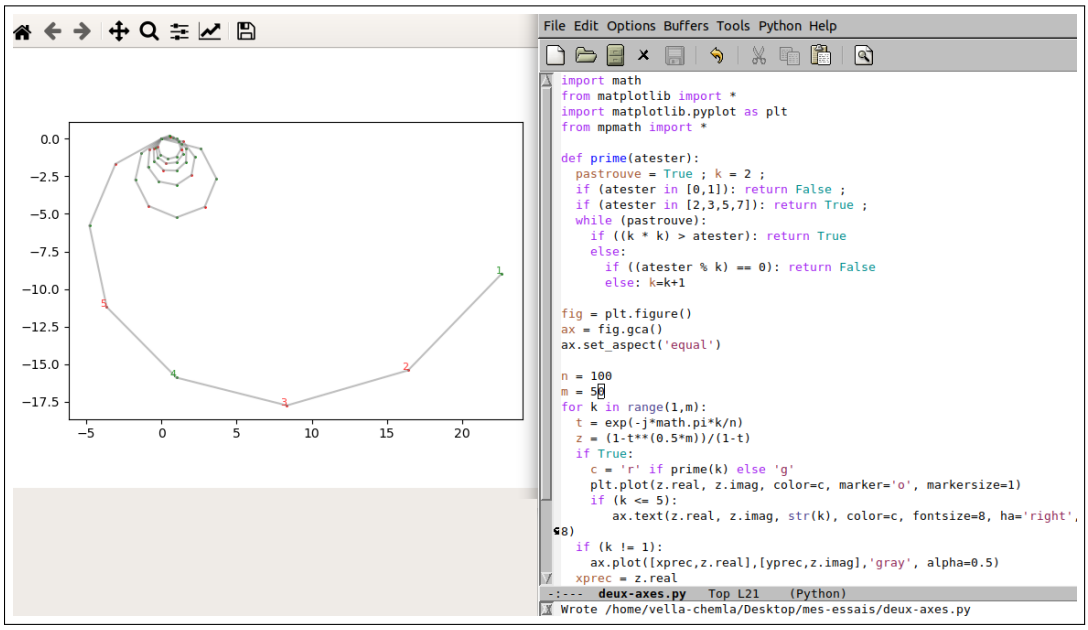
En modifiant les paramètres, on peut faire que les nombres soient “entrants” ou “sortants” sur les axes, qu’ils parcourent les axes vers le bas, ou vers le haut, de multiples variations sont bien sûr possibles et contrôlables. Sur les deux dessins suivants, on obtient un même résultat en utilisant deux programmes différents : dans le premier cas, on prend comme dernier terme de la somme le 50-ième tandis que dans le second cas, on va bien jusqu’au 100-ième terme, mais en multipliant l’exposant par 1/2 au numérateur dans le calcul de la somme.



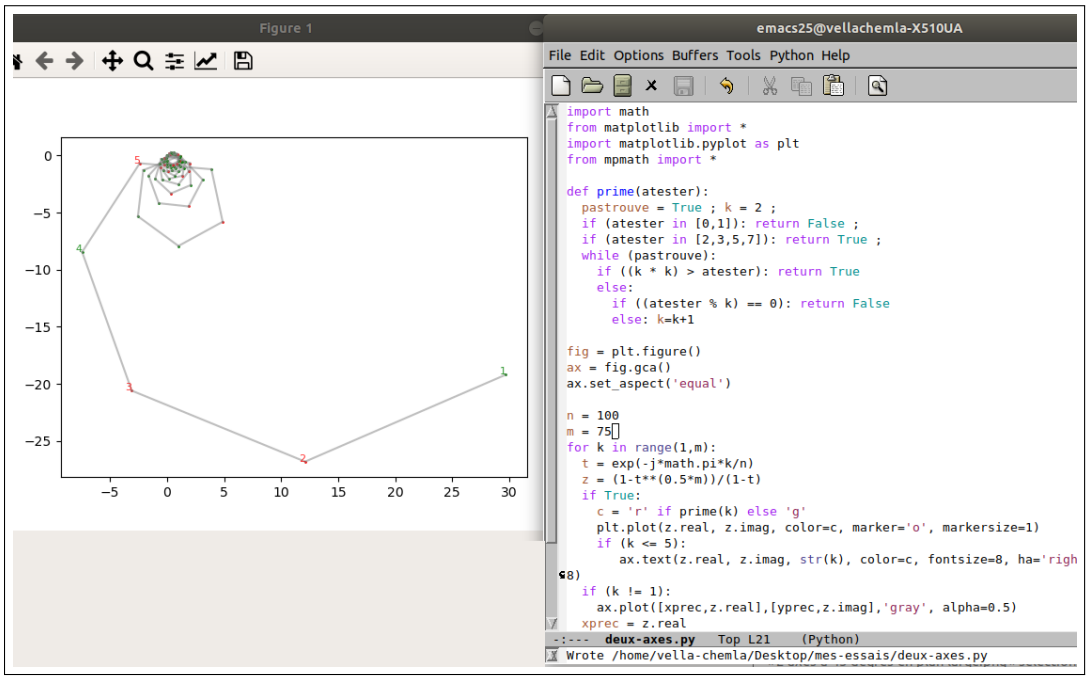




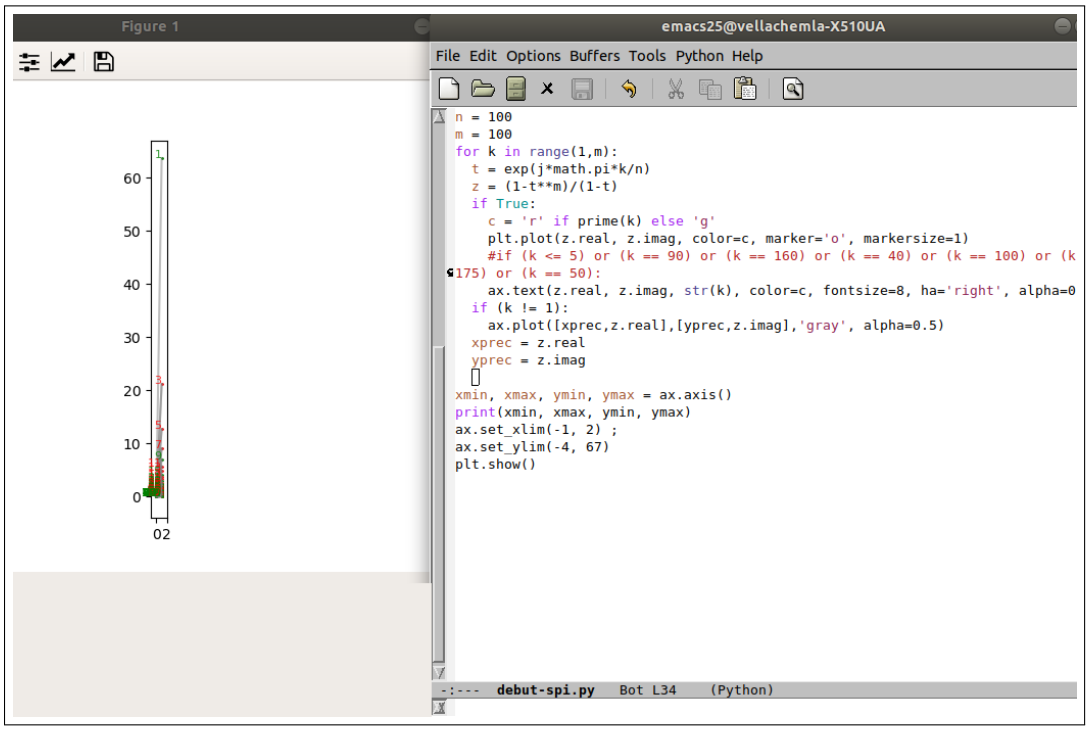
On peut multiplier les axes sur lesquels les nombres sont alignés. Ci-dessous, on distingue 6 axes, on a fait 3 variations : on n'est allé que jusqu'à $m = n/2$, on a pris la puissance opposée de e dans le calcul de t, on a pris la moitié de la puissance au numérateur dans le calcul de la somme...



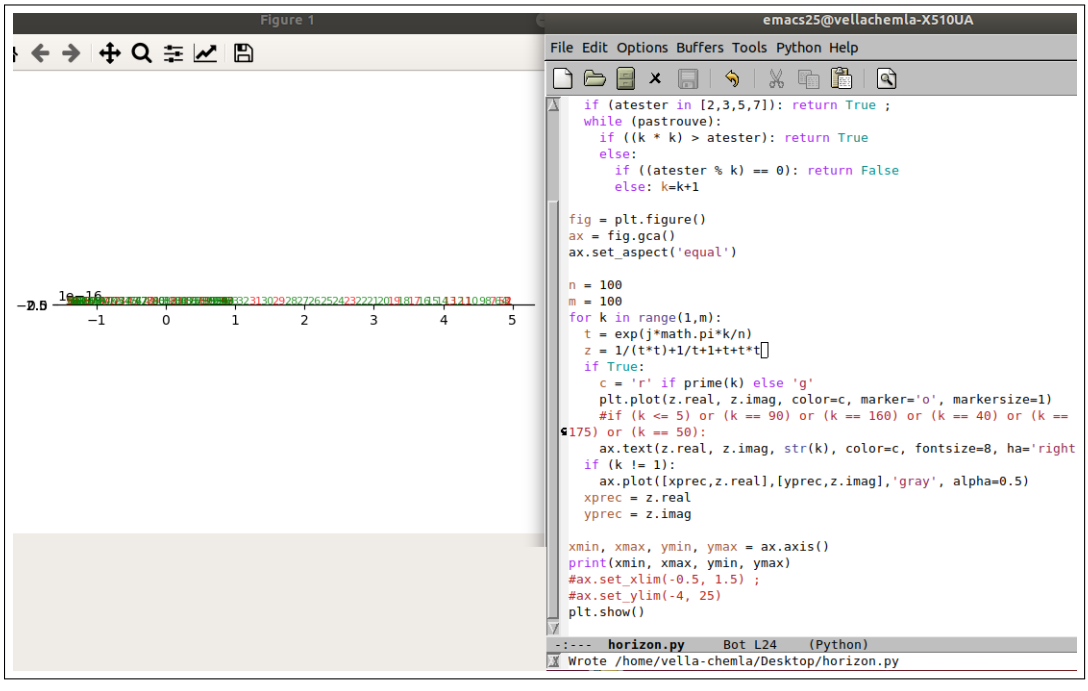
Il suffit parfois d'un rien pour décaler un graphique, lui conservant sa forme mais le "tordant" légèrement (là, le dernier sommant est le 75-ième).



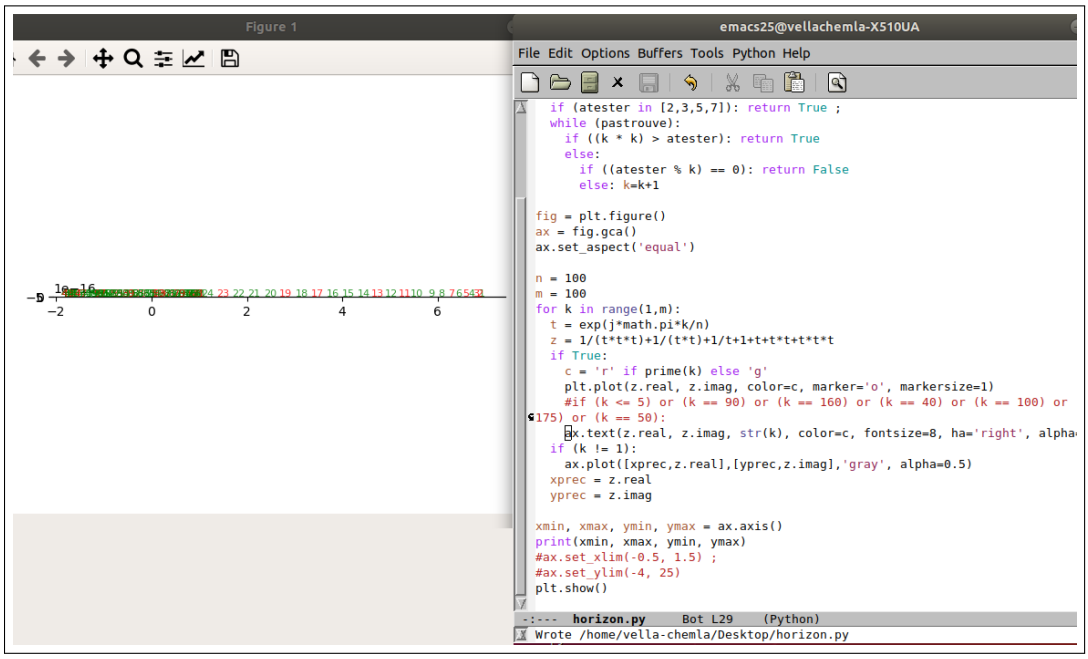
Le but reste l'alignement vertical, qu'on obtient en prenant $m = n$.



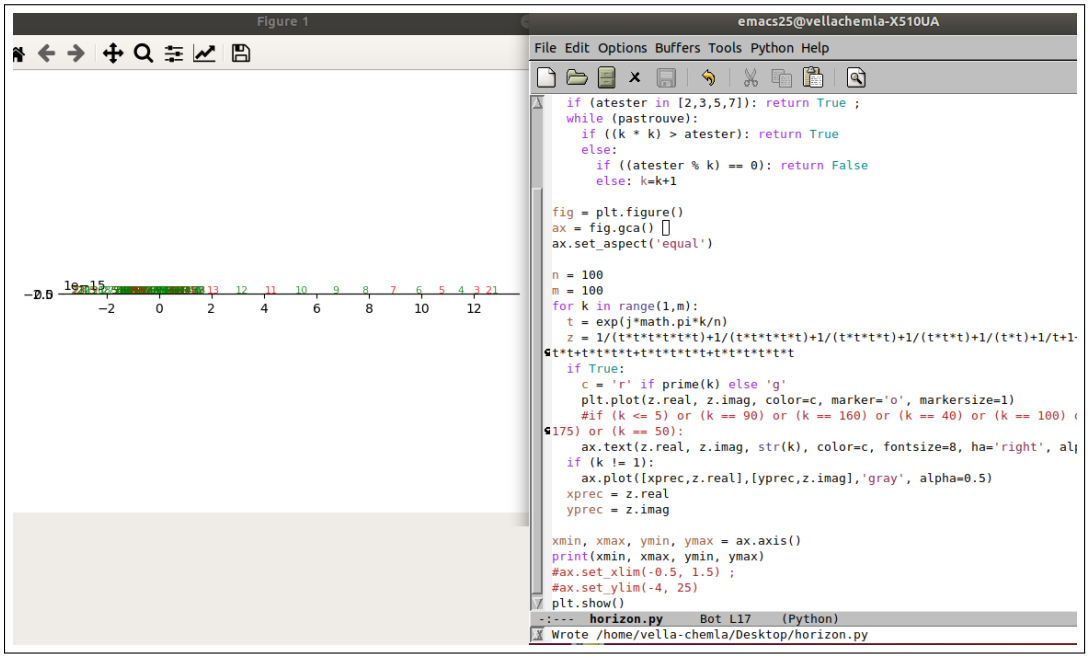
On obtient des alignements horizontaux en faisant varier les termes de la somme depuis l'inverse d'une certaine puissance (qu'on dira "à gauche"), jusqu'à la puissance en question (qu'on dira "à droite"), par exemple ici en sommant depuis $1/t^2$ (à gauche) jusqu'à t^2 (à droite) (ci-dessous, les nombres 1, 2 et 3, etc. partent depuis l'extrémité droite du dessin, vers la gauche).



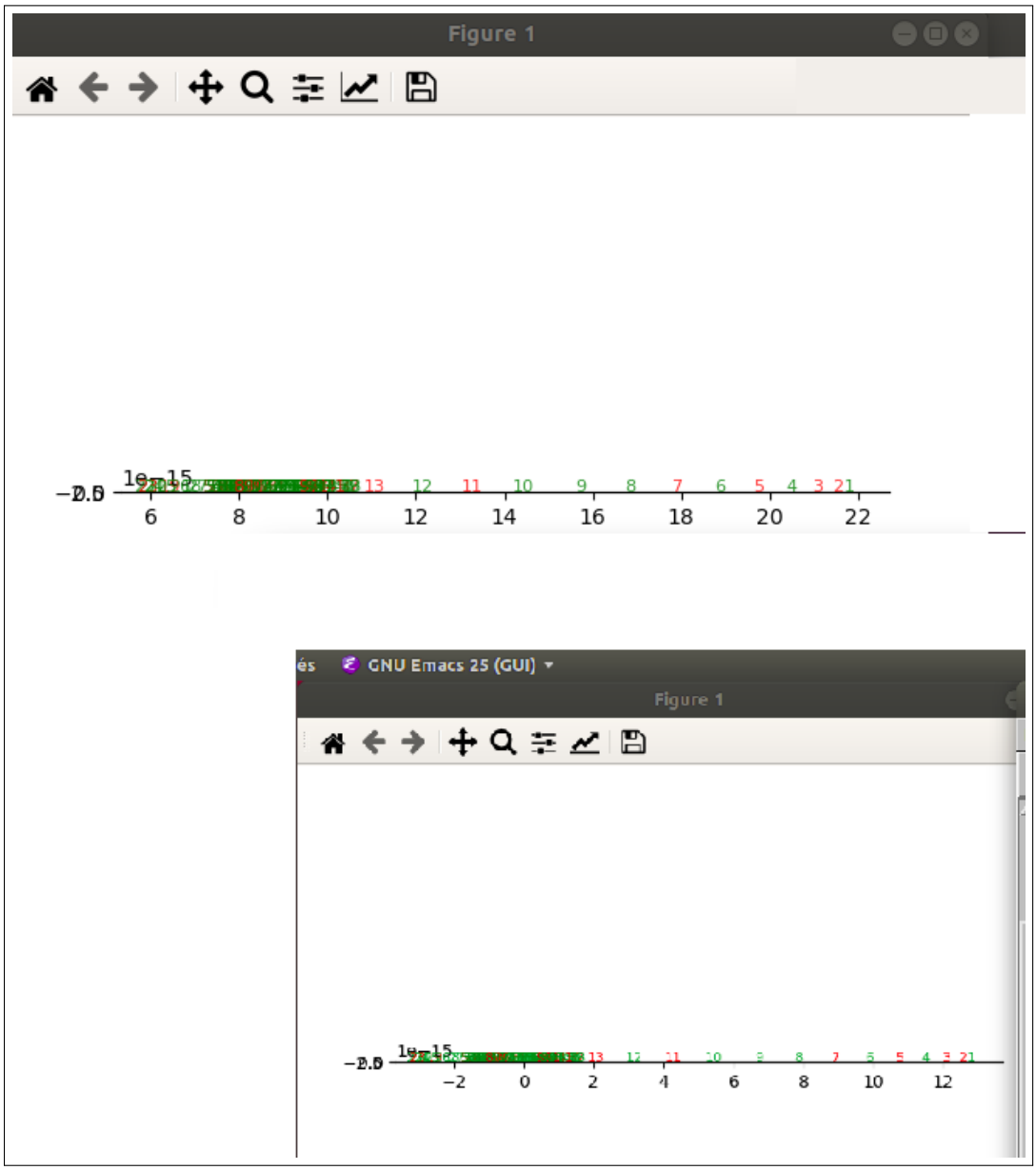
En allant "jusqu'aux cubes", les nombres sont plus écartés.



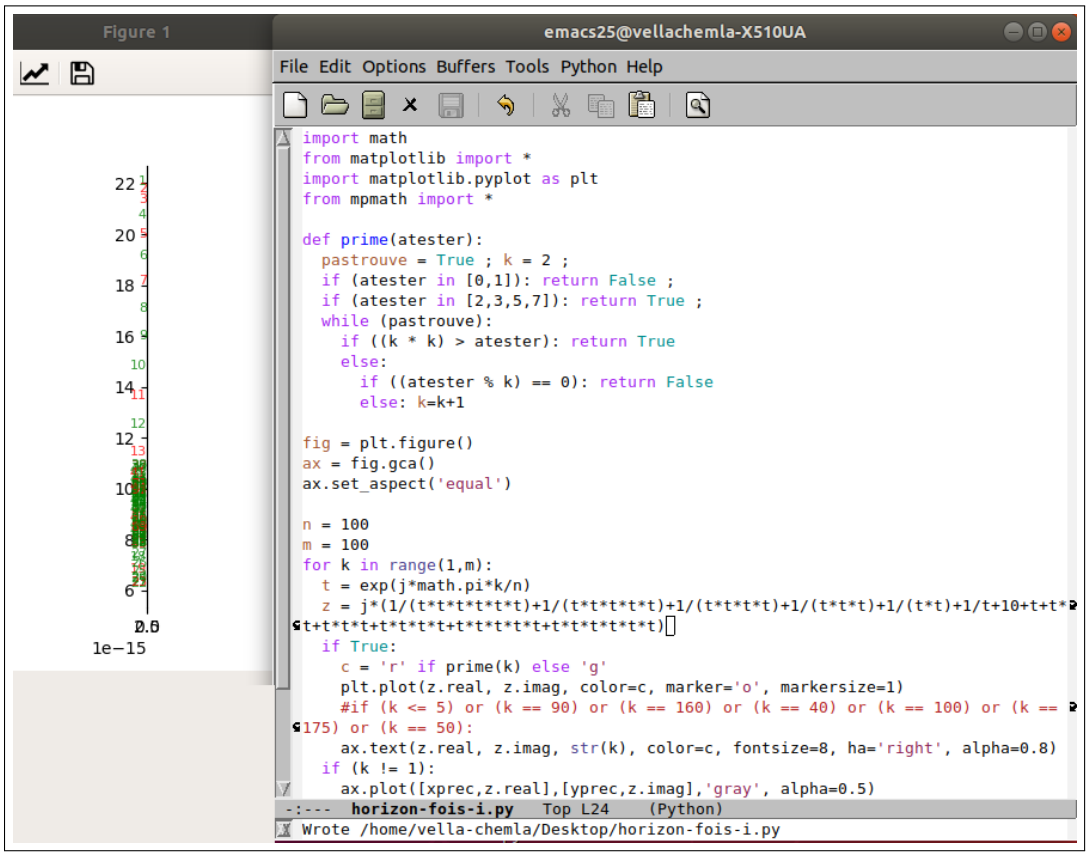
Même remarque jusqu'à la puissance 6 (côté inverse et côté puissances pures).



Si au milieu de la somme, on met bêtement un 2 au lieu de 1, ça translate, c'est tout.



Quand on a un alignement horizontal, on peut aussi très bêtement le transformer en alignement vertical en multipliant tout par i (ceci est une solution de facilité, sic).



On peut aussi obtenir des alignements sous-prétexte qu'on aurait pris l'exponentielle complexe de multiples de $2i\pi$, les deux dessins ci-après l'illustrent en montrant surtout la façon dont le fait de mettre n^3 ou bien n^4 au dénominateur de l'exposant fait chuter drastiquement les valeurs des images.

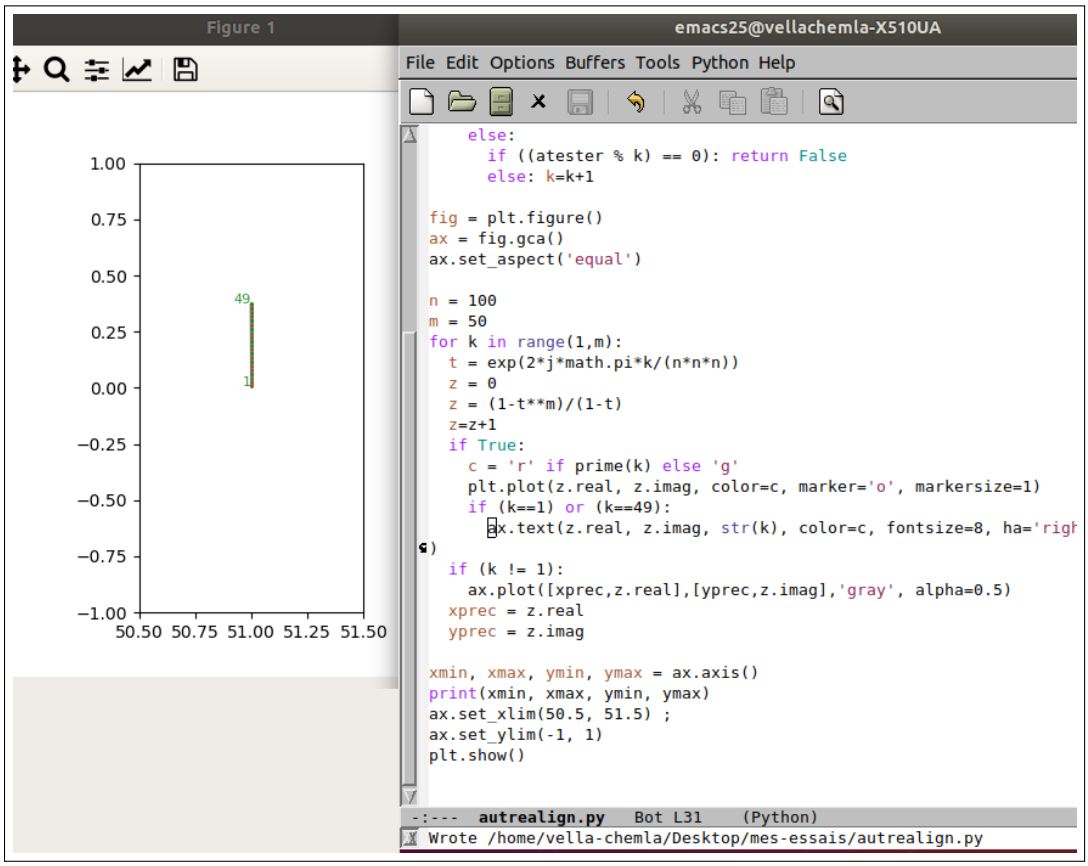


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
50.50 50.75 51.00 51.25 51.50
49
1
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 50
for k in range(1,m):
t = exp(2*j*math.pi*k/(n*n*n))
z = 0
z = (1-t**m)/(1-t)
z=z+1
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
if (k==1) or (k==49):
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='righ
t')
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
ax.set_xlim(50.5, 51.5) ;
ax.set_ylim(-1, 1)
plt.show()
-:--- autrealign.py Bot L31 (Python)
Wrote /home/vella-chemla/Desktop/mes-essais/autrealign.py

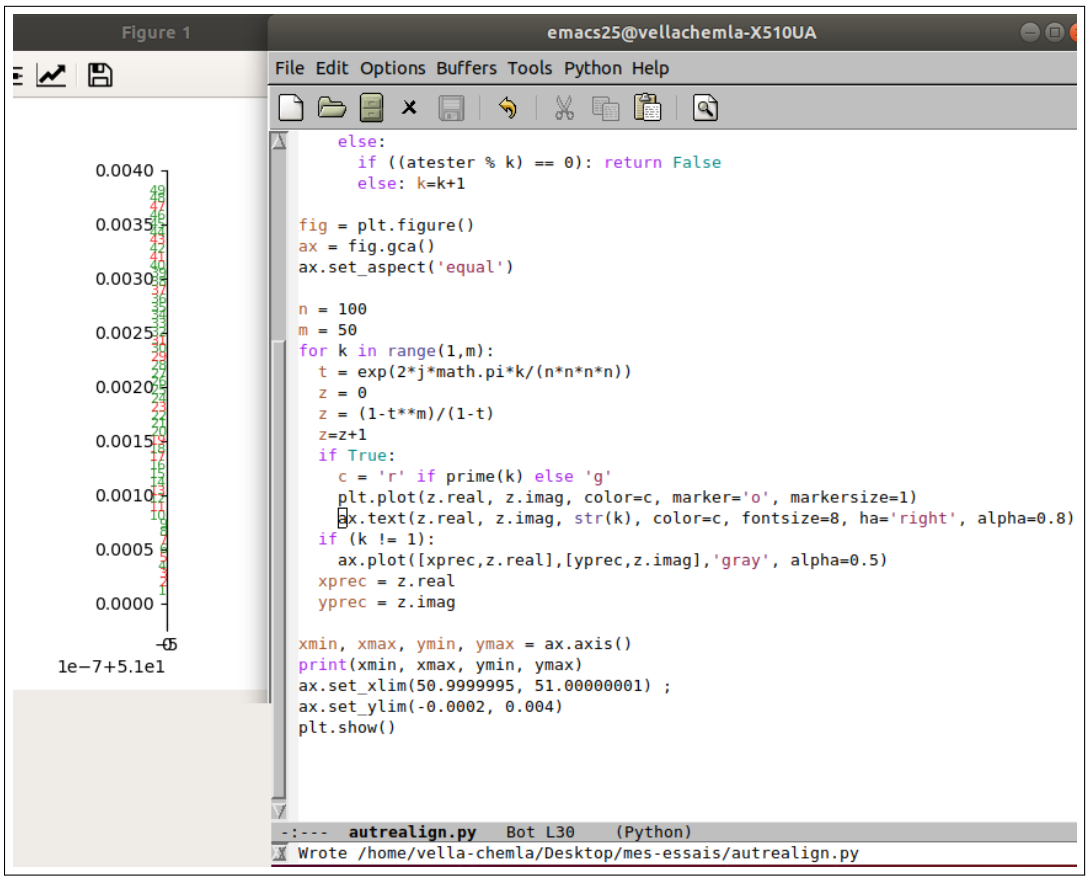
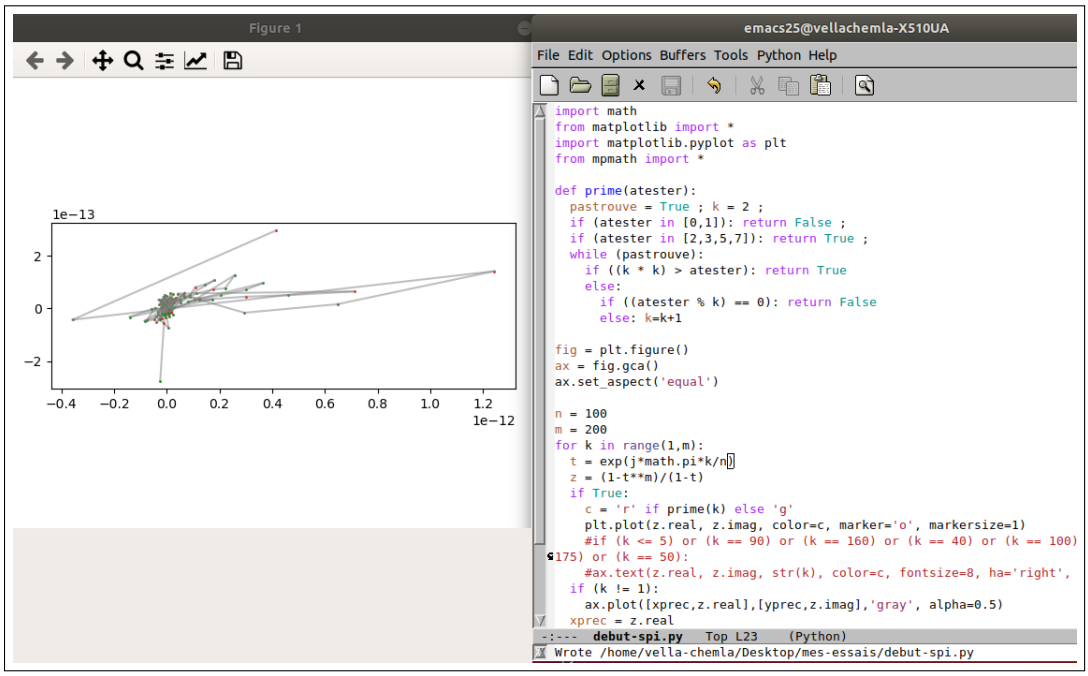
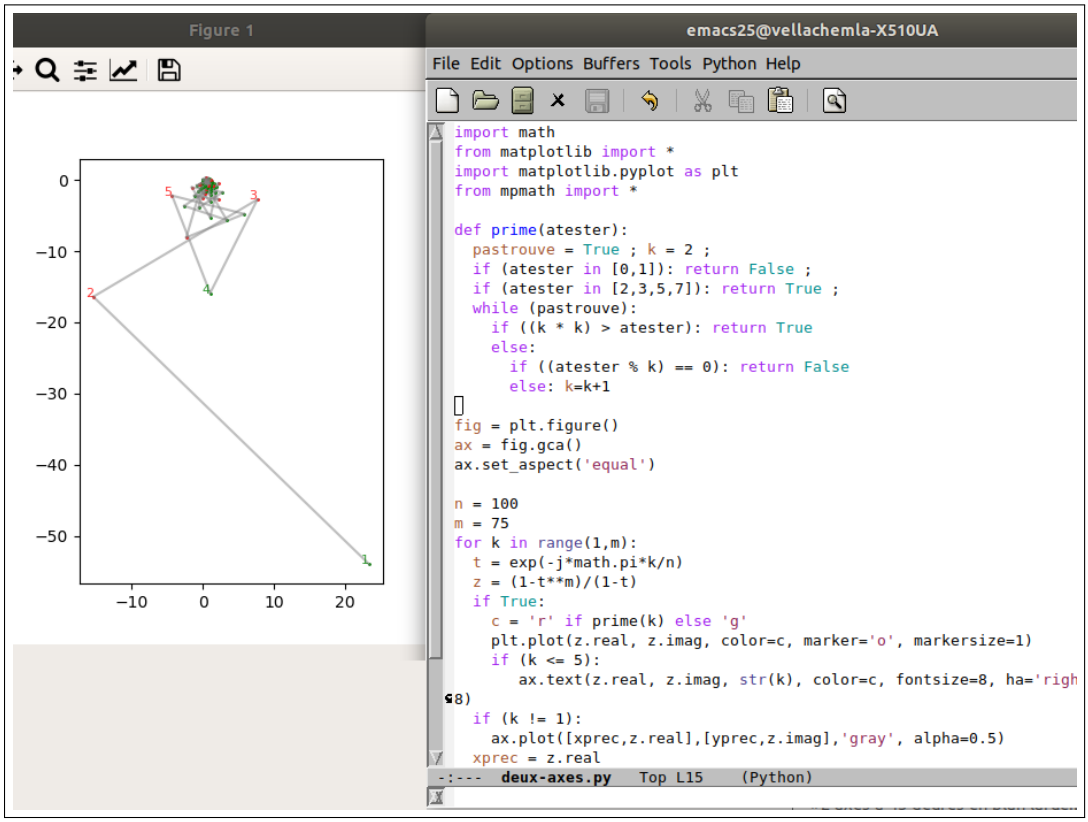


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
0.0040
0.0035
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
1e−7+5.1e1
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 50
for k in range(1,m):
t = exp(2*j*math.pi*k/(n*n*n*n))
z = 0
z = (1-t**m)/(1-t)
z=z+1
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
ax.set_xlim(50.9999995, 51.00000001) ;
ax.set_ylim(-0.0002, 0.004)
plt.show()
-:--- autrealign.py Bot L30 (Python)
Wrote /home/vella-chemla/Desktop/mes-essais/autrealign.py

De toute façon, il faut toujours garder à l'esprit qu'on n'est pas à l'abri d'un accident de parcours illisible, comme sur l'exemple 1 :

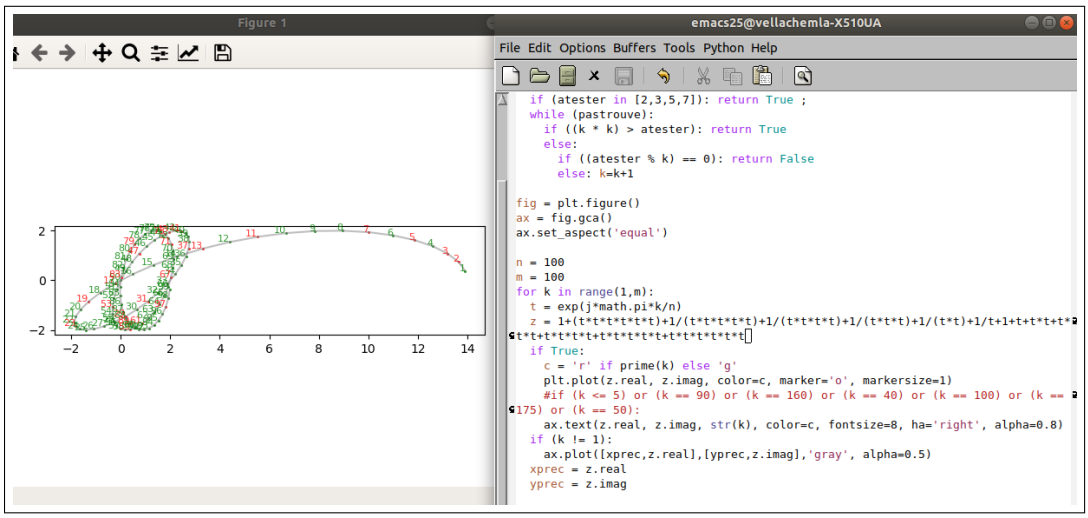


Même remarque à propos de l'exemple 2.



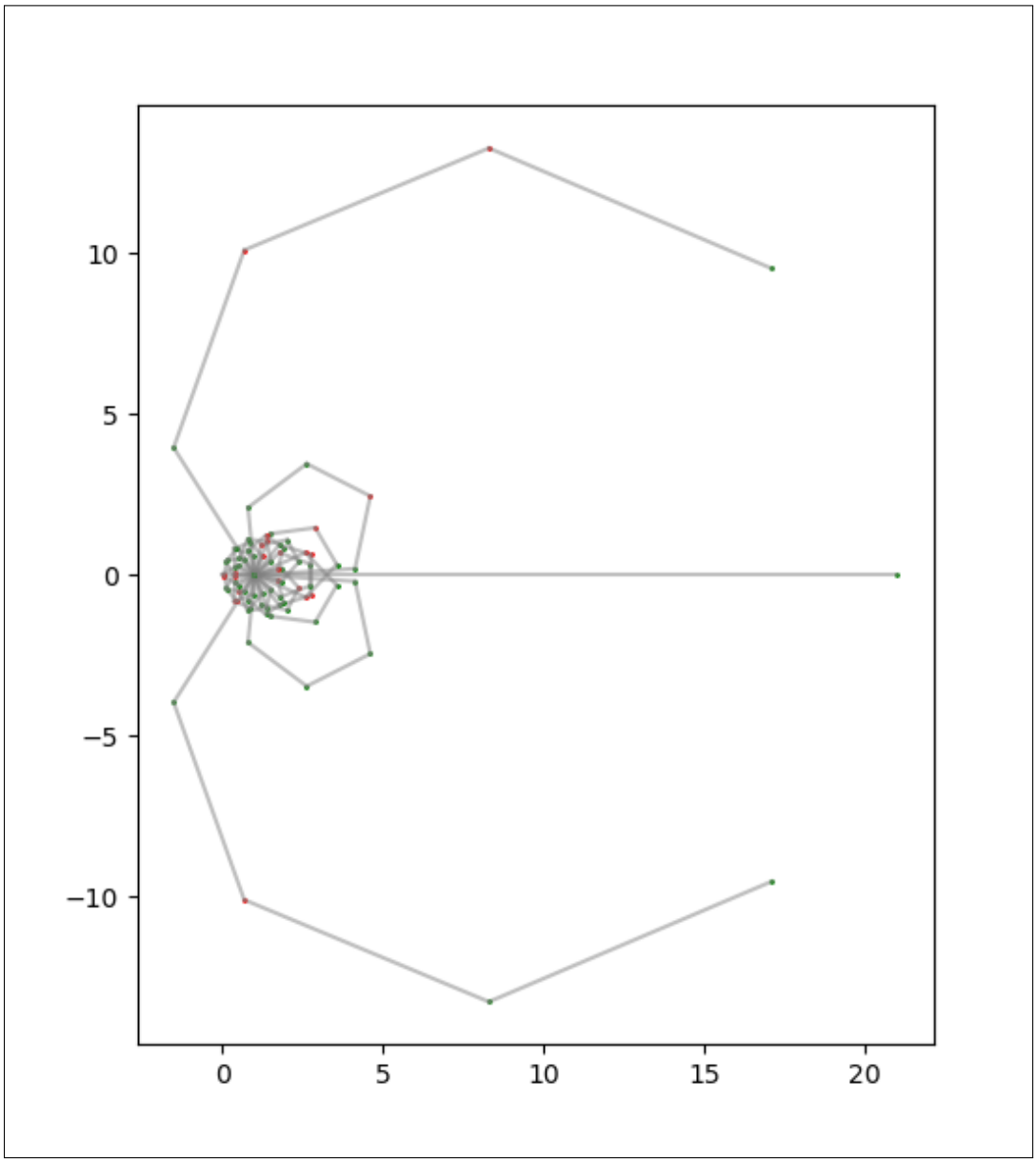
Merveilleux accident de parcours (regarder le code, un signe / est bêtement devenu un +) qui donne

naissance à une superbe signature !



Il faudra revenir sur celle qu'on a envie d'appeler l'araignée. Elle est plaisante parce qu'elle "rapproche très près" les décomposants de 100 et on distingue bien ici les décompositions de Goldbach de 100, deux points rouges proches. Peut-être tendraient-ils à être confondus...

```
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *

def prime(atester):
  pastrouve = True ; k = 2 ;
  if (atester in [0,1]): return False ;
  if (atester in [2,3,5,7]): return True ;
  while (pastrouve):
    if ((k * k) > atester): return True
    else:
      if ((atester % k) == 0): return False
      else: k=k+1

fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')

n = 100
m = 100
for k in range(1,m):
  t = exp(2*j*math.pi*k/n)
  z = 0
  for m in range(-1,19):
     z = z+t**m
  z=z+1
  if True:
    c = 'r' if prime(k) else 'g'
    plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
    ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
  if (k != 1):
    ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
  xprec = z.real
  yprec = z.imag

xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
ax.set_xlim(-3, 6) ;
ax.set_ylim(-3, 3)
plt.show()
```


10
5
0
−5
−10
0
5
10
15
20

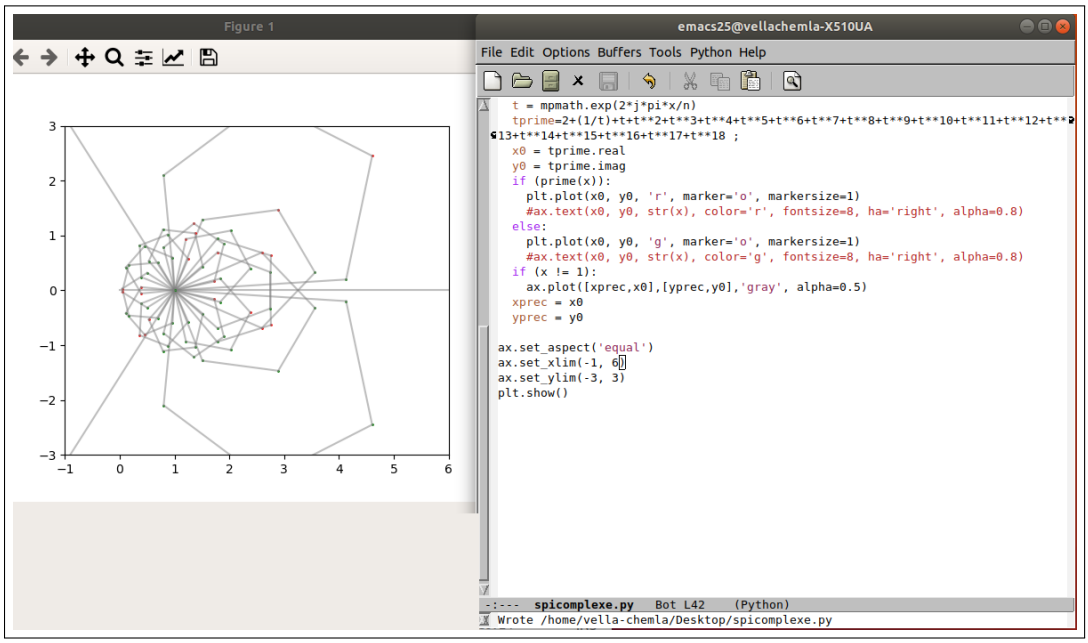
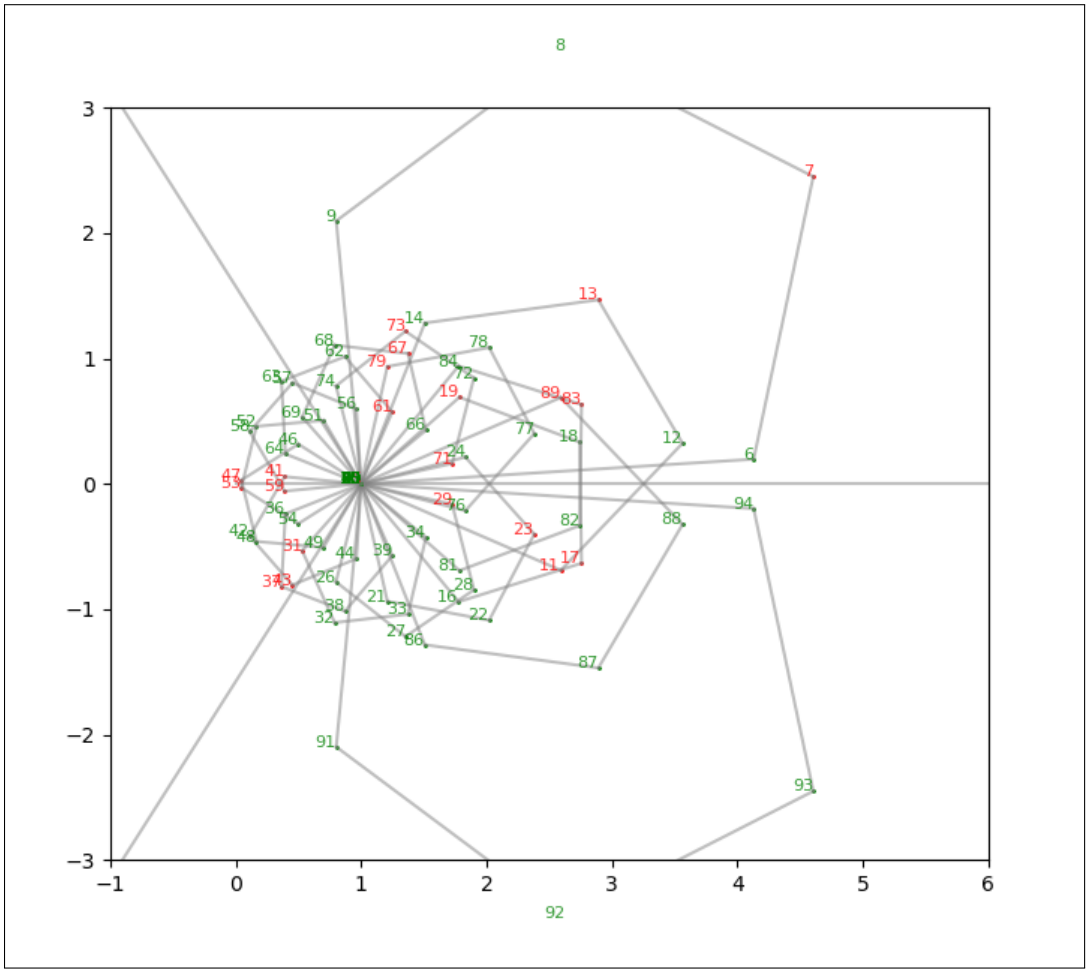


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
t = mpmath.exp(2*j*pi*x/n)
tprime=2+(1/t)+t+t**2+t**3+t**4+t**5+t**6+t**7+t**8+t**9+t**10+t**11+t**12+t**13+t**14+t**15+t**16+t**17+t**18 ;
x0 = tprime.real
y0 = tprime.imag
if (prime(x)):
plt.plot(x0, y0, 'r', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
else:
plt.plot(x0, y0, 'g', marker='o', markersize=1)
#ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
if (x != 1):
ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
xprec = x0
yprec = y0
ax.set_aspect('equal')
ax.set_xlim(-1, 6)
ax.set_ylim(-3, 3)
plt.show()
:--- spicomplexe.py Bot L42 (Python)
Wrote /home/vella-chemla/Desktop/spicomplexe.py


8
92

verif-eq-Tannery

October 18, 2020

[1]:
```
from sympy import *
from sympy.interactive import printing
from mpmath import j

printing.init_printing(use_latex=True)
u, t, du, dt = symbols('u t du dt')
x = -j*cos(u)*cos(t)
y = -j*cos(u)*sin(t)
z = 1 - cos(u/2) + sin(u/2)
dx = diff(x,u)*du + diff(x,t)*dt
dy = diff(y,u)*du + diff(y,t)*dt
dz = diff(z,u)*du + diff(z,t)*dt
display(dx)
display(dy)
display(dz)
ds2 = dx**2 + dy**2 + dz**2
display(simplify(ds2))
```

$$1.0idt \sin(t)\cos(u) + 1.0idu \sin(u)\cos(t)$$

$$-1.0idt \cos(t)\cos(u) + 1.0idu \sin(t)\sin(u)$$

$$du\left(\frac{\sin\left(\frac{u}{2}\right)}{2} + \frac{\cos\left(\frac{u}{2}\right)}{2}\right)$$

$$-1.0dt^2\cos^2(u) + 0.25du^2\sin(u) + 1.0du^2\cos^2(u) - 0.75du^2$$

[ ]:
```

```

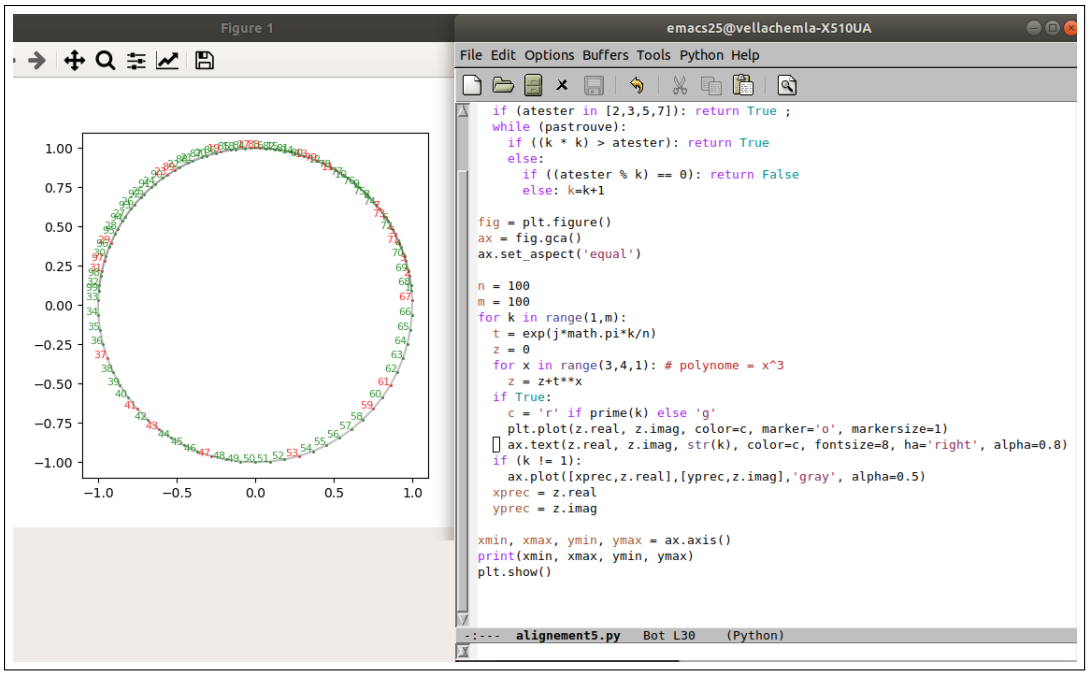


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(3,4,1): # polynome = x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
plt.show()
-:--- alignement5.py Bot L30 (Python)

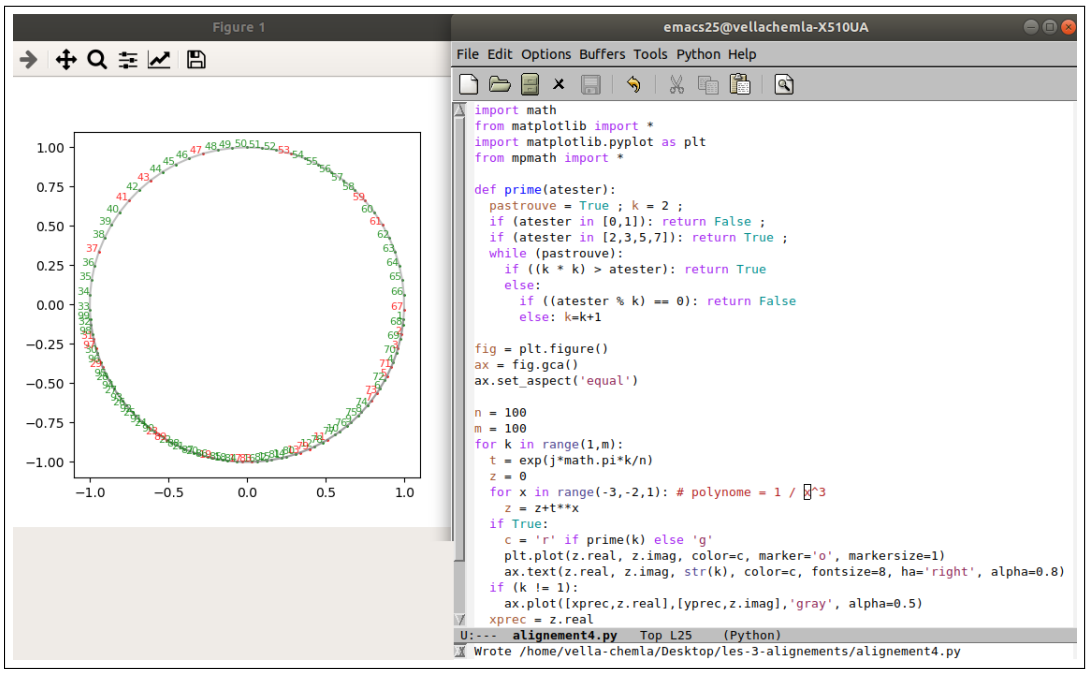


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-3,-2,1): # polynome = 1 / x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
U:--- alignement4.py Top L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement4.py

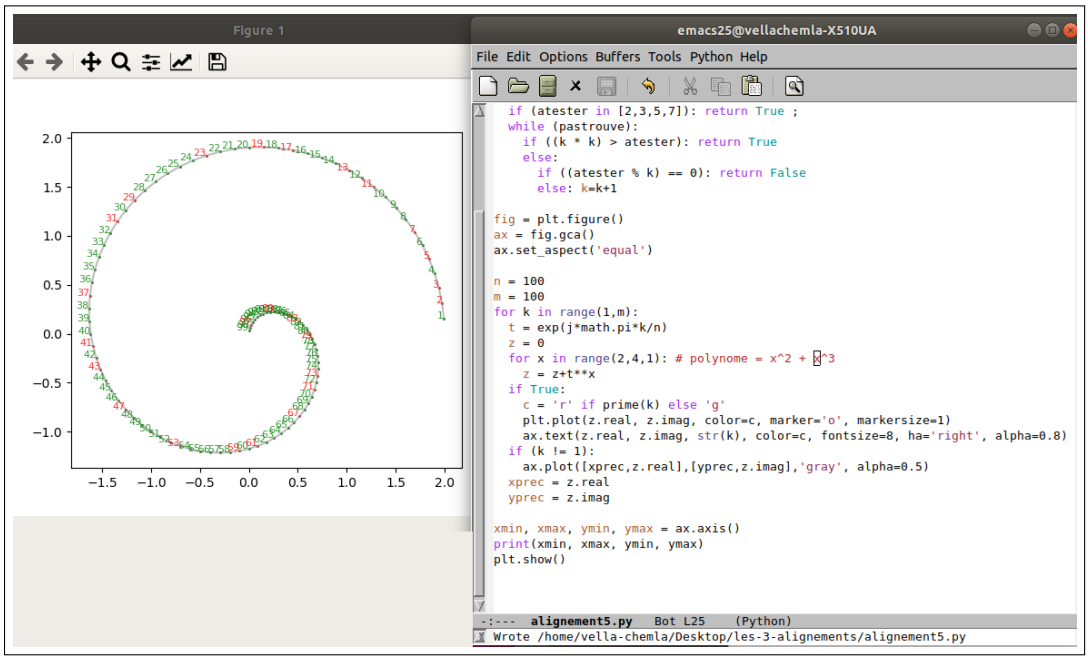


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(2,4,1): # polynome = x^2 + x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
plt.show()
-:--- alignement5.py Bot L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement5.py

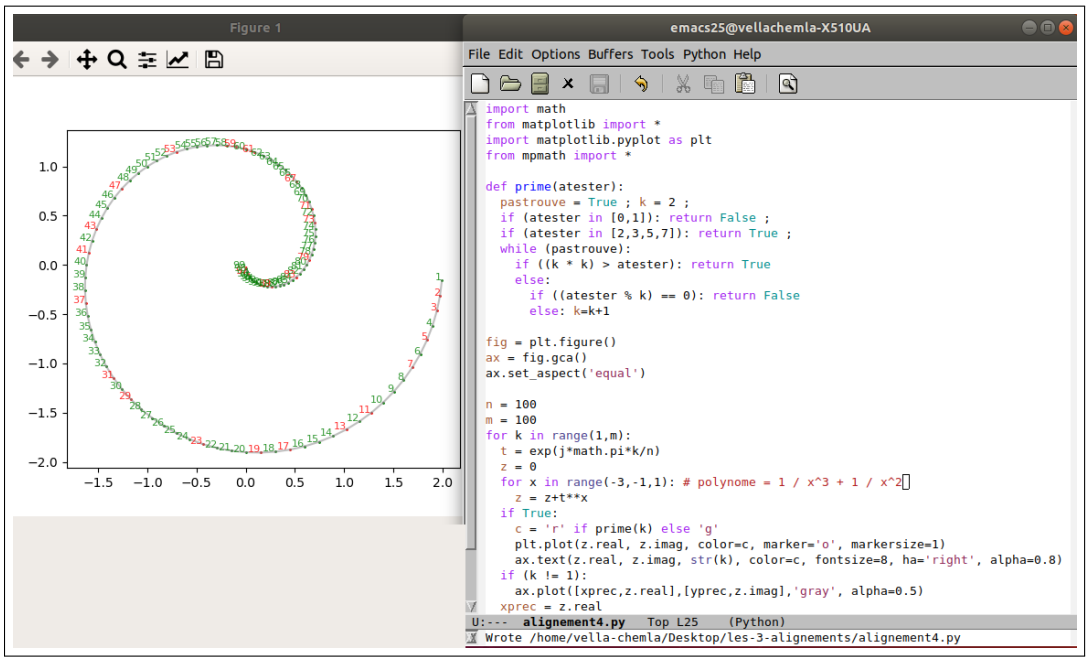


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-3,-1,1): # polynome = 1 / x^3 + 1 / x^2
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
U:--- alignement4.py Top L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement4.py

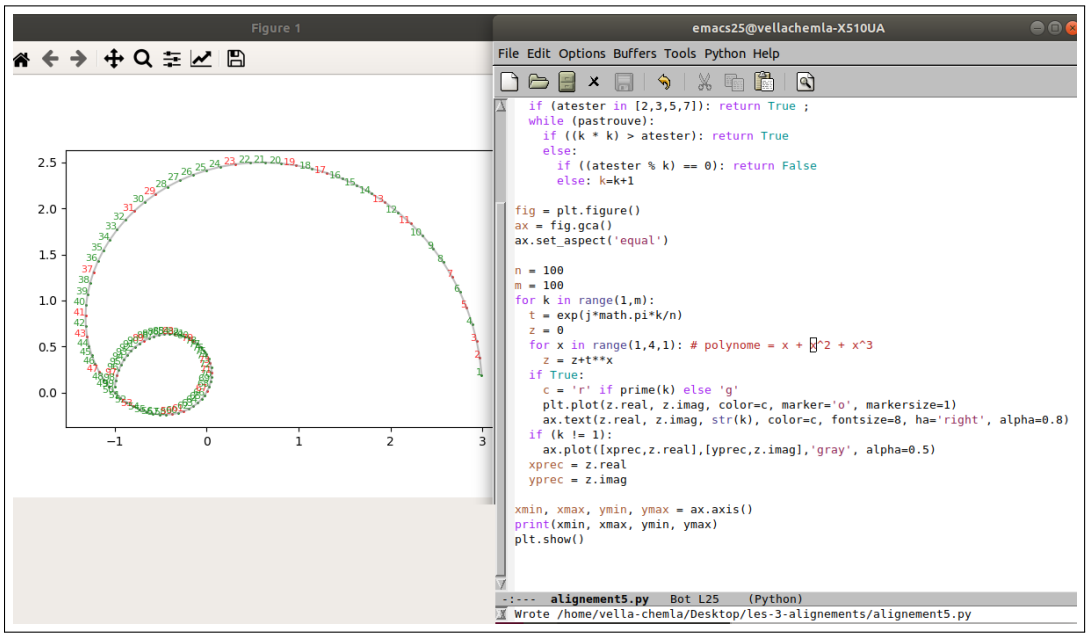


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(1,4,1): # polynome = x + x^2 + x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
plt.show()
-:--- alignement5.py Bot L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement5.py

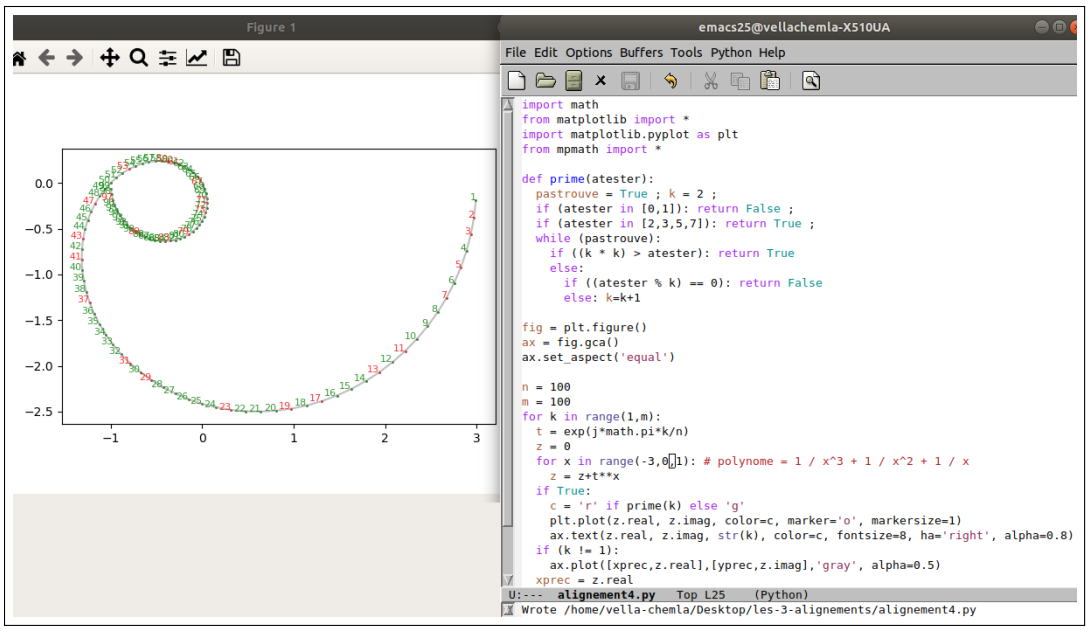


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-3,0,1): # polynome = 1 / x^3 + 1 / x^2 + 1 / x
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
U:--- alignement4.py Top L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement4.py

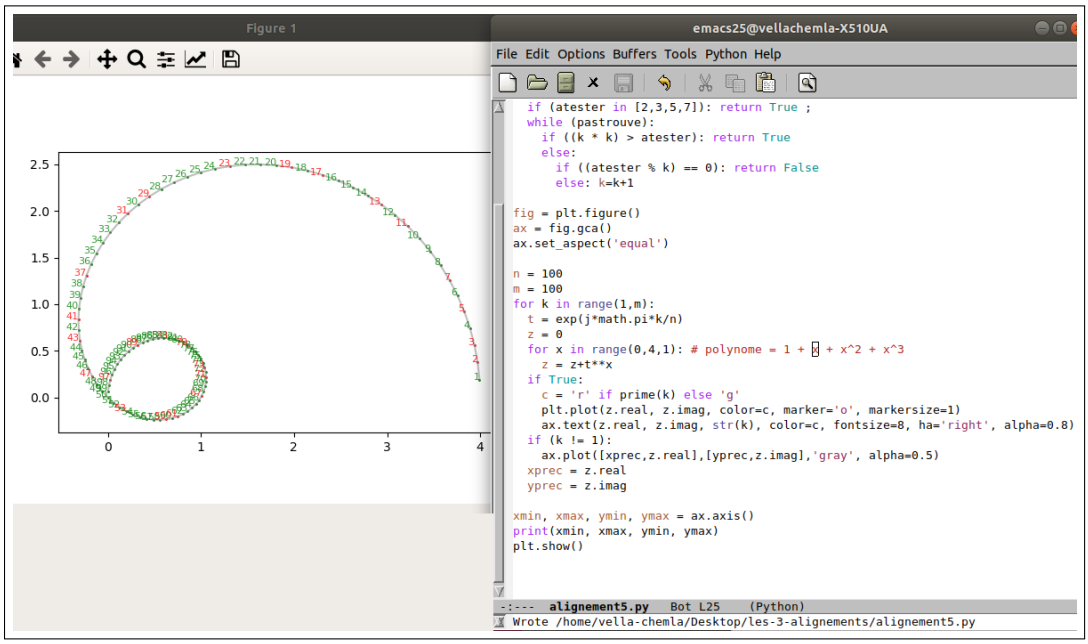


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(0,4,1): # polynome = 1 + x + x^2 + x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
plt.show()
-:--- alignement5.py Bot L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement5.py

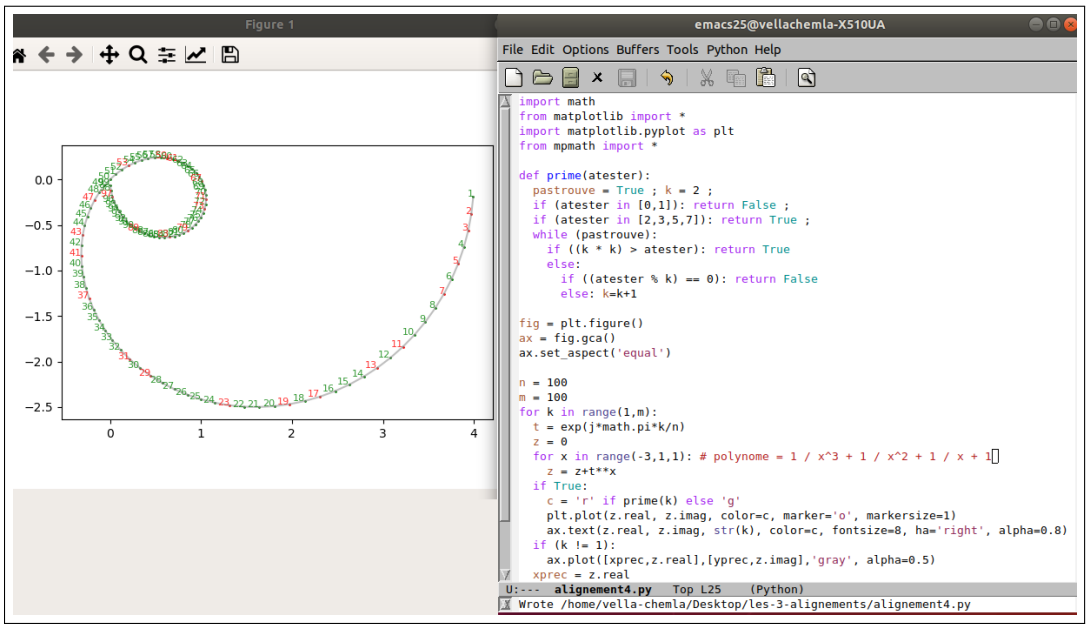


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-3,1,1): # polynome = 1 / x^3 + 1 / x^2 + 1 / x + 1
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
U:--- alignement4.py Top L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement4.py

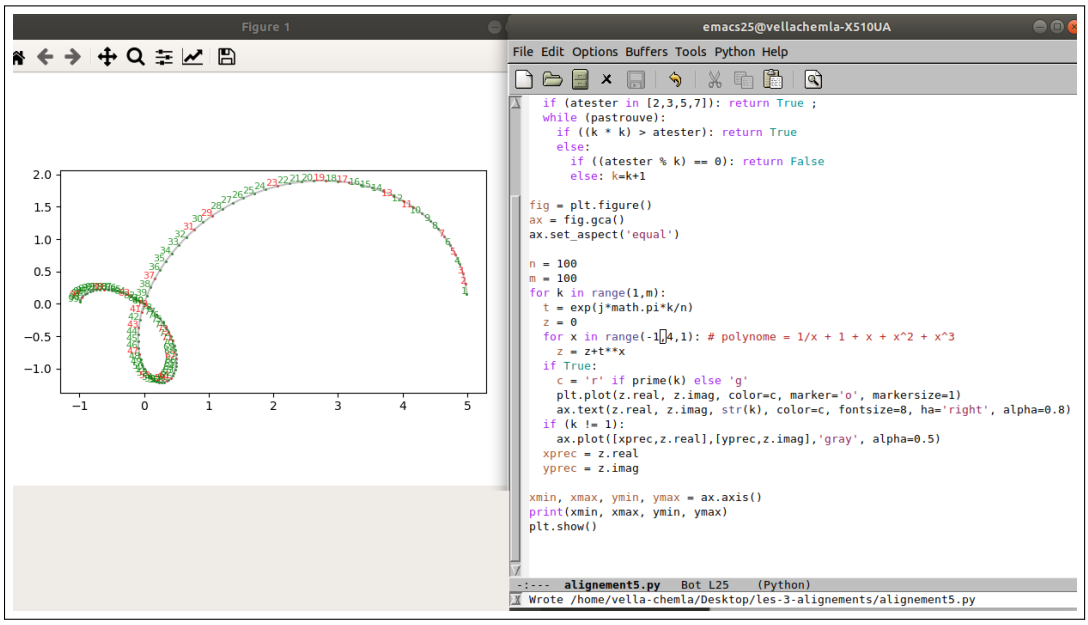


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-1[illegible]4,1): # polynome = 1/x + 1 + x + x^2 + x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
plt.show()
-:--- alignement5.py Bot L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement5.py

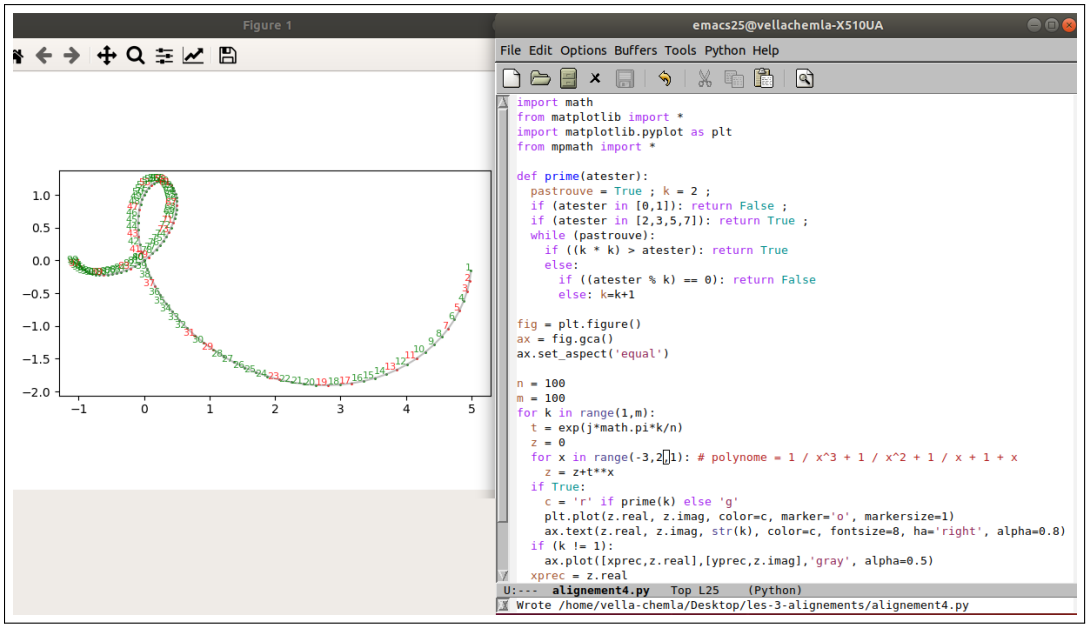


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-3,2[illegible]1): # polynome = 1 / x^3 + 1 / x^2 + 1 / x + 1 + x
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
U:--- alignement4.py Top L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement4.py

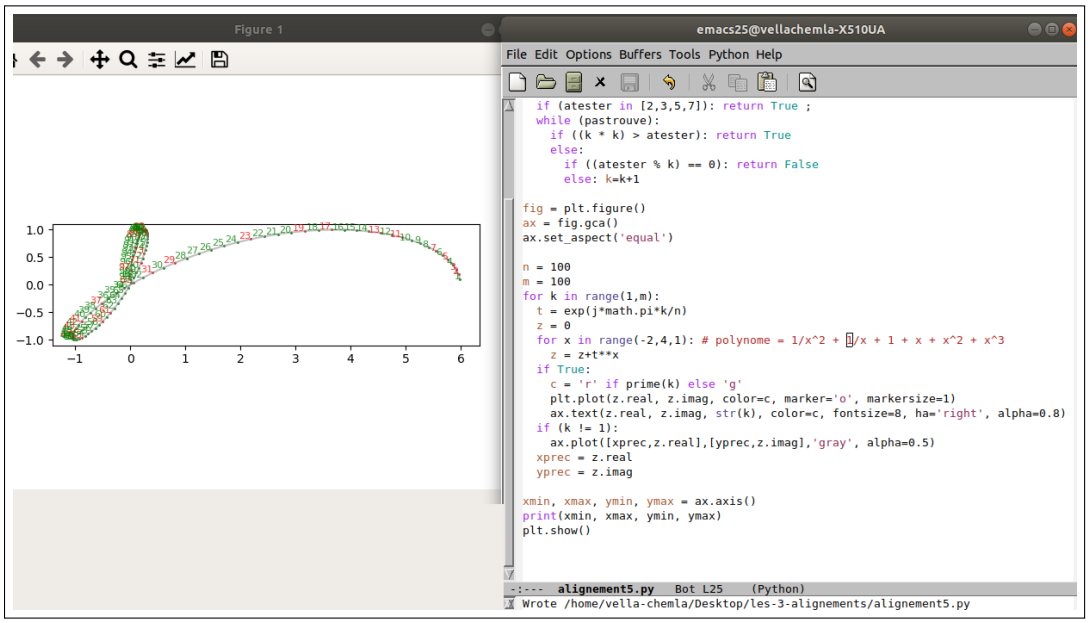


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-2,4,1): # polynome = 1/x^2 + 1/x + 1 + x + x^2 + x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
plt.show()
-:--- alignement5.py Bot L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement5.py

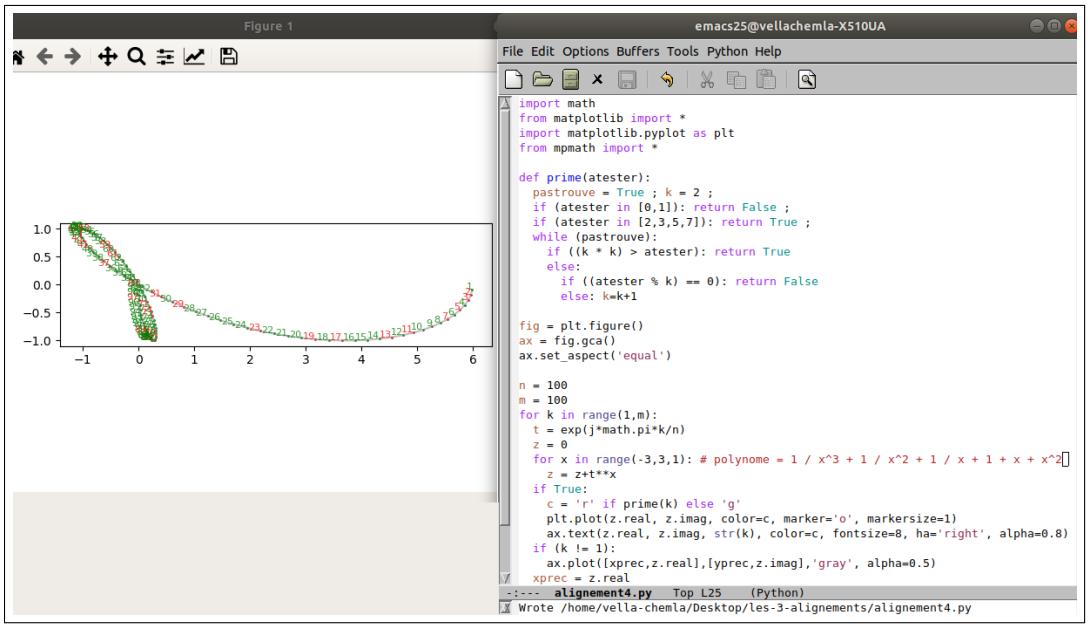


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-3,3,1): # polynome = 1 / x^3 + 1 / x^2 + 1 / x + 1 + x + x^2
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
-:--- alignement4.py Top L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement4.py

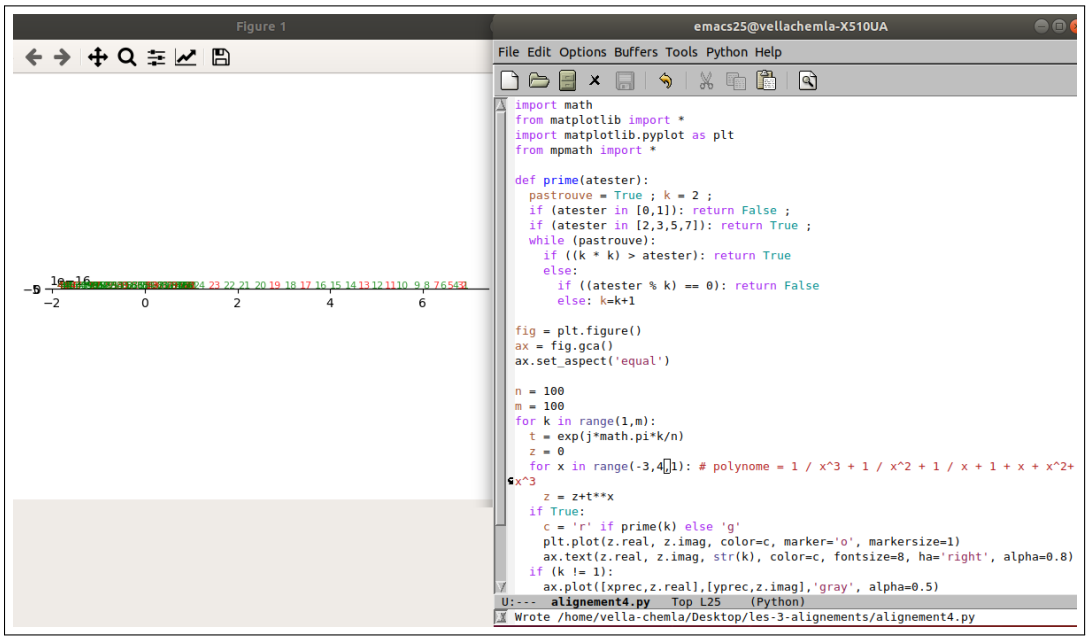
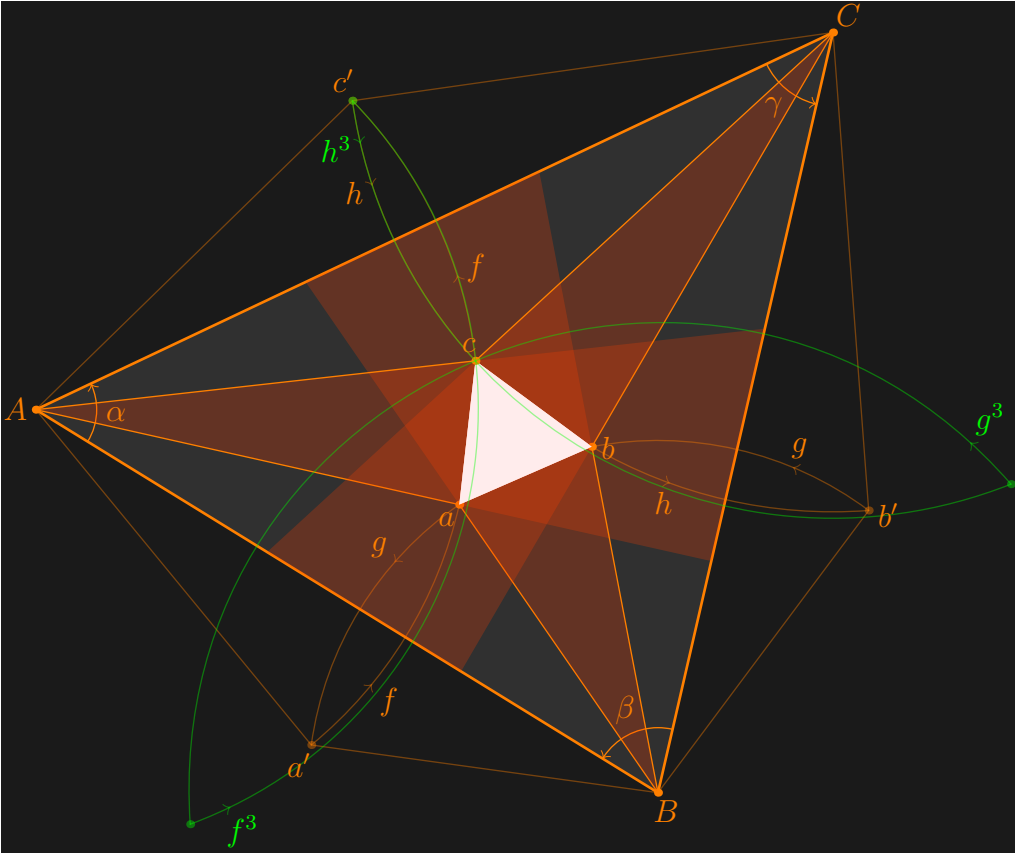


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-3,4,1): # polynome = 1 / x^3 + 1 / x^2 + 1 / x + 1 + x + x^2+x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
U:--- alignement4.py Top L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement4.py

```
import bisect, math, time

primes = [2]

def next_prime(n):
    while any([n % d == 0 for d in primes]): n += 1
    return n

def add_primes(n):
    r = math.sqrt(n)
    while primes[-1] < r: primes.append(next_prime(primes[-1] + 1))

def pi(n):
    # Knuth's algorithm G (Gray binary generation) is used to generate all
subsets of range(m)
    m = bisect.bisect_right(primes, math.sqrt(n))
    a = [0 for _ in range(m)]
    s, μ = m, 1
    while (True):
        d = math.prod([primes[i] for i in range(m) if a[i]])
        s, μ = s + μ * math.floor(n/d), -μ
        j = 0 if μ == -1 else a.index(1) + 1
        if j == m: break
        a[j] = 1 - a[j]
    return s

add_primes(10000)
for k in range(1, 11):
    n = 1000*k
    t0 = time.time(); p = pi(n); t1 = time.time()
    print('n = {:5d}, pi = {:4d}, time = {:10.6f} s'.format(n, p, t1-t0))
```



$$f^3 g^3 h^3 = 1 \iff a + jb + j^2 c = 0$$

```
size(13cm);
unitsize(1cm);

// Colors: T = outer triangle, M = Morley triangle
// Use the "invisible" pen to remove any attribute
pen legend = orange; //blue;
pen sketch = orange + opacity(0.4); //paleblue;
pen sketch2 = green + opacity(0.4);;
pen T_edge = orange + 1bp; //blue + 1bp;
pen M_edge = invisible; //blue + 1bp;
pen T_beam = interp(red, orange, 0.5) + opacity(0.25);
pen M_fill = interp(mediumred, white, 0.9);
pen T_fill = interp(white, deepgray, 0.9);
pen T_back = deepgray;

// Names of triangle vertices, angles and rotations
string[] T_vertex = {"$A$", "$B$", "$C$"};
string[] T_corner = {"$\alpha$", "$\beta$", "$\gamma$"};
string[] M_vertex = {"$a$", "$b$", "$c$"};
string[] S_vertex = {"$a'$", "$b'$", "$c'$"};
string[] rotation = {"$f$", "$g$", "$h$"};
string[] rotation3 = {"$f^3$", "$g^3$", "$h^3$"};

// Outer triangle
path T = (8.83, 47.89)--(72.74, 8.55)--(90.72, 86.63)--cycle;

// Compute inner angles of outer triangle
real[] T_angle;
for (int k=0; k<3; ++k) {
    pair CA = dir(T,k,-1), AB = dir(T,k,+1);
    T_angle[k] = angle(-CA/AB);
}

// Compute Morley triangle
guide m;
for (int k=0; k<3; ++k) {
    int l = (k+1)%3;
    pair A = point(T,k), B = point(T,k+1), AB = dir(T,k,+1), BC = dir(T,k+1,+1);
    // Compute Morley vertices
    pair a = extension(A, A + AB*expi(1/3*T_angle[k]), B, B +
BC*expi(2/3*T_angle[l]));
    m = m--a;
}
path M = m--cycle;

// Draw outer triangle
fill(T, T_fill);
for (int k=0; k<3; ++k) {
    pair A = point(T,k), B = point(T,k+1), AB = dir(T,k,+1), AC = -dir(T,k,-1);
    // Draw vertices
    dot(T_vertex[k], A, -(AB+AC)/2, legend);
    // Draw inner angles
    pair rr = (B-A)/12;
    pair z1 = rr*expi(0), z2 = rr*expi(T_angle[k]);
    draw(arc(A, A+z1, A+z2), arrow=Arrow(TeXHead), L=Label(T_corner[k],
position=MidPoint, legend), legend);
/*
    for (int i=-1; i<4; ++i) {
        pen p = (0 <= i && i < 3) ? legend : sketch;
        pair z1 = rr*expi(i/3*T_angle[k]), z2 = rr*expi((i+1)/3*T_angle[k]);
        draw(arc(A, A+z1, A+z2), arrow=Arrow(TeXHead), L=Label(T_corner[k],
position=MidPoint, p), p);
    }
*/
```

```
}
draw(T, T_edge);

// Draw Morley triangle
for (int k=0; k<3; ++k) {
    pair A = point(T,k), B = point(T,k+1), C = point(T,k+2), a = point(M,k), c =
point(M,k+2);
    // Compute symmetrical Morley vertices
    int l = (k+1)%3;
    pair s = B + (a - B)*expi(2/3*T_angle[l]);
    // Fill light beams
    fill(A--extension(A, a, B, C)--extension(A, c, B, C)--cycle, T_beam);
    // Fill Morley triangle
    fill(M, M_fill);
    // Draw trisectors
    draw(A--a--B, legend);
    draw(A--s--B, sketch);
    // Draw rotations
    position pos = Relative(0.57);
    pair a_out = (a - B)*expi(pi/2), s_out = (s - A)*expi(pi/2);
    draw(a{a_out}..s, arrow=Arrow(TeXHead, pos), L=Label(rotation[l],
position=pos, sketch), sketch);
    draw(s{s_out}..a, arrow=Arrow(TeXHead, pos), L=Label(rotation[k],
position=pos, sketch), sketch);
    // Draw vertices
    dot(M_vertex[k], a, dir(C--a), legend);
    dot(S_vertex[k], s, dir(a--s), sketch);
}
draw(M, M_edge);

// Draw f^3 g^3 h^3 = 1
pair p = point(T,2) + (point(M,2) - point(T,2))*expi(-2/3*T_angle[2]);
for (int k=2; k>=0; --k) {
    pair C = point(T,k);
    position pos = Relative(0.05);
    pair p_out = (p - C)*expi(pi/2), q = C + (p - C)*expi(2*T_angle[k]);
    draw(p{p_out}..q, arrow=Arrow(TeXHead, pos), L=Label(rotation3[k],
position=pos, sketch2), sketch2);
    dot(p, sketch2);
    p = q;
}

pair ll = min(currentpicture,true), ur = max(currentpicture,true);
label("$f^3g^3h^3=1\iff a+jb+j^2c=0$", ll+(ur.x-ll.x)/2+S*0.15cm, sketch);

// Fill background
shipout(bbox(T_back, Fill));
```

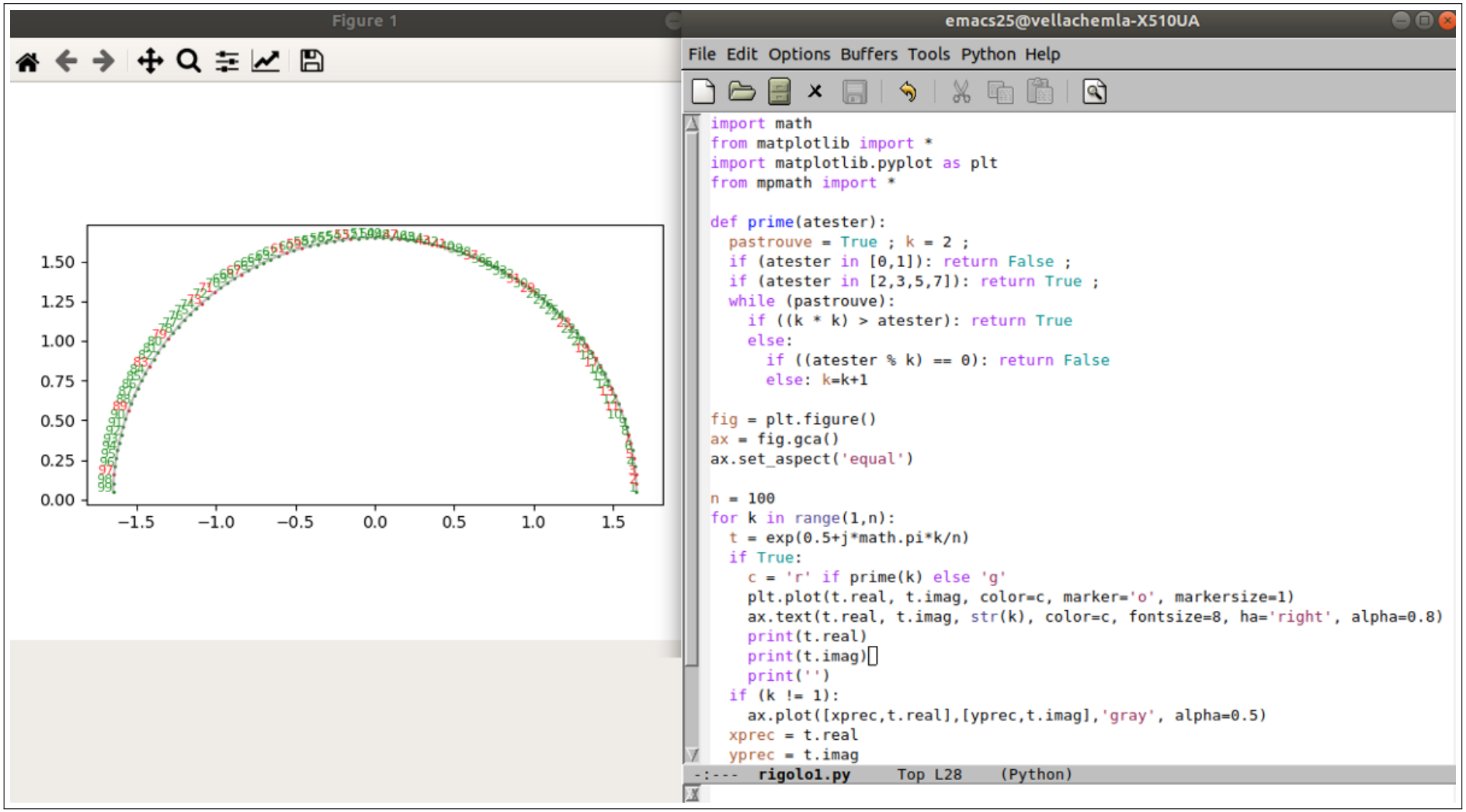


Figure 1
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
for k in range(1,n):
t = exp(0.5+j*math.pi*k/n)
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L28 (Python)

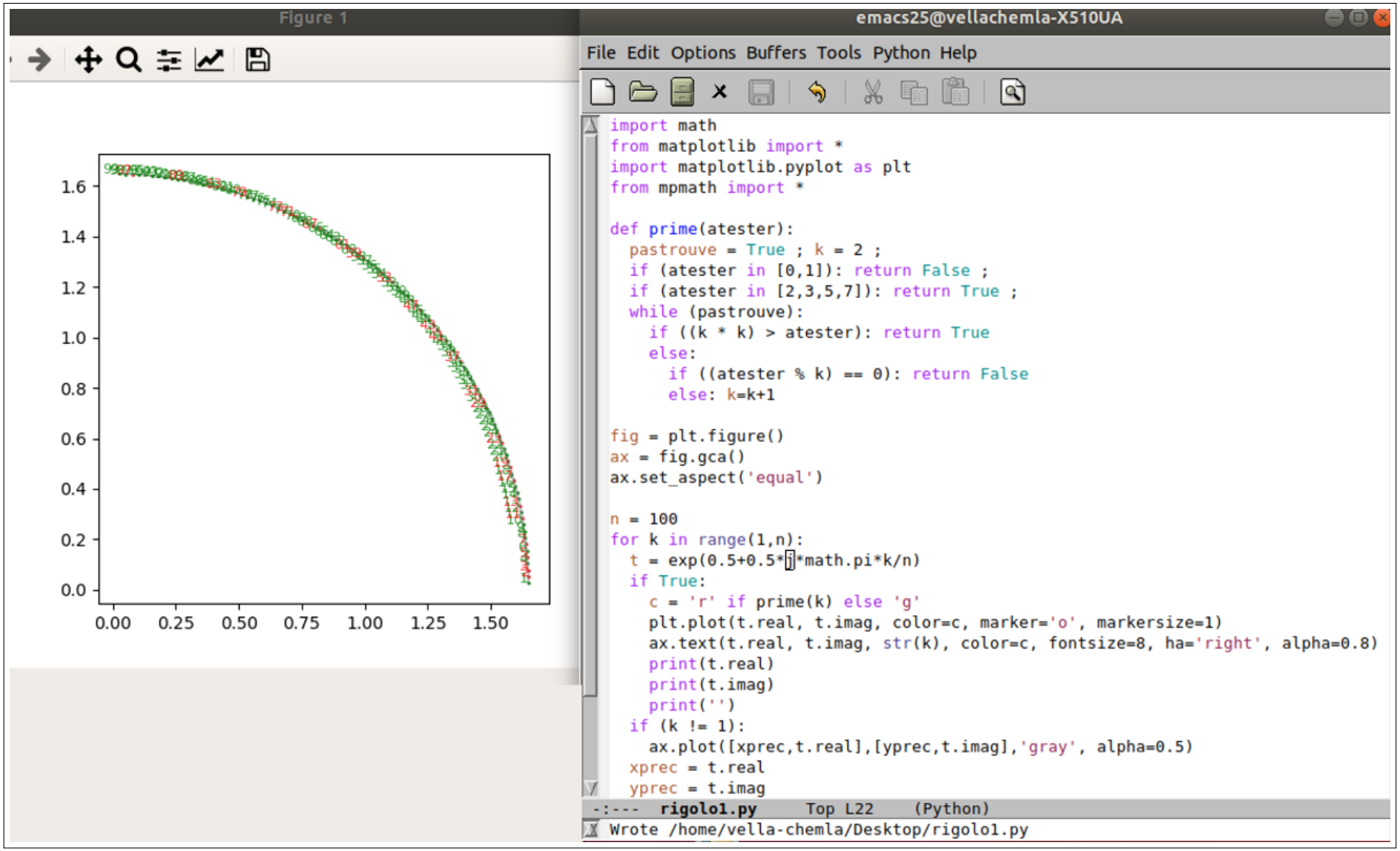


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
for k in range(1,n):
t = exp(0.5+0.5*j*math.pi*k/n)
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L22 (Python)
Wrote /home/vella-chemla/Desktop/rigolo1.py
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
0.00
0.25
0.50
0.75
1.00
1.25
1.50

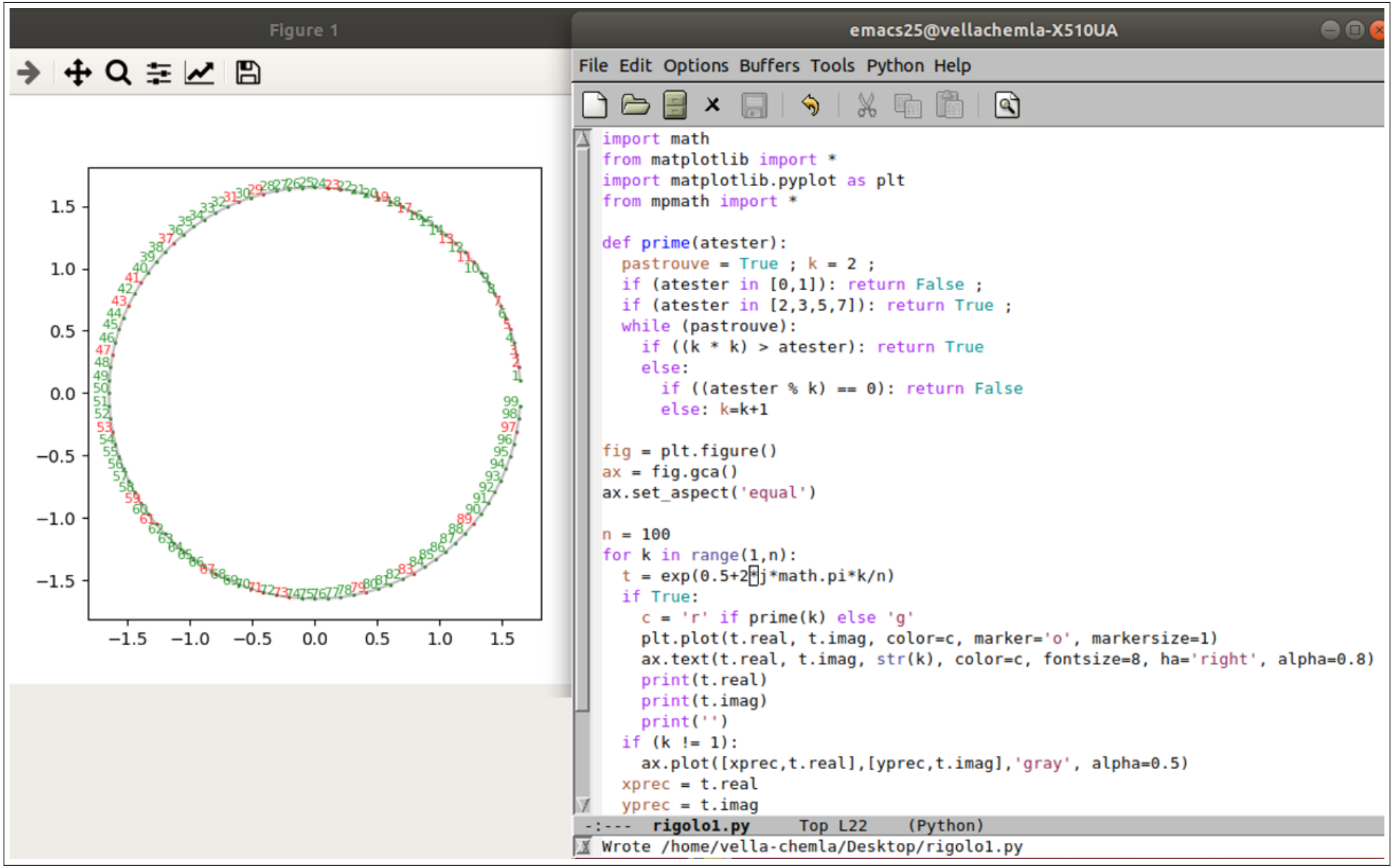


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
for k in range(1,n):
t = exp(0.5+2*j*math.pi*k/n)
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L22 (Python)
Wrote /home/vella-chemla/Desktop/rigolo1.py

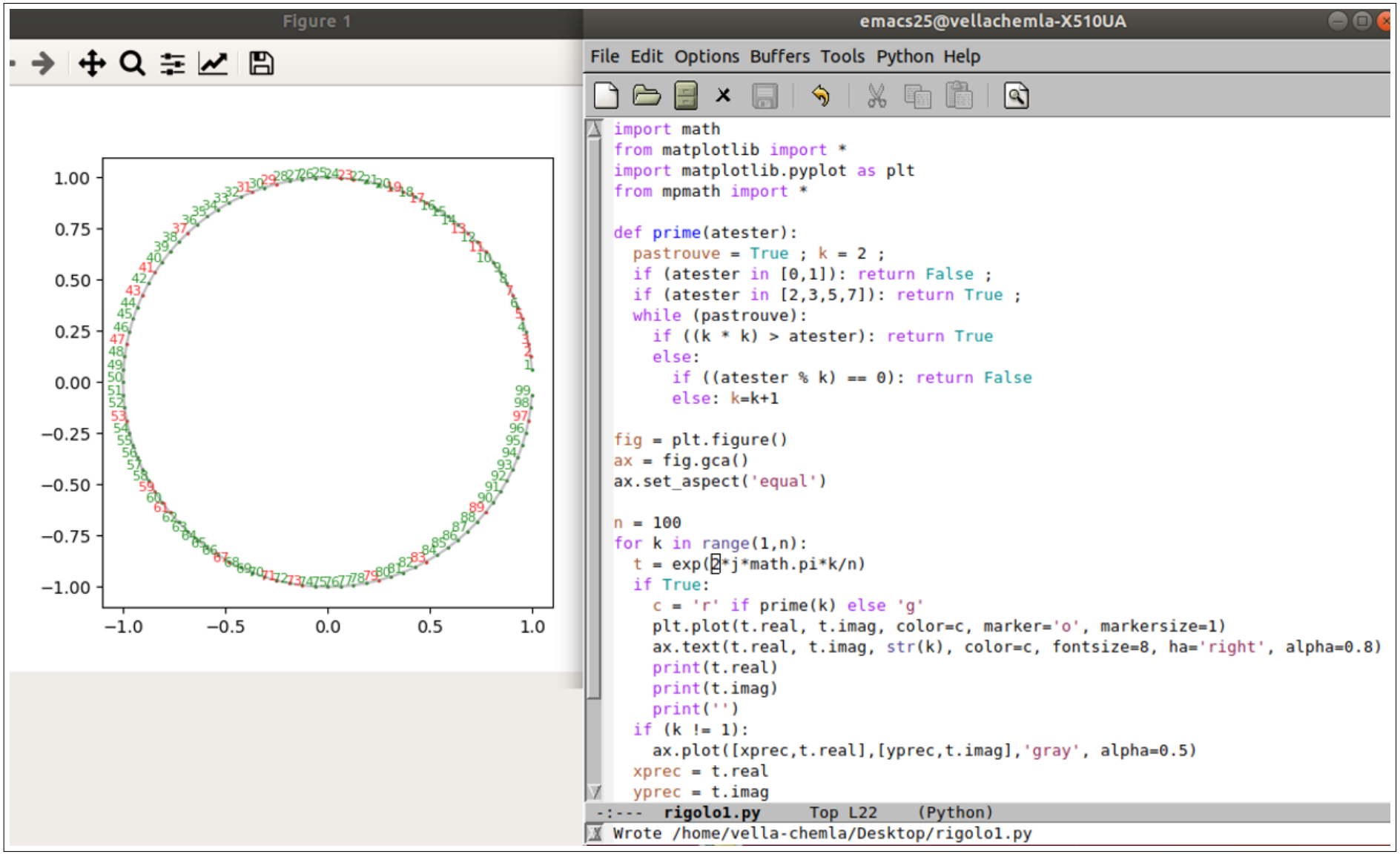


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
for k in range(1,n):
t = exp(2*j*math.pi*k/n)
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L22 (Python)
Wrote /home/vella-chemla/Desktop/rigolo1.py
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.0
−0.5
0.0
0.5
1.0

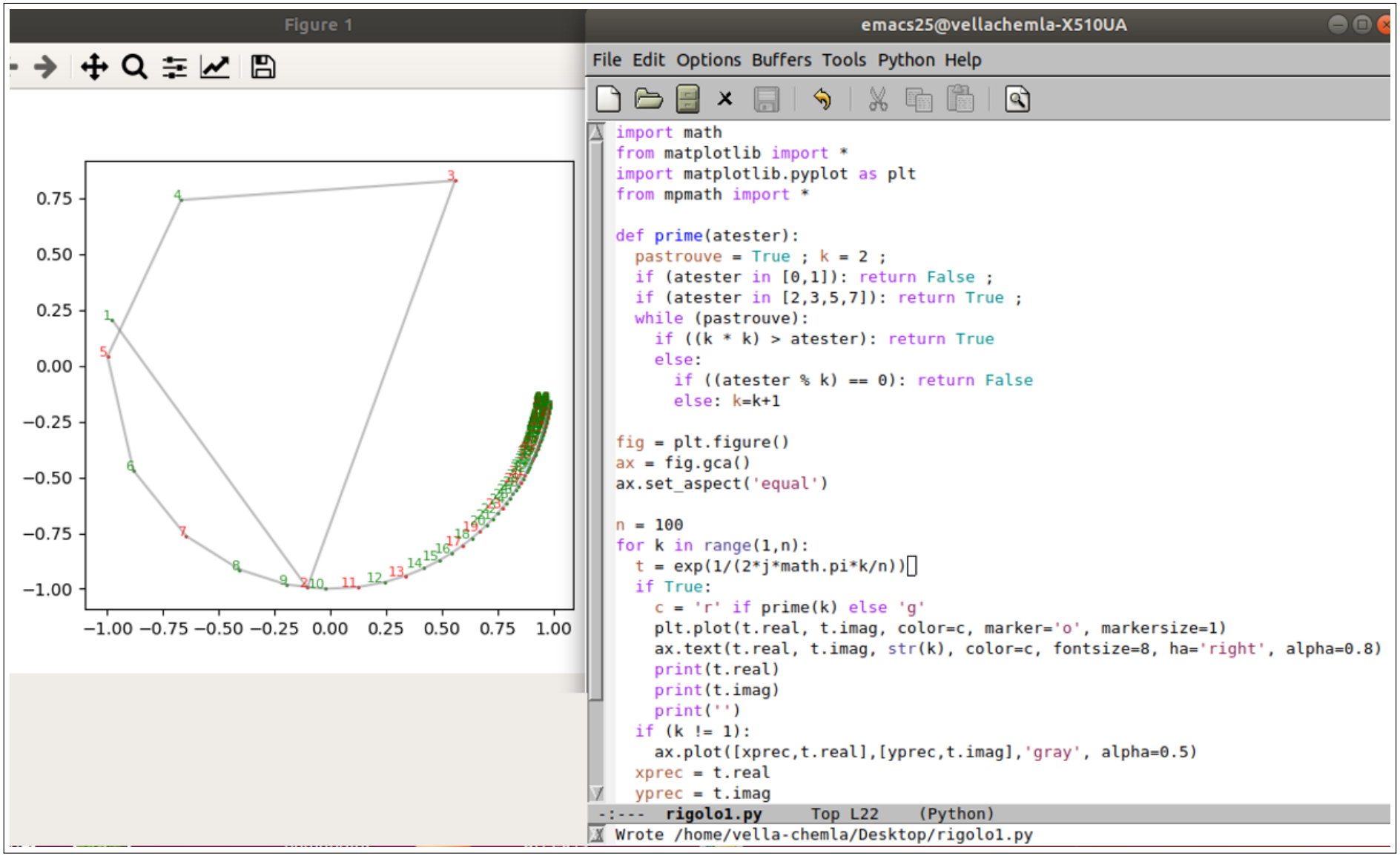


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
for k in range(1,n):
t = exp(1/(2*j*math.pi*k/n))
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L22 (Python)
Wrote /home/vella-chemla/Desktop/rigolo1.py
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.00 −0.75 −0.50 −0.25 0.00 0.25 0.50 0.75 1.00

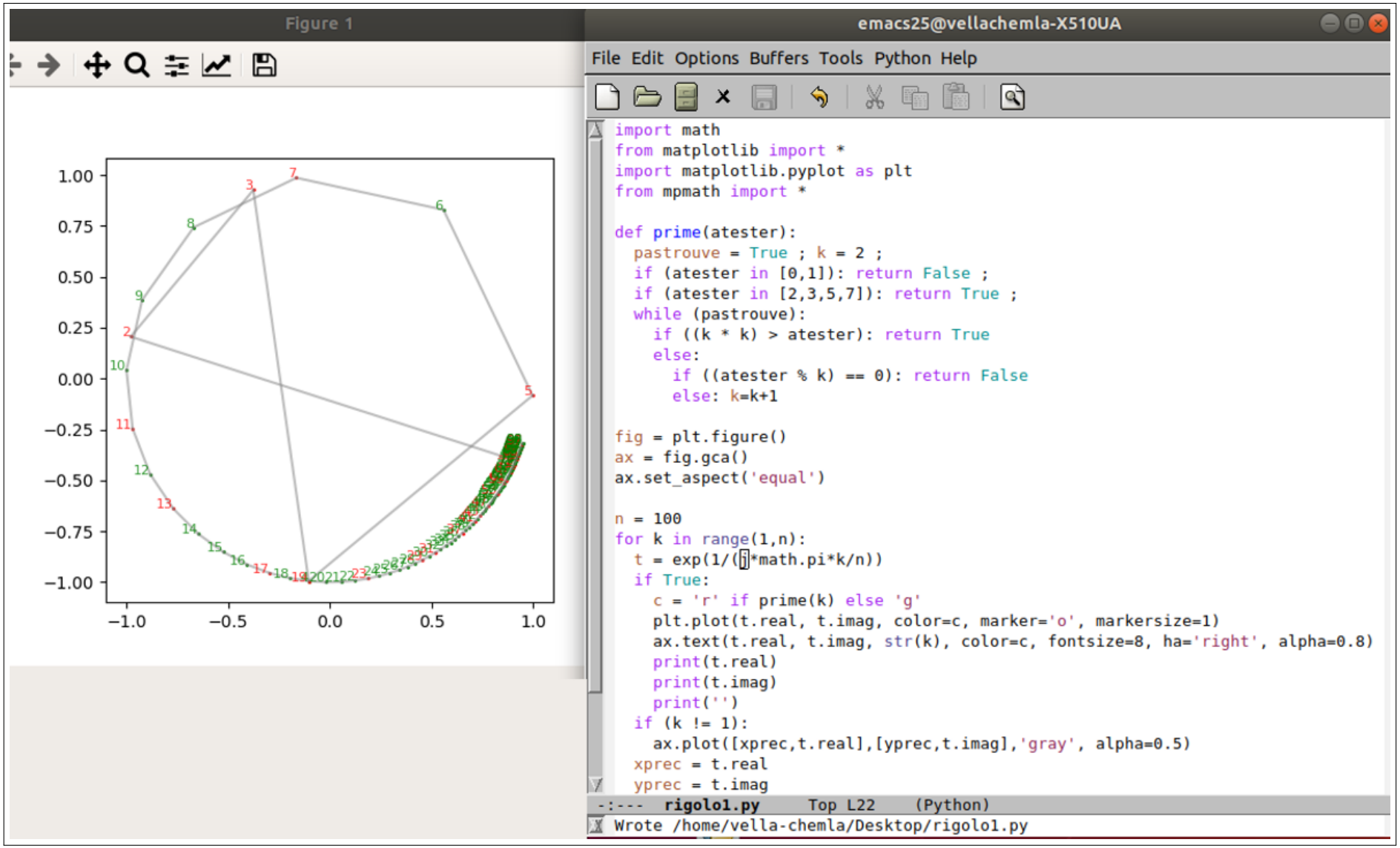


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
for k in range(1,n):
t = exp(1/(j*math.pi*k/n))
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L22 (Python)
Wrote /home/vella-chemla/Desktop/rigolo1.py

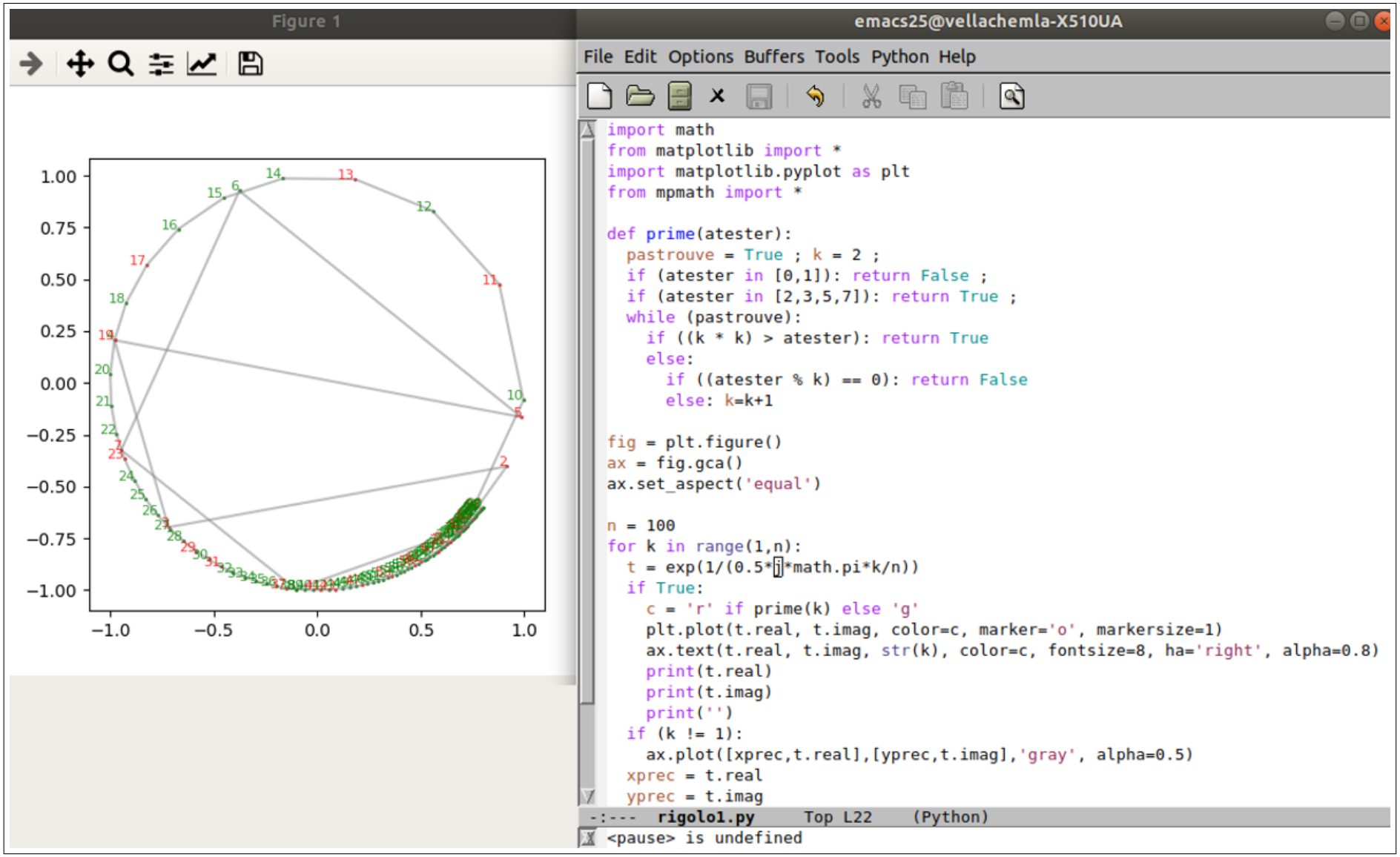


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
for k in range(1,n):
t = exp(1/(0.5*j*math.pi*k/n))
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L22 (Python)
<pause> is undefined
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.0
−0.5
0.0
0.5
1.0

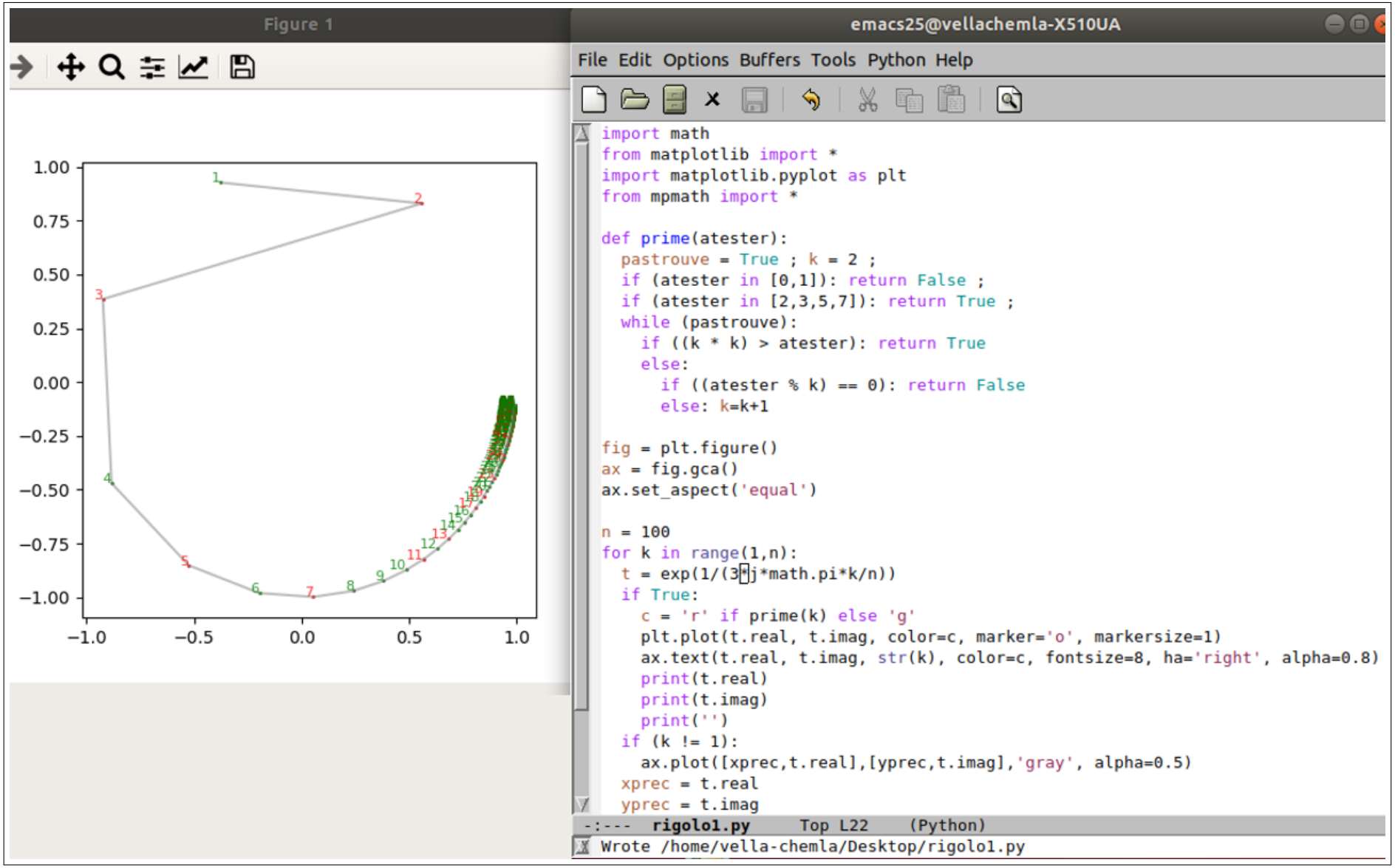


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
for k in range(1,n):
t = exp(1/(3*j*math.pi*k/n))
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L22 (Python)
Wrote /home/vella-chemla/Desktop/rigolo1.py
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.0
−0.5
0.0
0.5
1.0

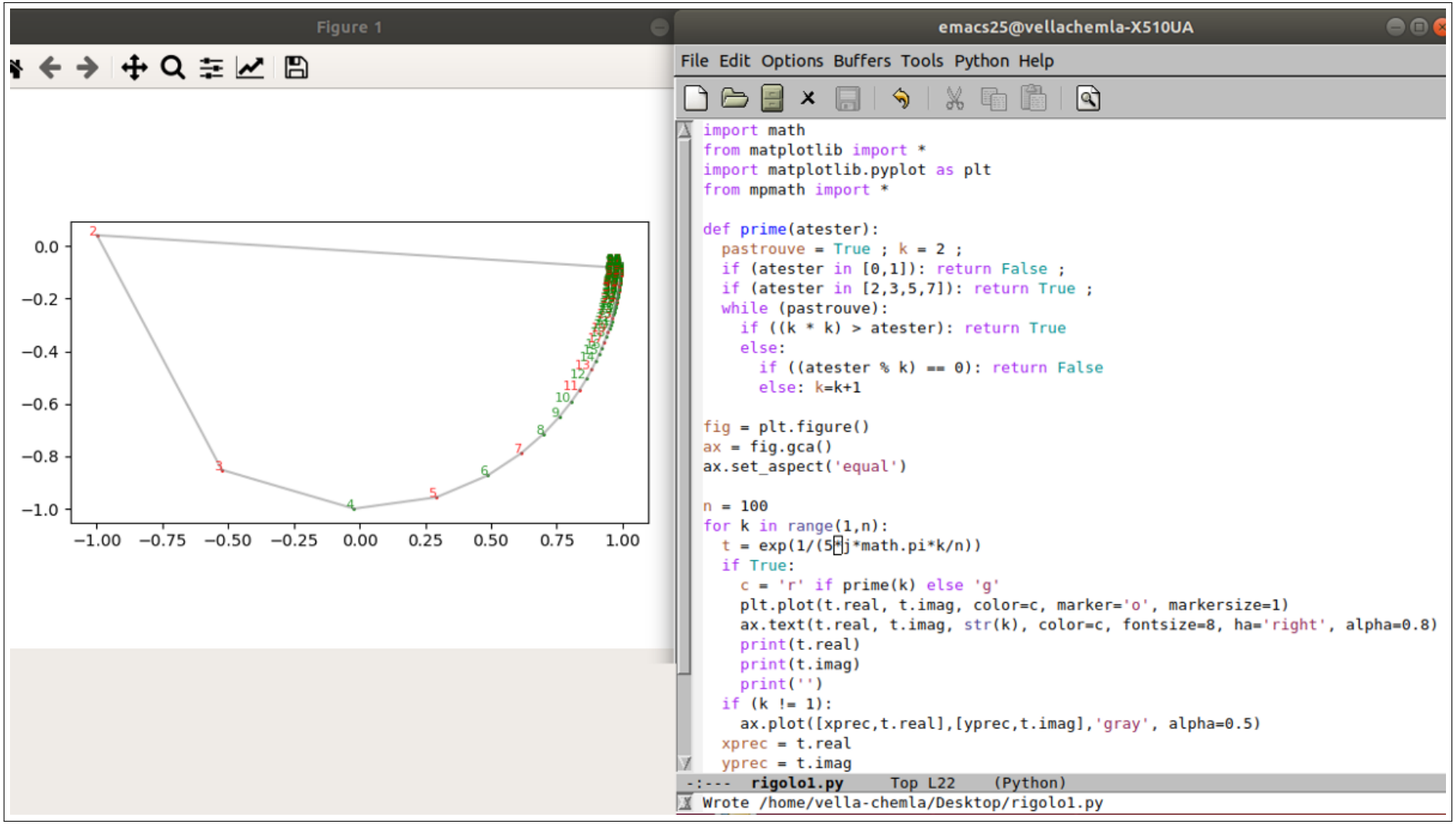


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
for k in range(1,n):
t = exp(1/(5*j*math.pi*k/n))
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L22 (Python)
Wrote /home/vella-chemla/Desktop/rigolo1.py
0.0
−0.2
−0.4
−0.6
−0.8
−1.0
−1.00 −0.75 −0.50 −0.25 0.00 0.25 0.50 0.75 1.00

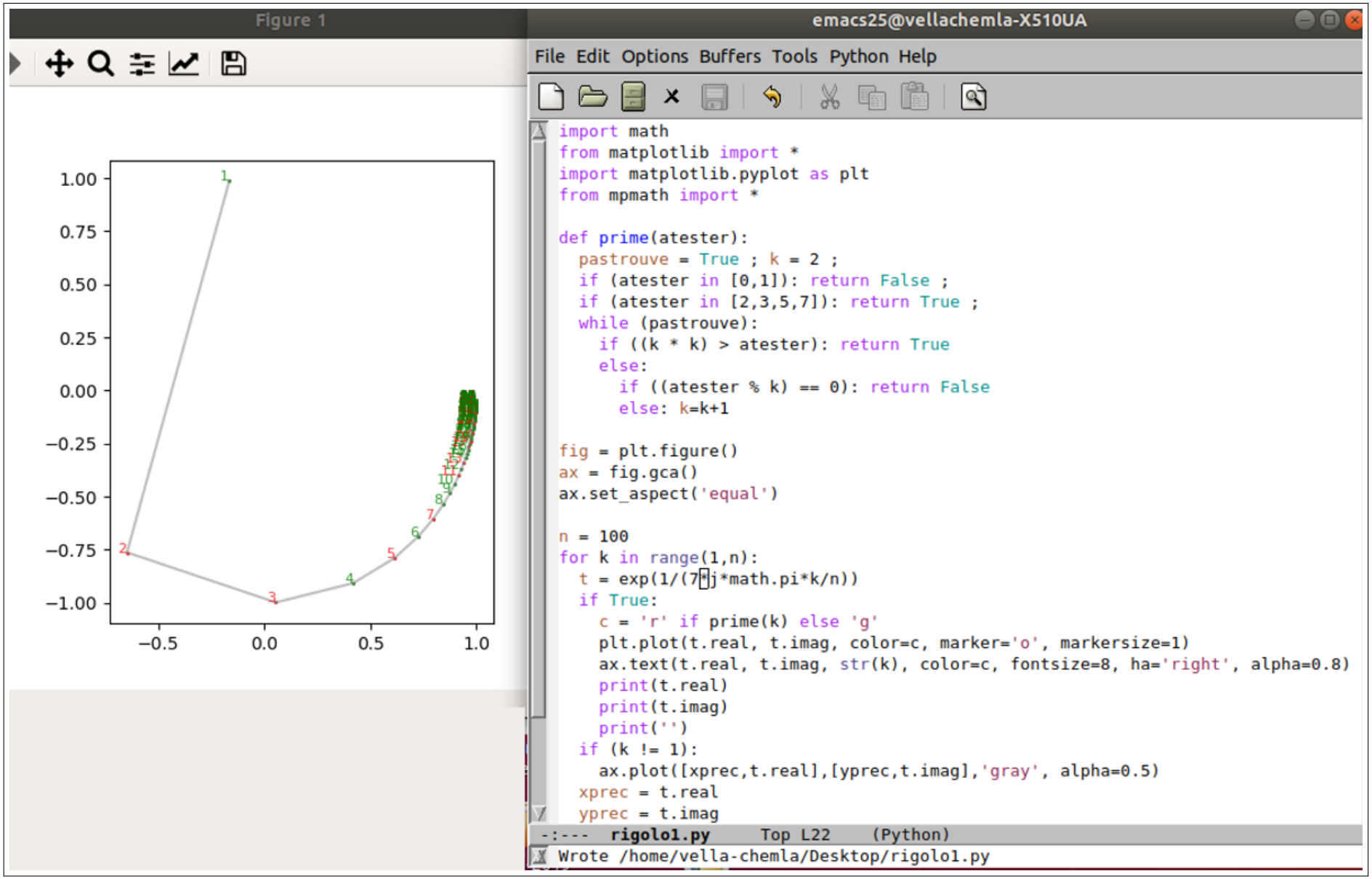


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
for k in range(1,n):
t = exp(1/(7*j*math.pi*k/n))
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L22 (Python)
Wrote /home/vella-chemla/Desktop/rigolo1.py
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−0.5
0.0
0.5
1.0

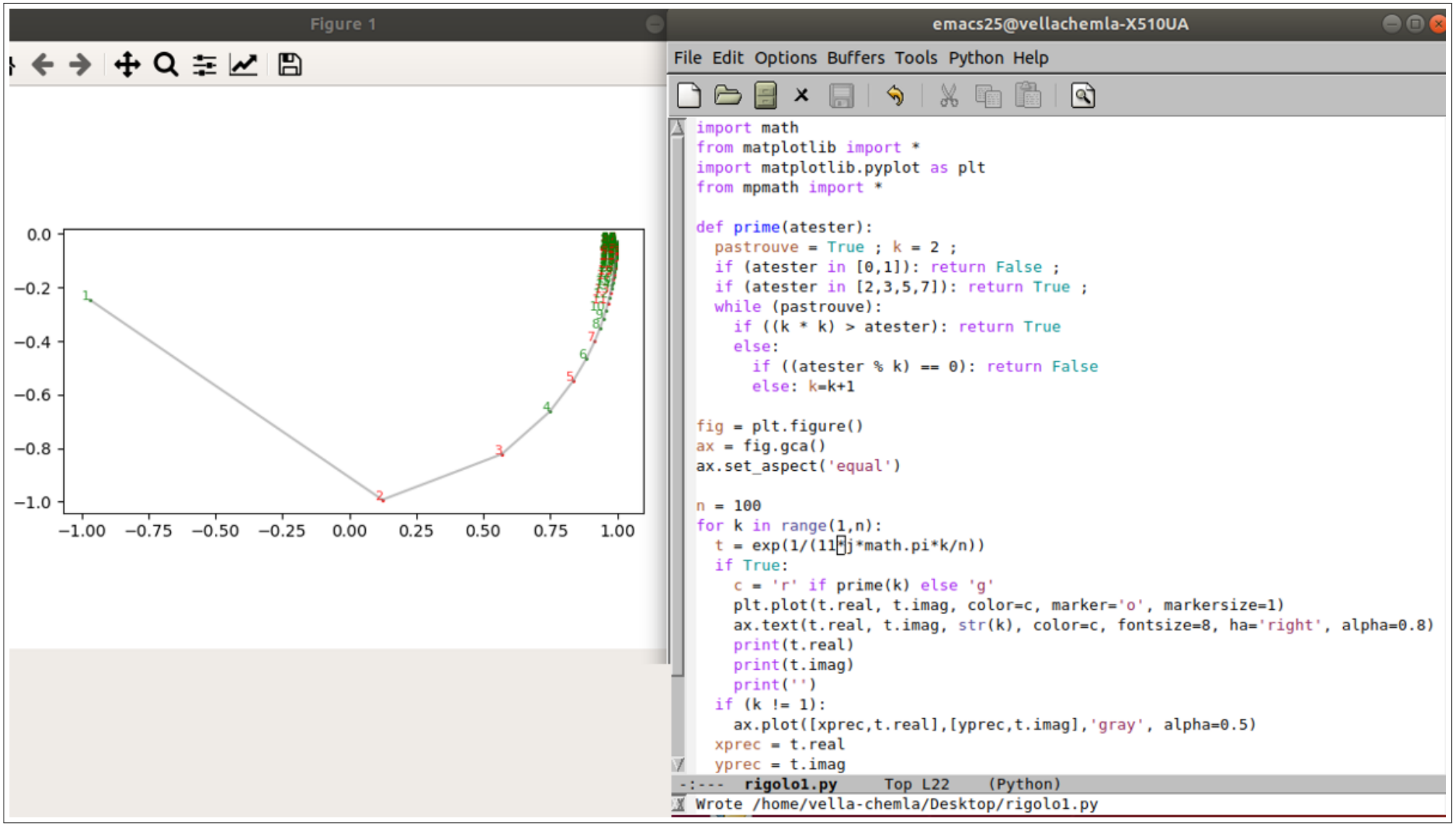


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
for k in range(1,n):
t = exp(1/(11*j*math.pi*k/n))
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L22 (Python)
Wrote /home/vella-chemla/Desktop/rigolo1.py
0.0
−0.2
−0.4
−0.6
−0.8
−1.0
−1.00 −0.75 −0.50 −0.25 0.00 0.25 0.50 0.75 1.00

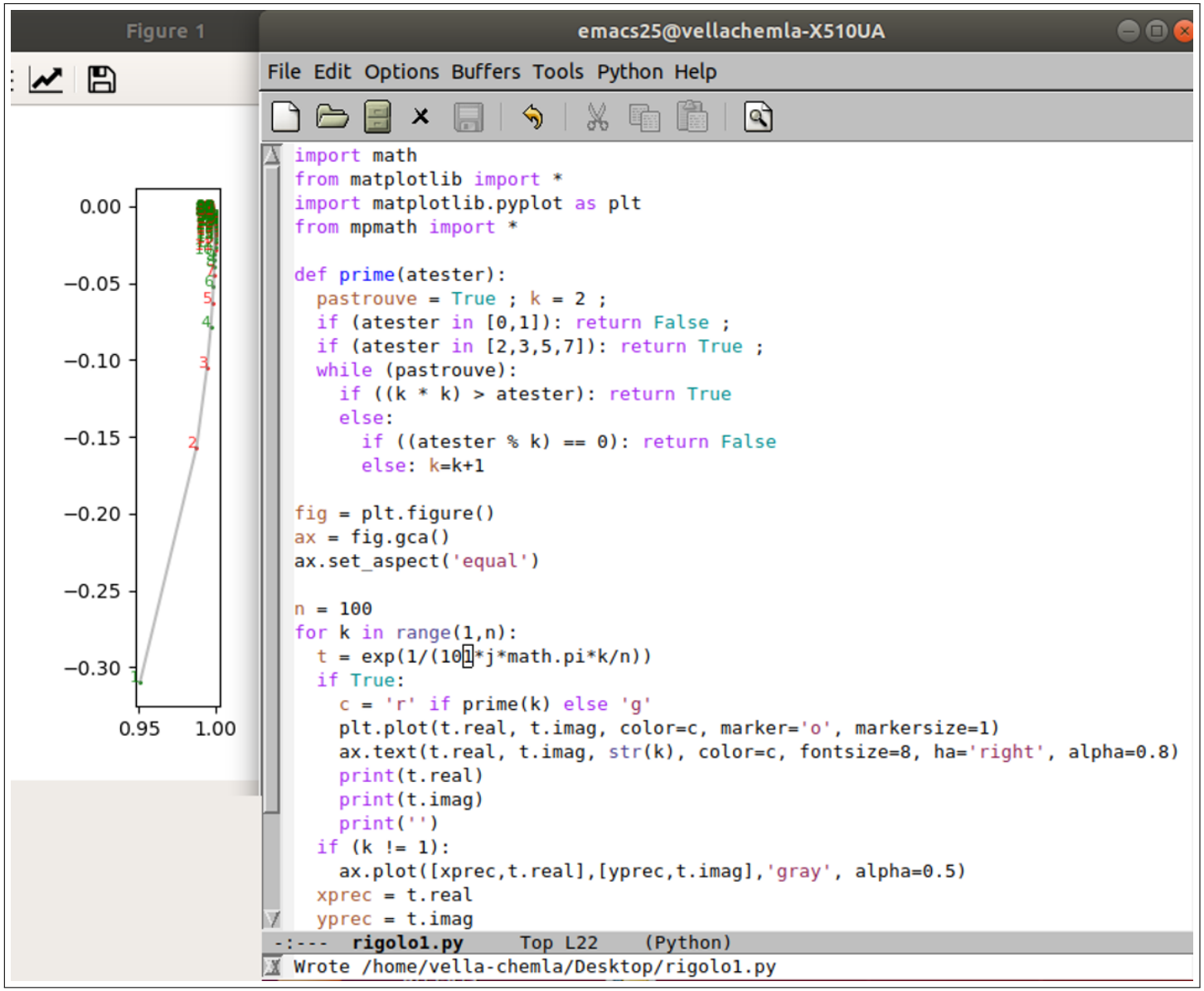


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
0.00
−0.05
−0.10
−0.15
−0.20
−0.25
−0.30
0.95
1.00
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
for k in range(1,n):
t = exp(1/(101*j*math.pi*k/n))
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L22 (Python)
Wrote /home/vella-chemla/Desktop/rigolo1.py

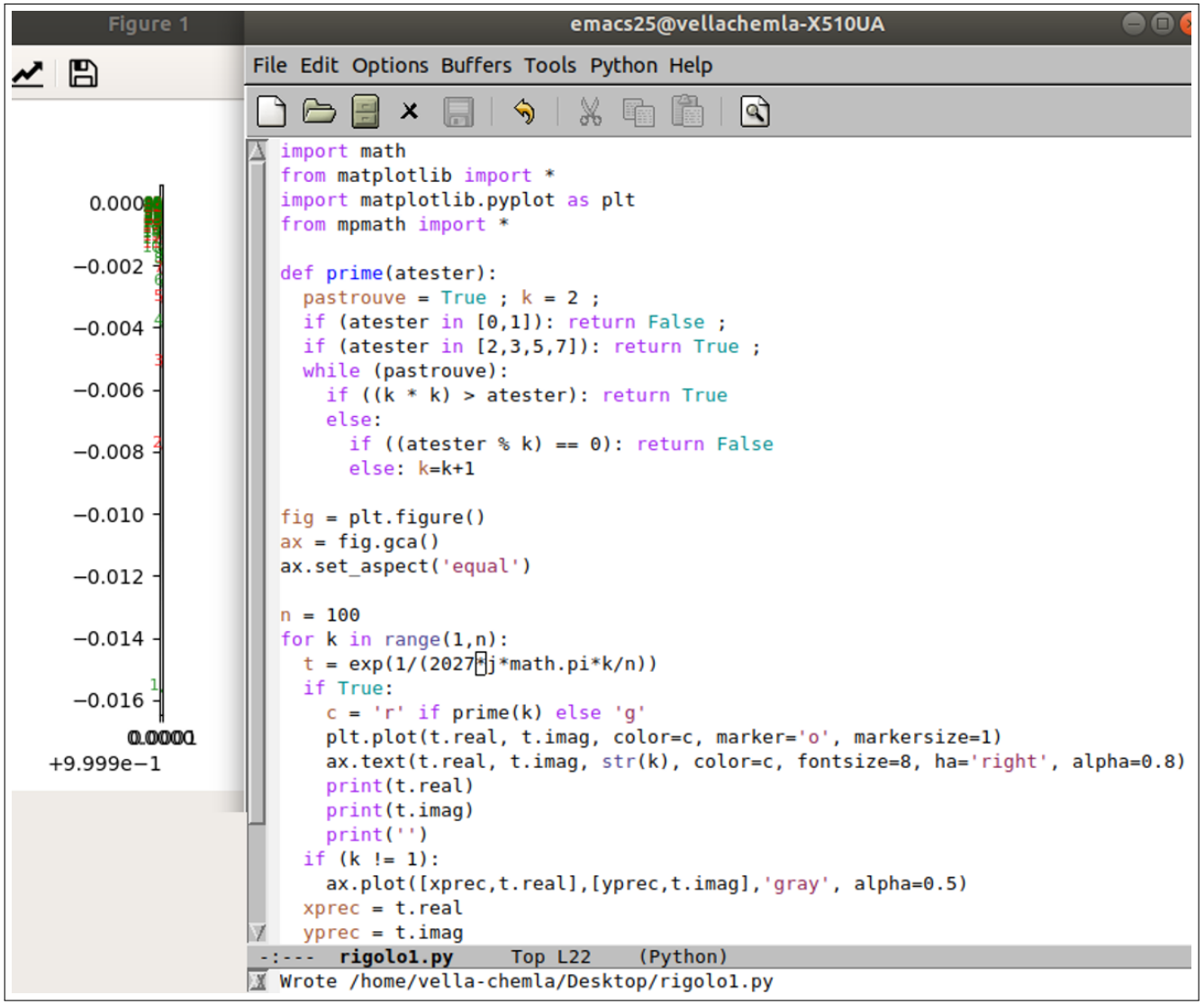


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
0.000
−0.002
−0.004
−0.006
−0.008
−0.010
−0.012
−0.014
−0.016
0.00001
+9.999e−1
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
for k in range(1,n):
t = exp(1/(2027*j*math.pi*k/n))
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L22 (Python)
Wrote /home/vella-chemla/Desktop/rigolo1.py

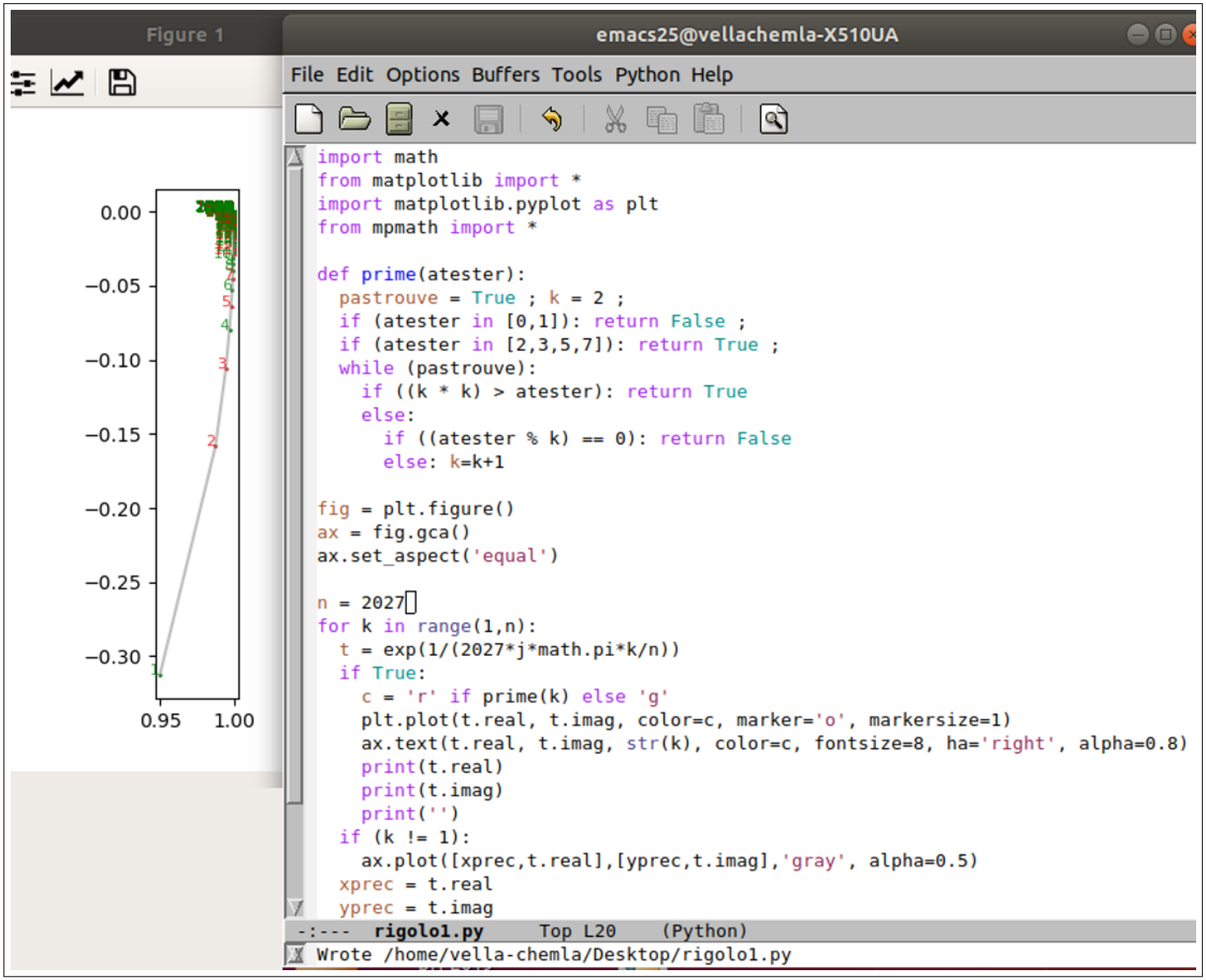


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
0.00
−0.05
−0.10
−0.15
−0.20
−0.25
−0.30
0.95
1.00
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 2027
for k in range(1,n):
t = exp(1/(2027*j*math.pi*k/n))
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L20 (Python)
Wrote /home/vella-chemla/Desktop/rigolo1.py

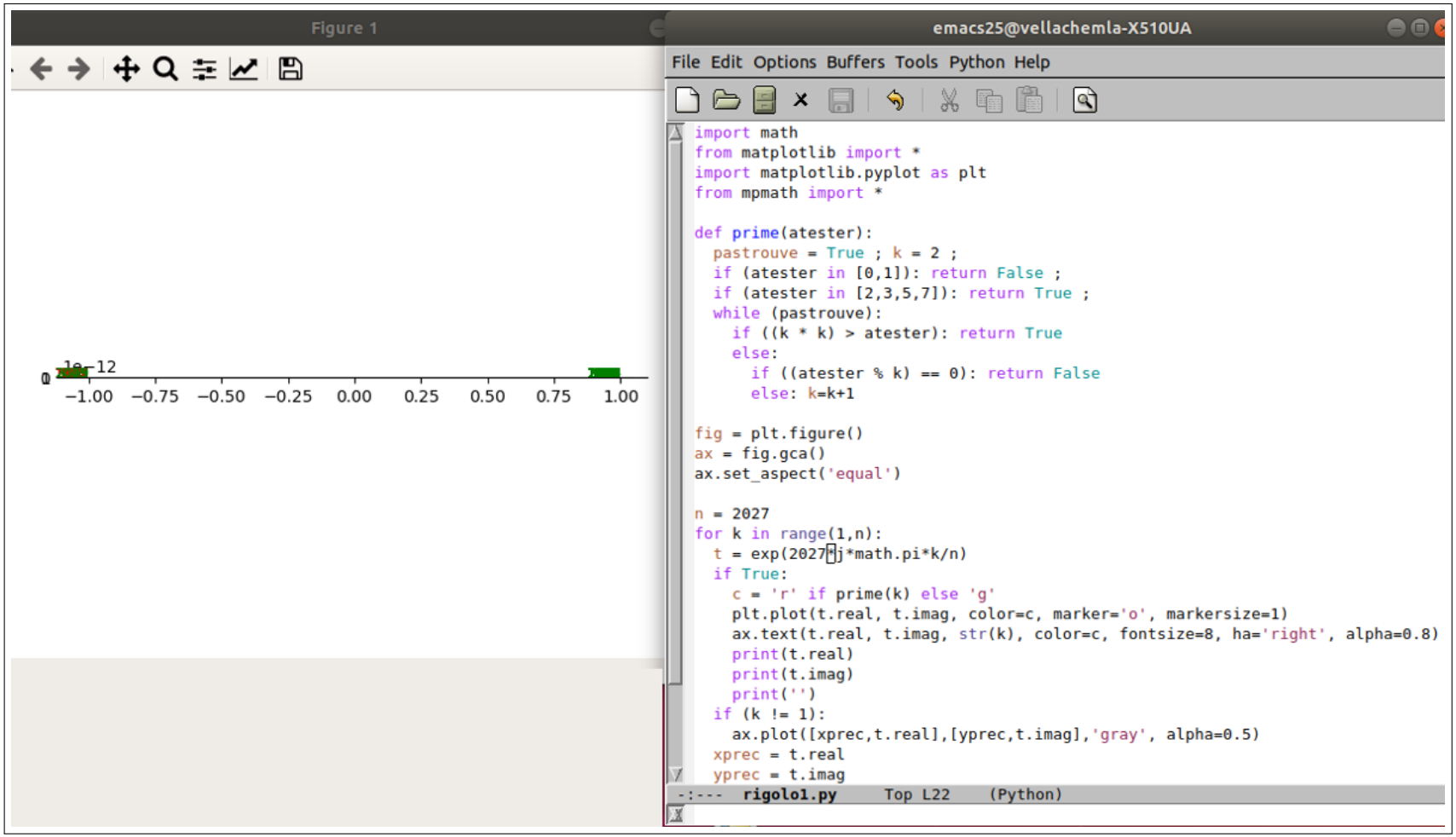


Figure 1
1e−12
−1.00 −0.75 −0.50 −0.25 0.00 0.25 0.50 0.75 1.00
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 2027
for k in range(1,n):
t = exp(2027*j*math.pi*k/n)
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L22 (Python)

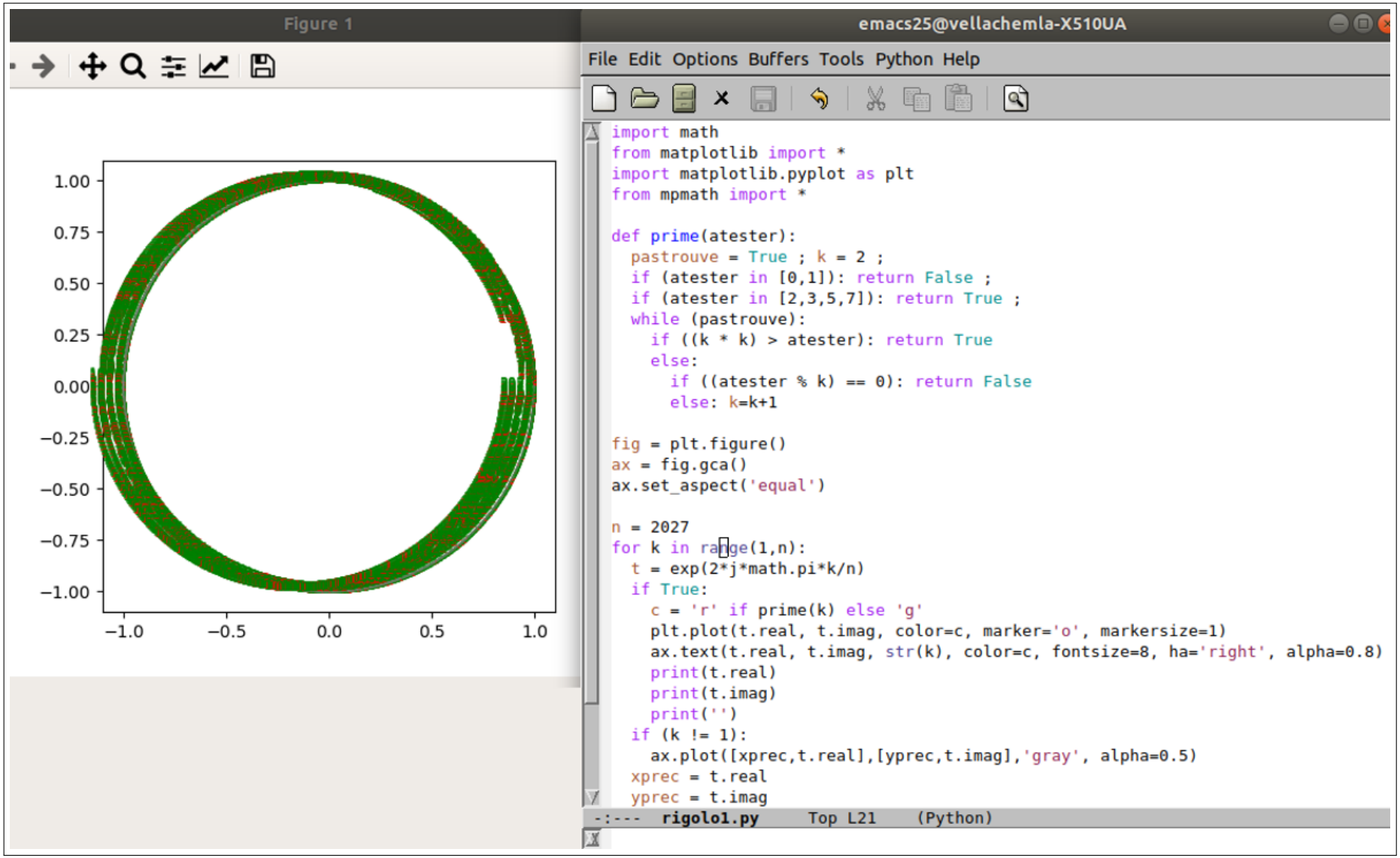
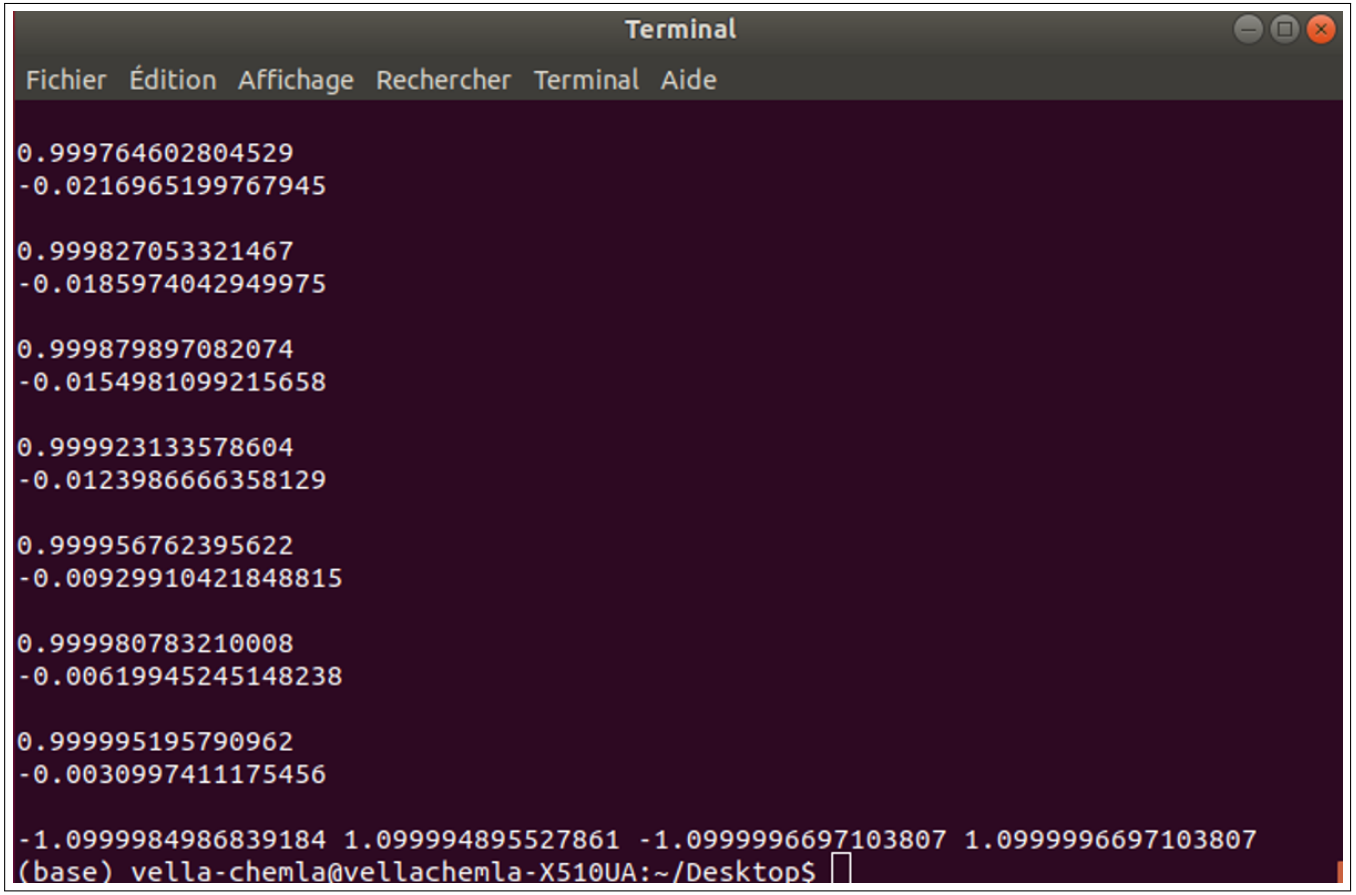


Figure 1
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.0
−0.5
0.0
0.5
1.0
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 2027
for k in range(1,n):
t = exp(2*j*math.pi*k/n)
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
-:--- rigolo1.py Top L21 (Python)


Terminal
Fichier Édition Affichage Rechercher Terminal Aide
0.999764602804529
-0.0216965199767945
0.999827053321467
-0.0185974042949975
0.999879897082074
-0.0154981099215658
0.999923133578604
-0.0123986666358129
0.999956762395622
-0.00929910421848815
0.999980783210008
-0.00619945245148238
0.999995195790962
-0.0030997411175456
-1.0999984986839184 1.099994895527861 -1.0999996697103807 1.0999996697103807
(base) vella-chemla@vellachemla-X510UA:~/Desktop$

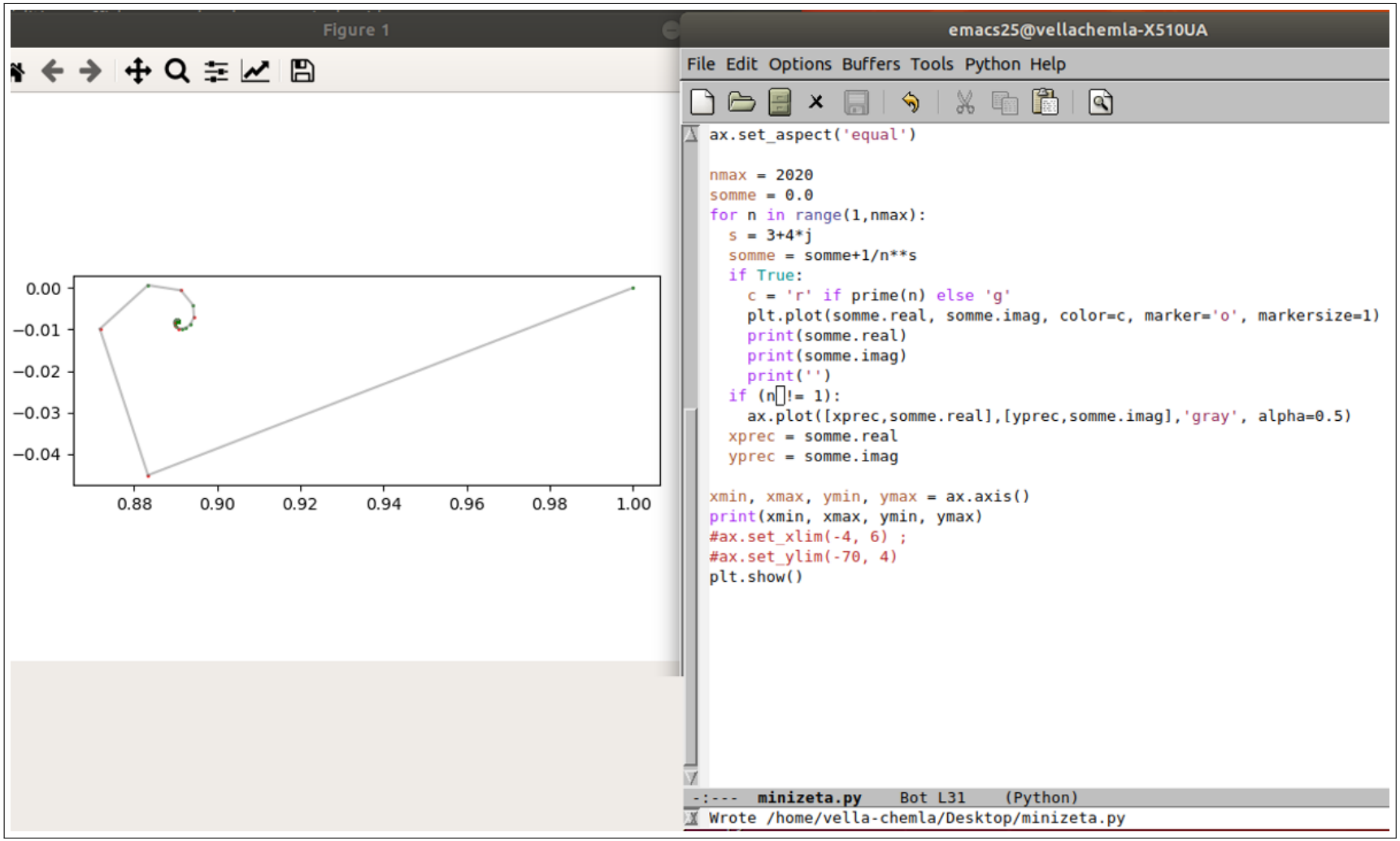


Figure 1
0.00
−0.01
−0.02
−0.03
−0.04
0.88
0.90
0.92
0.94
0.96
0.98
1.00
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
ax.set_aspect('equal')
nmax = 2020
somme = 0.0
for n in range(1,nmax):
s = 3+4*j
somme = somme+1/n**s
if True:
c = 'r' if prime(n) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
print(somme.real)
print(somme.imag)
print('')
if (n != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-4, 6) ;
#ax.set_ylim(-70, 4)
plt.show()
-:--- minizeta.py Bot L31 (Python)
Wrote /home/vella-chemla/Desktop/minizeta.py

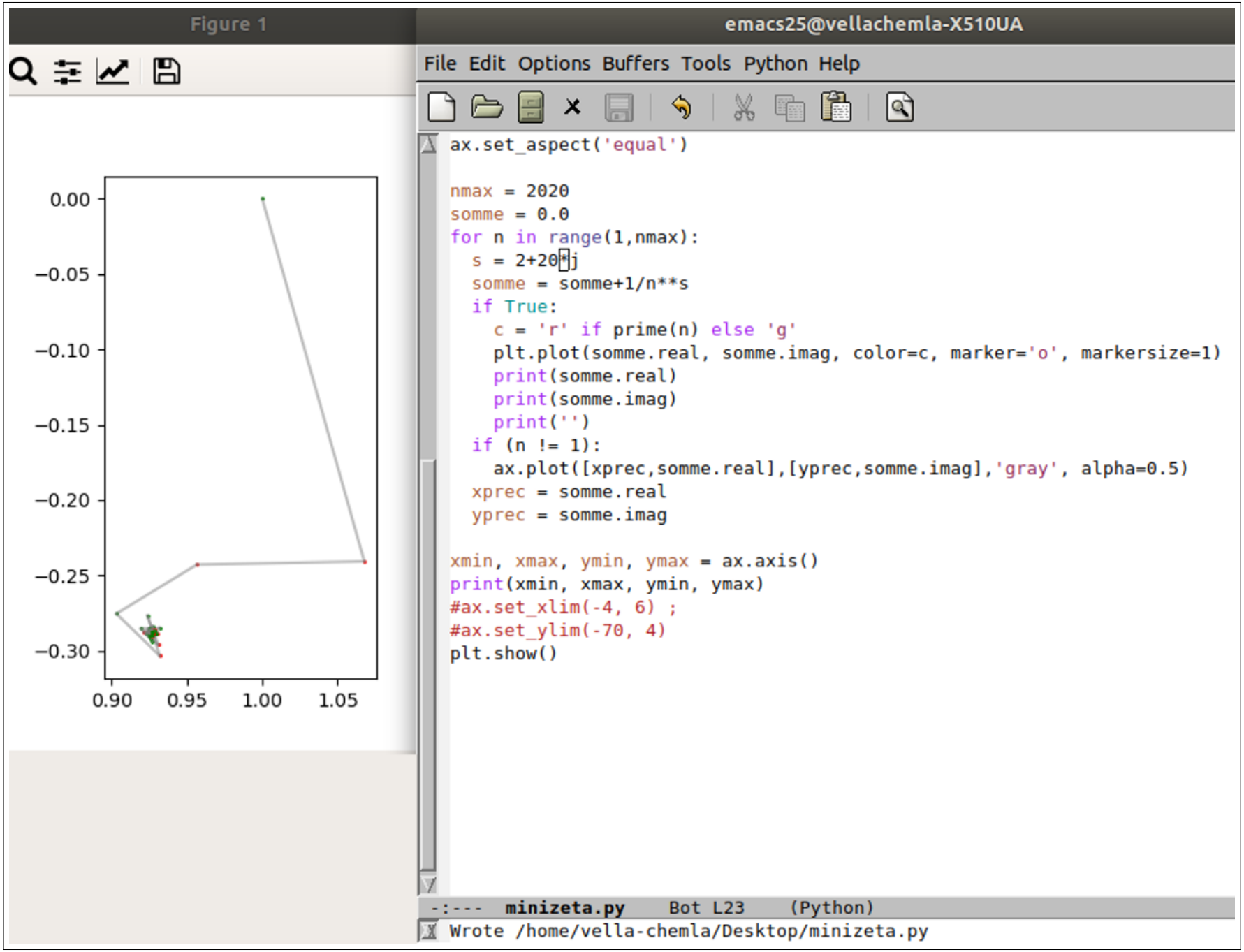


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
ax.set_aspect('equal')
nmax = 2020
somme = 0.0
for n in range(1,nmax):
s = 2+20*j
somme = somme+1/n**s
if True:
c = 'r' if prime(n) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
print(somme.real)
print(somme.imag)
print('')
if (n != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-4, 6) ;
#ax.set_ylim(-70, 4)
plt.show()
0.00
−0.05
−0.10
−0.15
−0.20
−0.25
−0.30
0.90
0.95
1.00
1.05
-:--- minizeta.py Bot L23 (Python)
Wrote /home/vella-chemla/Desktop/minizeta.py

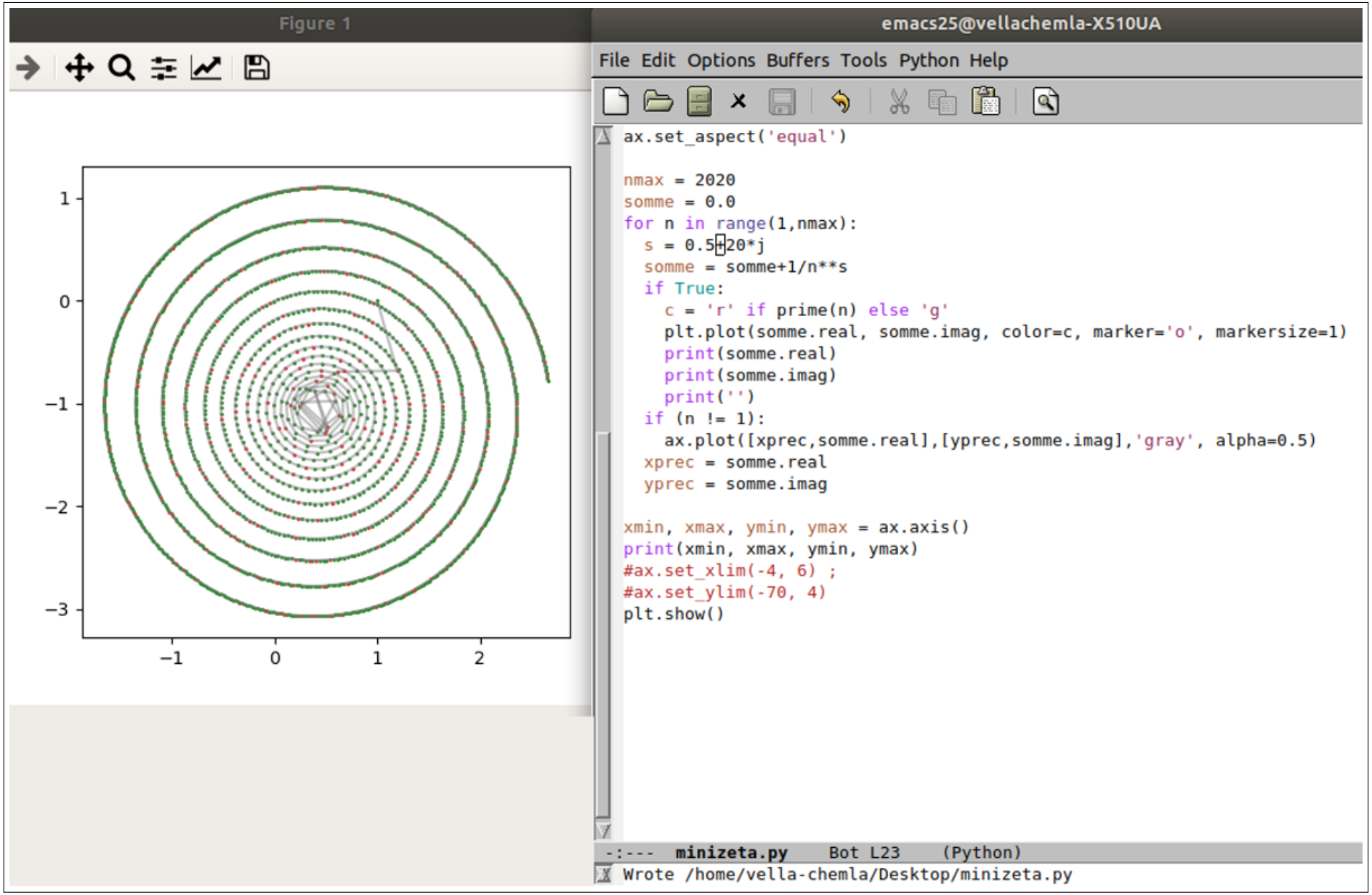


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
ax.set_aspect('equal')
nmax = 2020
somme = 0.0
for n in range(1,nmax):
s = 0.5+20*j
somme = somme+1/n**s
if True:
c = 'r' if prime(n) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
print(somme.real)
print(somme.imag)
print('')
if (n != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-4, 6) ;
#ax.set_ylim(-70, 4)
plt.show()
-:--- minizeta.py Bot L23 (Python)
Wrote /home/vella-chemla/Desktop/minizeta.py
1
0
−1
−2
−3
−1
0
1
2

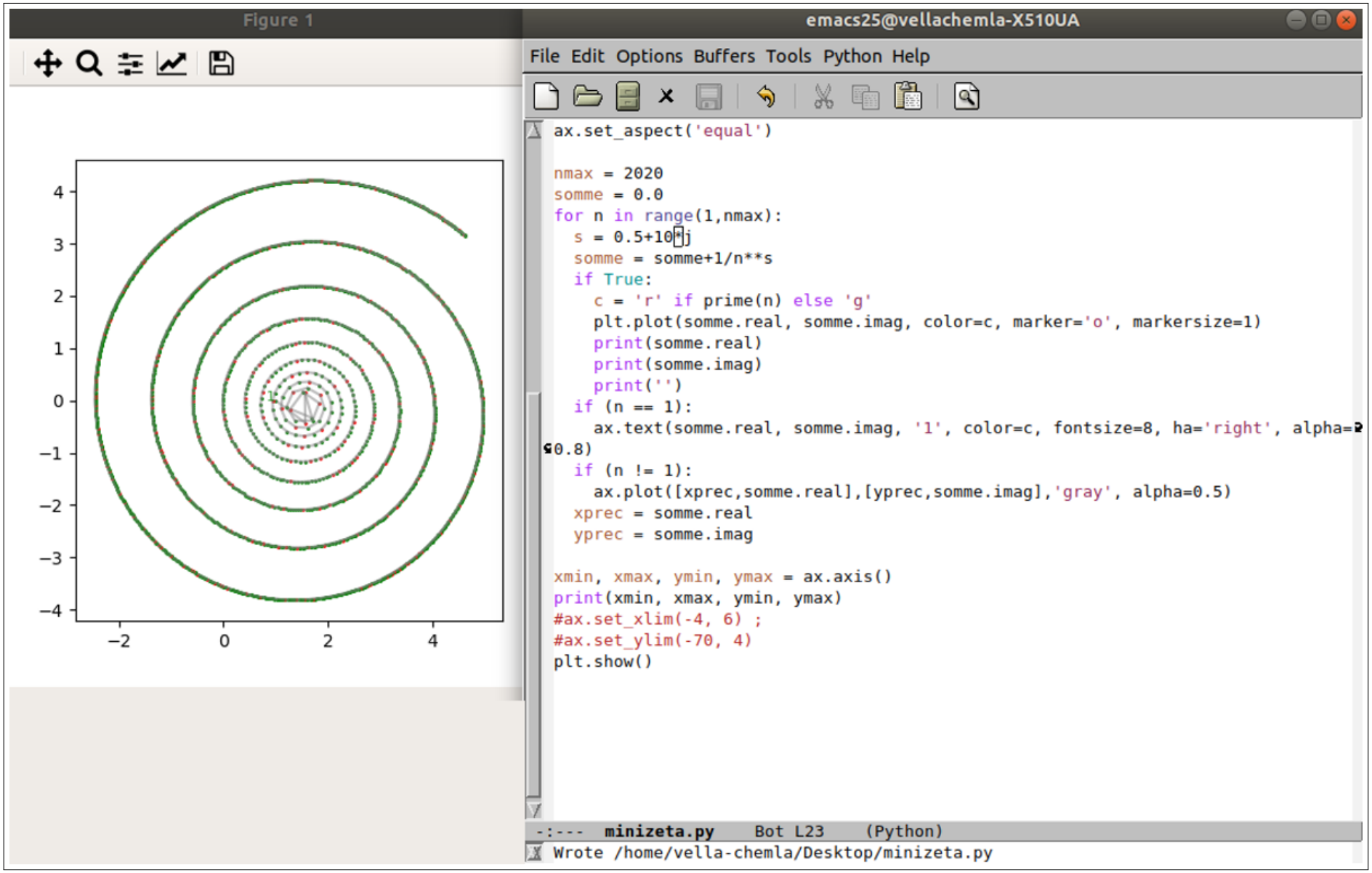


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
ax.set_aspect('equal')
nmax = 2020
somme = 0.0
for n in range(1,nmax):
s = 0.5+10*j
somme = somme+1/n**s
if True:
c = 'r' if prime(n) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
print(somme.real)
print(somme.imag)
print('')
if (n == 1):
ax.text(somme.real, somme.imag, '1', color=c, fontsize=8, ha='right', alpha=0.8)
if (n != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-4, 6) ;
#ax.set_ylim(-70, 4)
plt.show()
-:--- minizeta.py Bot L23 (Python)
Wrote /home/vella-chemla/Desktop/minizeta.py

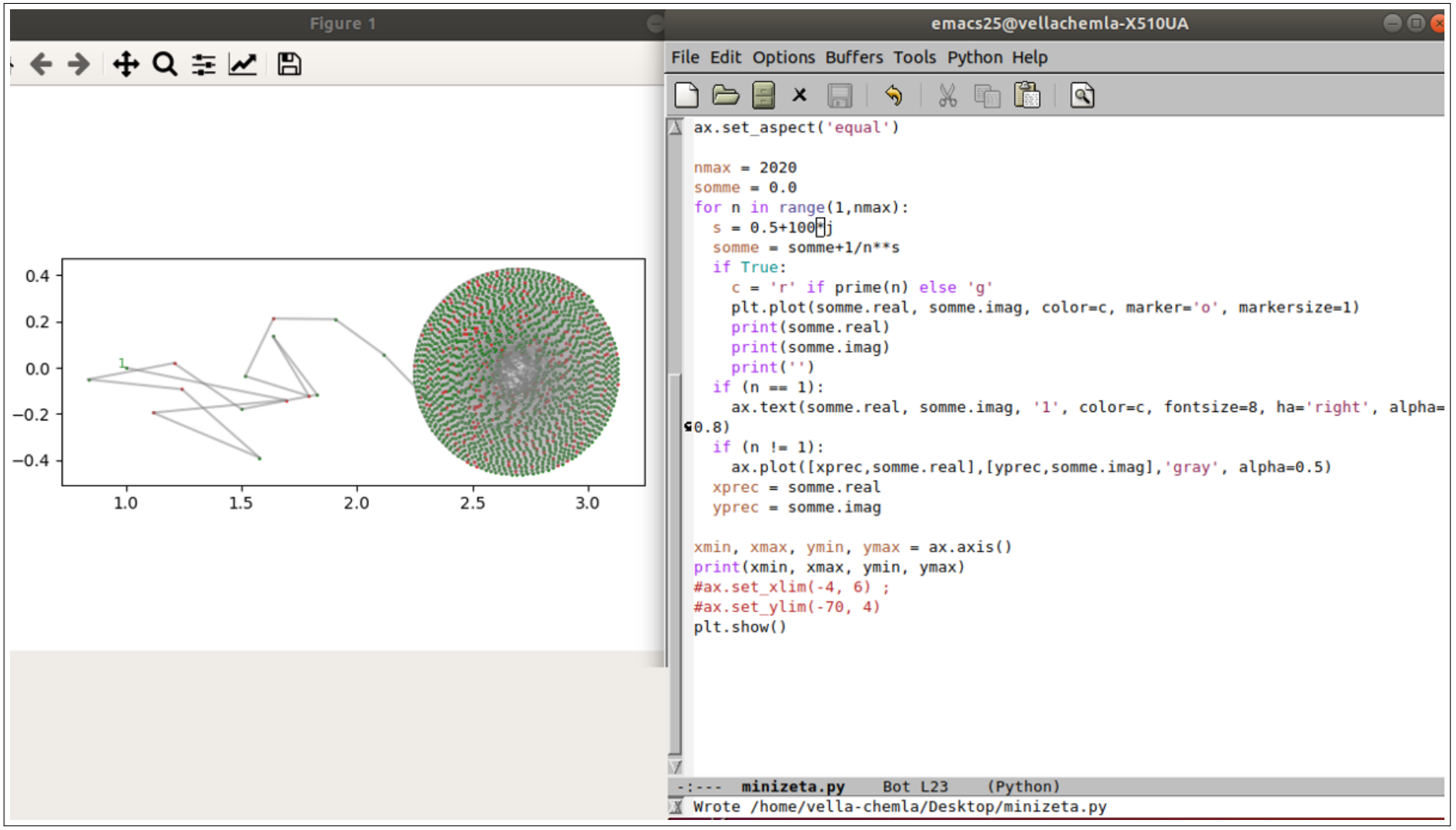


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
ax.set_aspect('equal')
nmax = 2020
somme = 0.0
for n in range(1,nmax):
s = 0.5+100*j
somme = somme+1/n**s
if True:
c = 'r' if prime(n) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
print(somme.real)
print(somme.imag)
print('')
if (n == 1):
ax.text(somme.real, somme.imag, '1', color=c, fontsize=8, ha='right', alpha=
0.8)
if (n != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-4, 6) ;
#ax.set_ylim(-70, 4)
plt.show()
-:--- minizeta.py Bot L23 (Python)
Wrote /home/vella-chemla/Desktop/minizeta.py
0.4
0.2
0.0
−0.2
−0.4
1.0
1.5
2.0
2.5
3.0
1

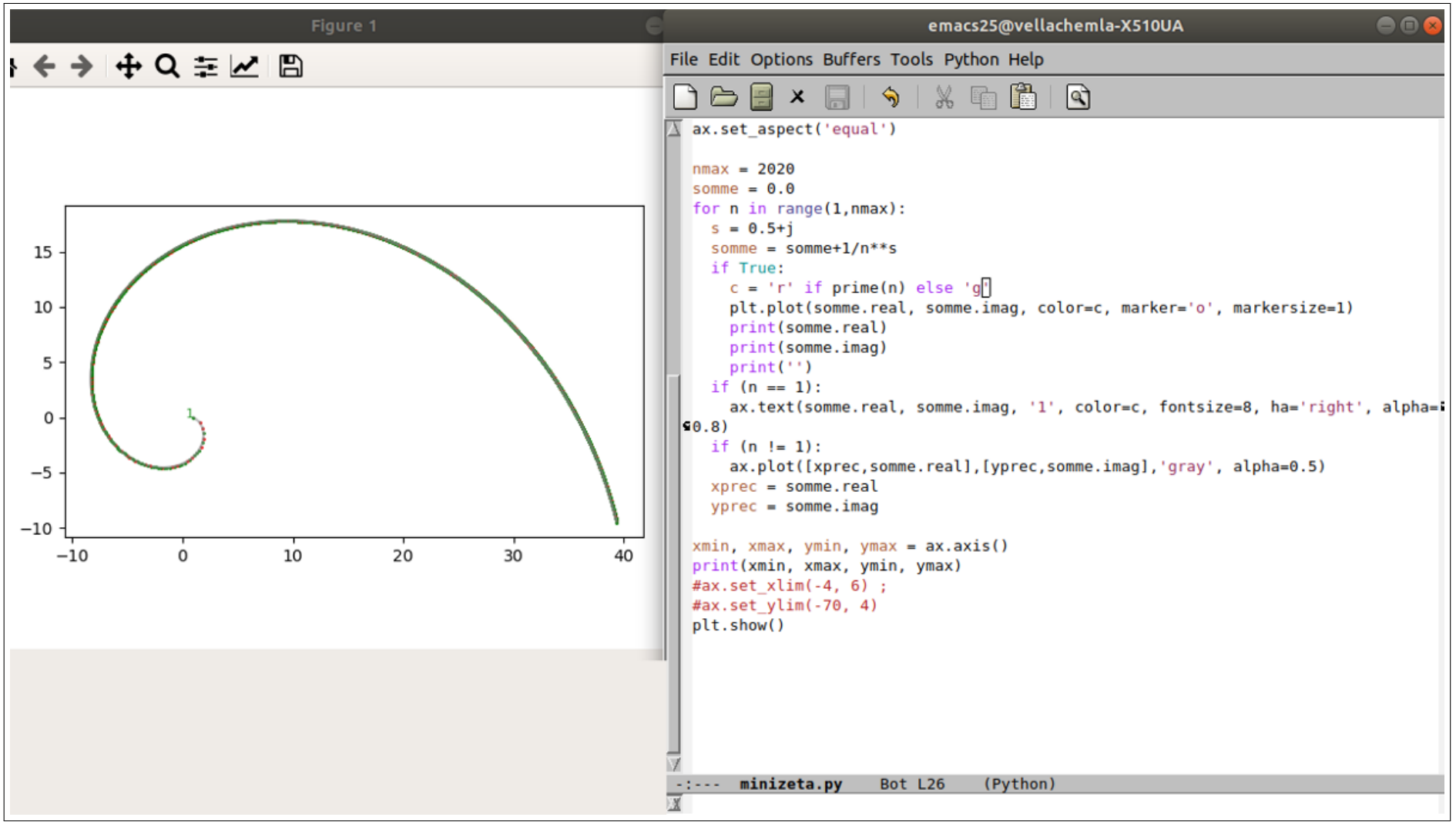


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
ax.set_aspect('equal')
nmax = 2020
somme = 0.0
for n in range(1,nmax):
s = 0.5+j
somme = somme+1/n**s
if True:
c = 'r' if prime(n) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
print(somme.real)
print(somme.imag)
print('')
if (n == 1):
ax.text(somme.real, somme.imag, '1', color=c, fontsize=8, ha='right', alpha=0.8)
if (n != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-4, 6) ;
#ax.set_ylim(-70, 4)
plt.show()
-:--- minizeta.py Bot L26 (Python)
15
10
5
0
−5
−10
−10
0
10
20
30
40
1

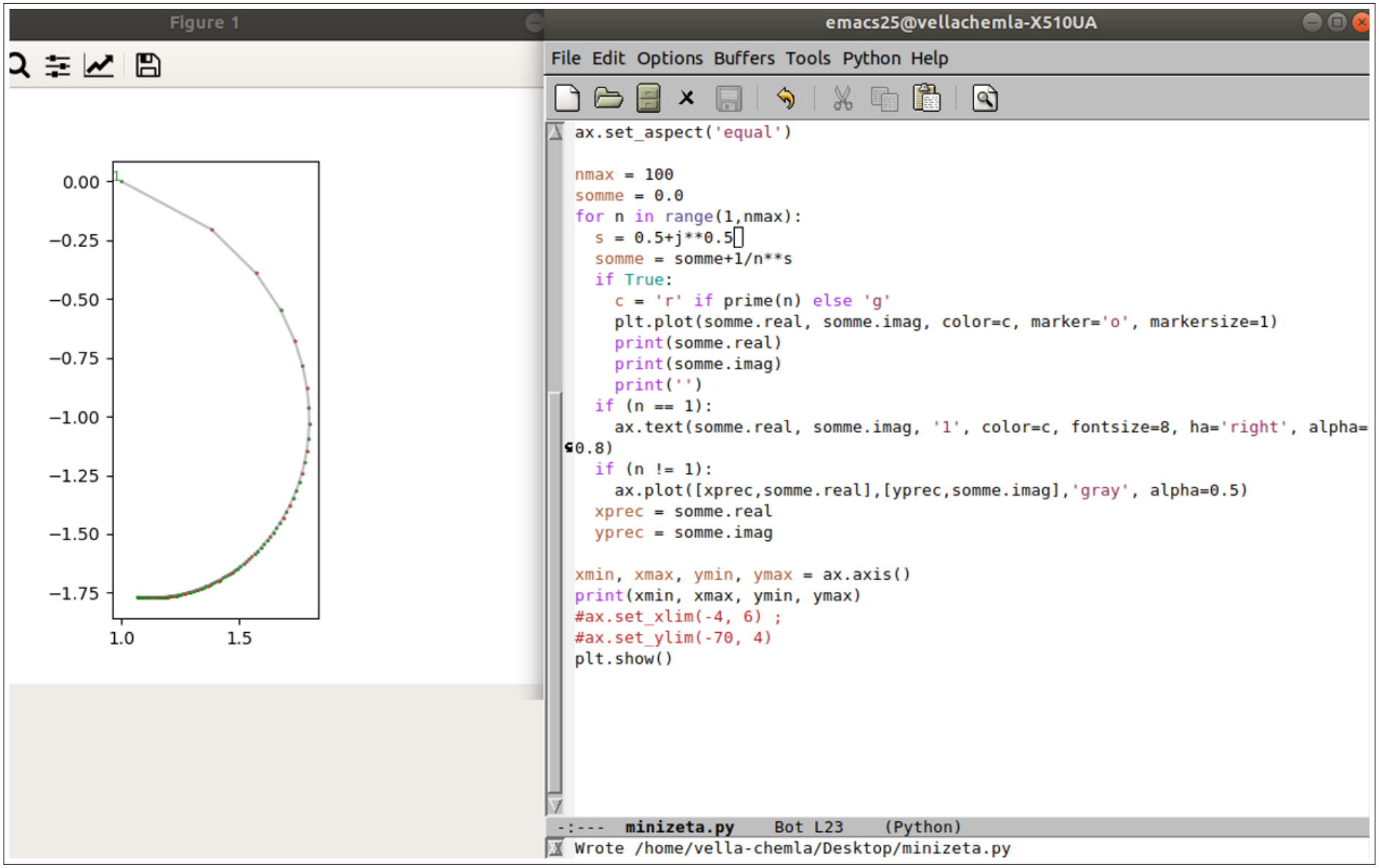


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
ax.set_aspect('equal')
nmax = 100
somme = 0.0
for n in range(1,nmax):
s = 0.5+j**0.5
somme = somme+1/n**s
if True:
c = 'r' if prime(n) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
print(somme.real)
print(somme.imag)
print('')
if (n == 1):
ax.text(somme.real, somme.imag, '1', color=c, fontsize=8, ha='right', alpha=
0.8)
if (n != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-4, 6) ;
#ax.set_ylim(-70, 4)
plt.show()
-:--- minizeta.py Bot L23 (Python)
Wrote /home/vella-chemla/Desktop/minizeta.py
0.00
−0.25
−0.50
−0.75
−1.00
−1.25
−1.50
−1.75
1.0
1.5

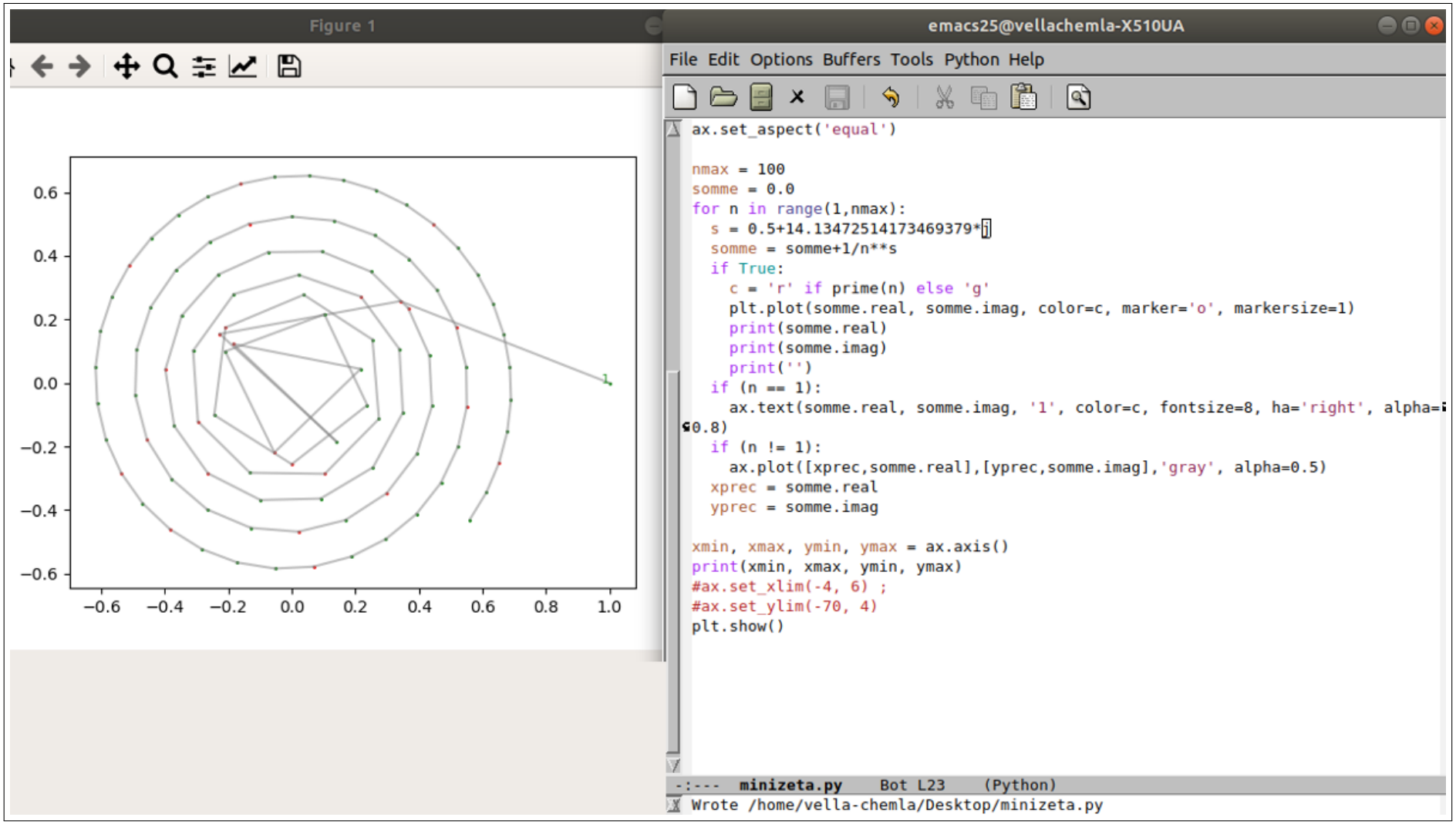


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
ax.set_aspect('equal')
nmax = 100
somme = 0.0
for n in range(1,nmax):
s = 0.5+14.13472514173469379*j
somme = somme+1/n**s
if True:
c = 'r' if prime(n) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
print(somme.real)
print(somme.imag)
print('')
if (n == 1):
ax.text(somme.real, somme.imag, '1', color=c, fontsize=8, ha='right', alpha=
0.8)
if (n != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-4, 6) ;
#ax.set_ylim(-70, 4)
plt.show()
-:--- minizeta.py Bot L23 (Python)
Wrote /home/vella-chemla/Desktop/minizeta.py
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.6 −0.4 −0.2 0.0 0.2 0.4 0.6 0.8 1.0
1

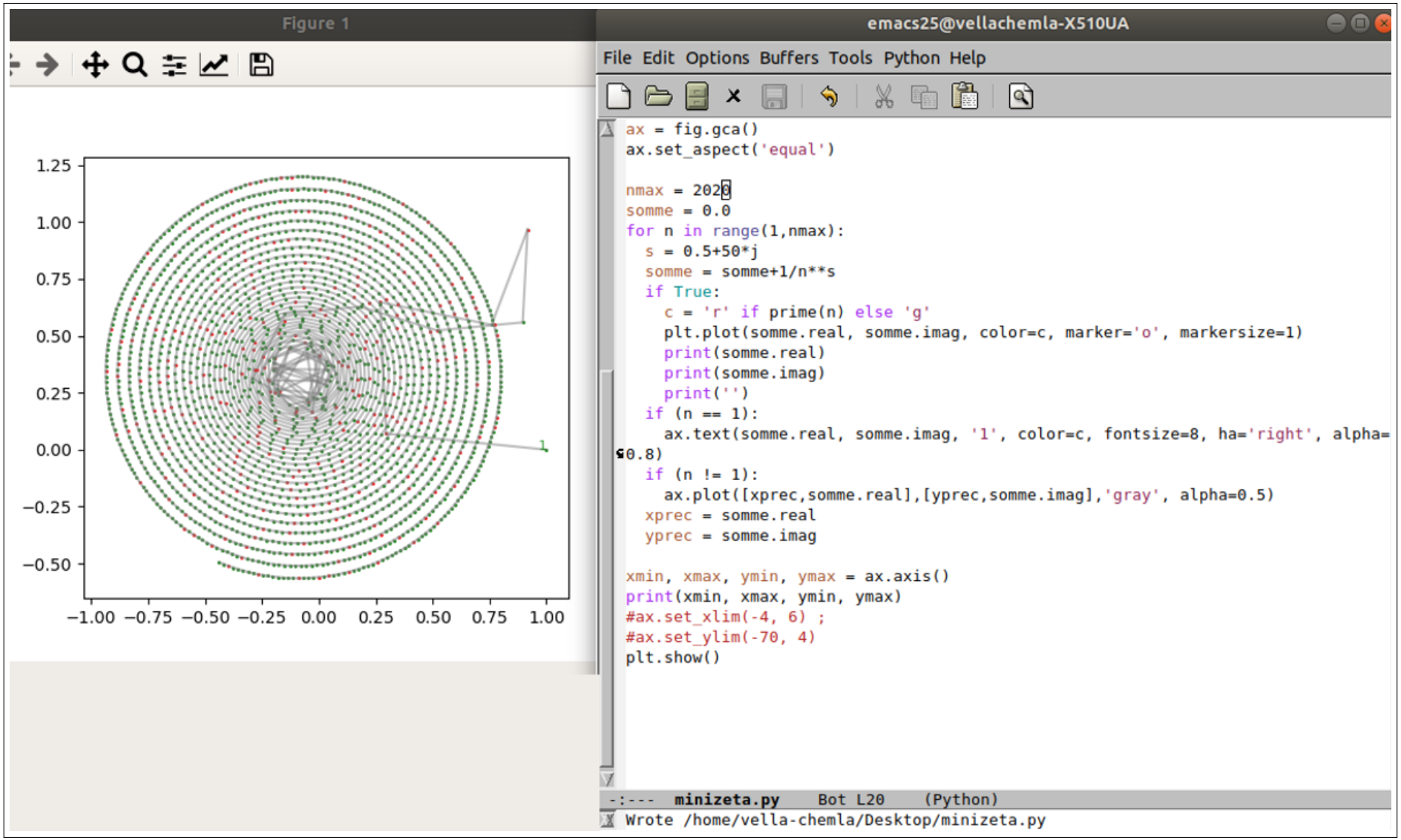
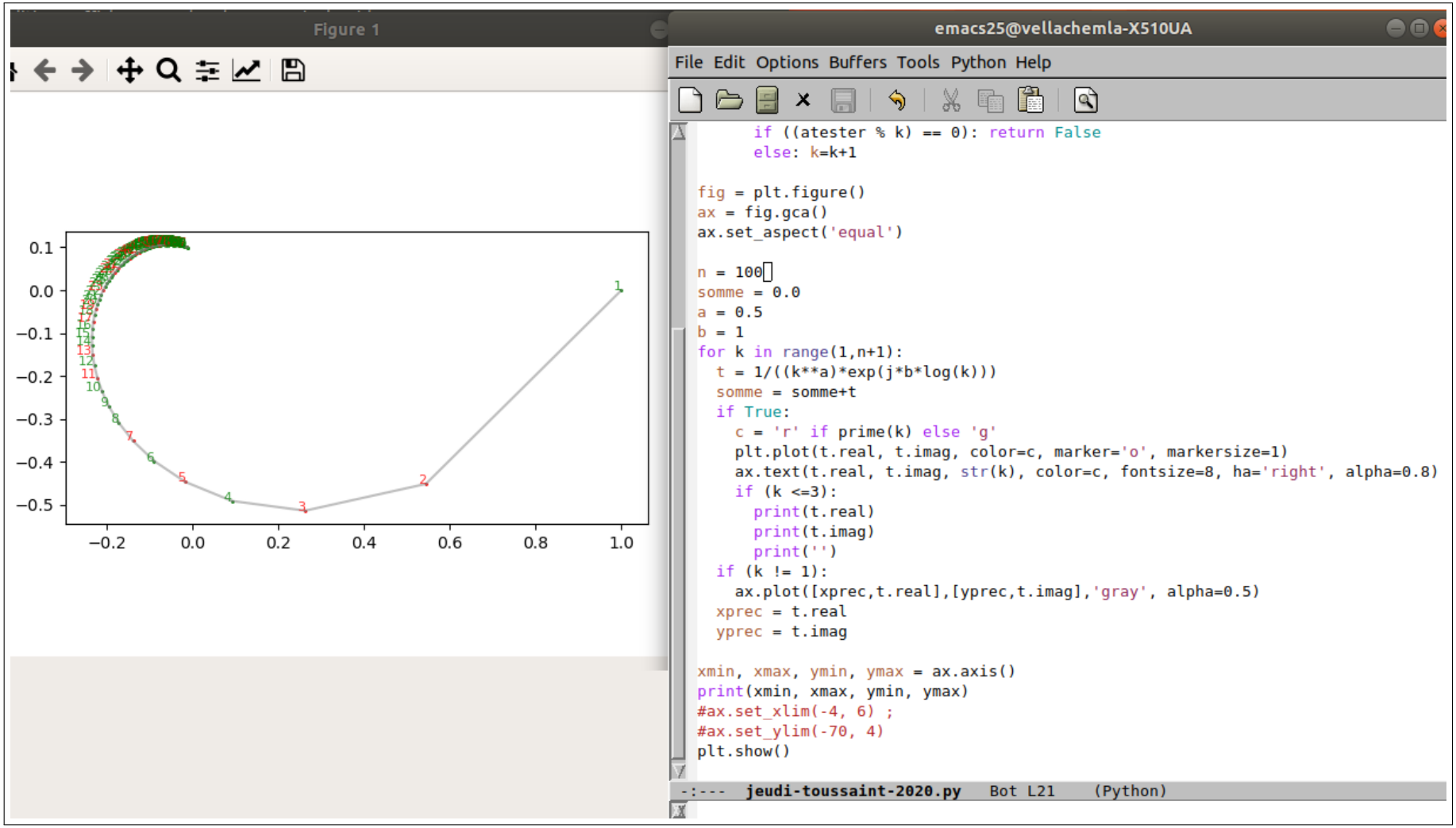


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
ax = fig.gca()
ax.set_aspect('equal')
nmax = 2020
somme = 0.0
for n in range(1,nmax):
s = 0.5+50*j
somme = somme+1/n**s
if True:
c = 'r' if prime(n) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
print(somme.real)
print(somme.imag)
print('')
if (n == 1):
ax.text(somme.real, somme.imag, '1', color=c, fontsize=8, ha='right', alpha=0.8)
if (n != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-4, 6) ;
#ax.set_ylim(-70, 4)
plt.show()
-:--- minizeta.py Bot L20 (Python)
Wrote /home/vella-chemla/Desktop/minizeta.py
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00 −0.75 −0.50 −0.25 0.00 0.25 0.50 0.75 1.00
1

Multiples spirales (Denise Vella-Chemla, 22.10.2020)

Dans les dessins ci-dessous, on "plotte" soit $\frac{1}{k^s} = k^{-s} = \frac{1}{k^a \exp(i\, b \ln(k))}$ pour k de 1 à n, soit la somme cumulée de tels éléments.

$n = 100, a = 0.5, b = 1.$



$n = 1000, a = 0.5, b = 1.$

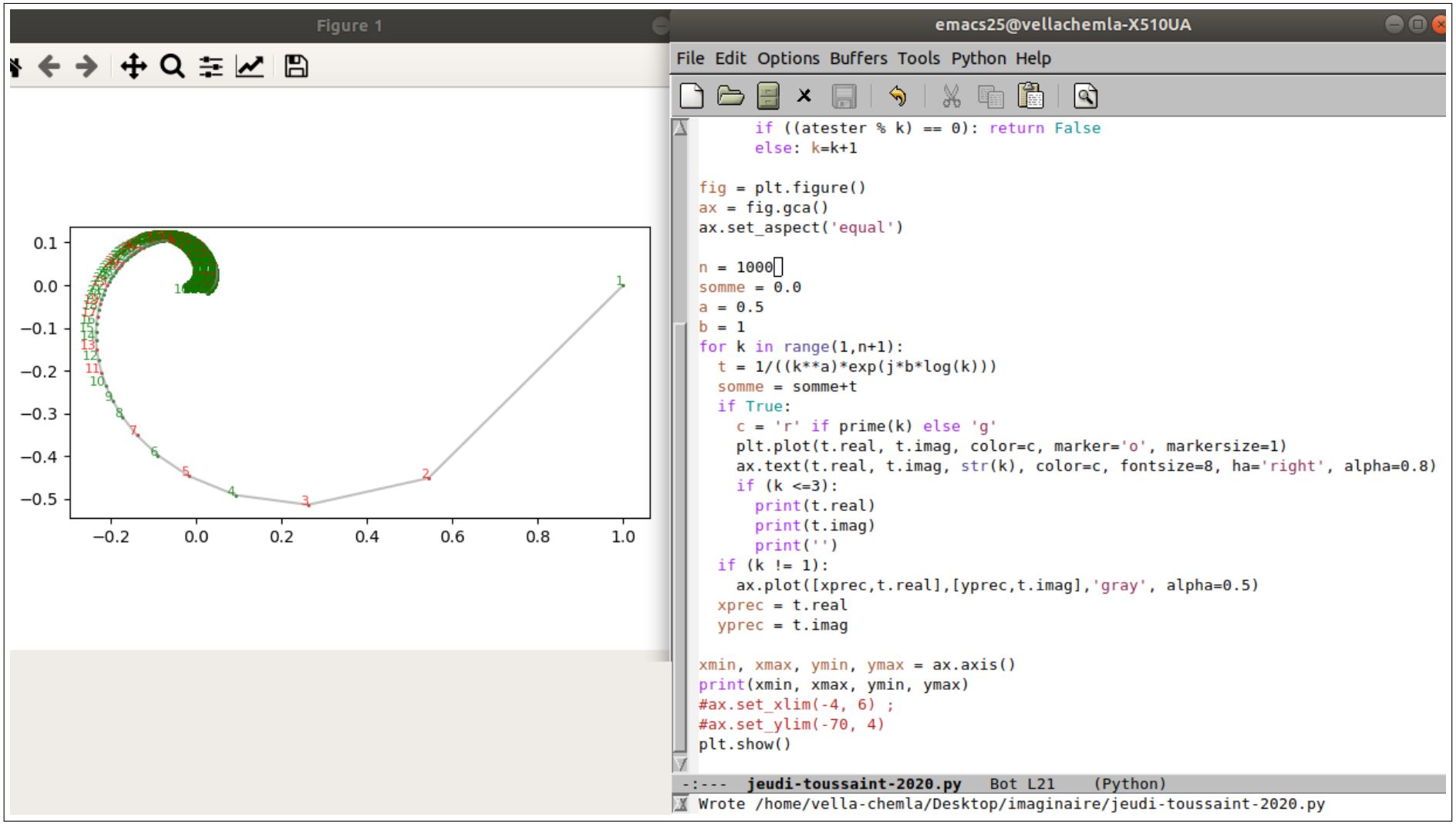
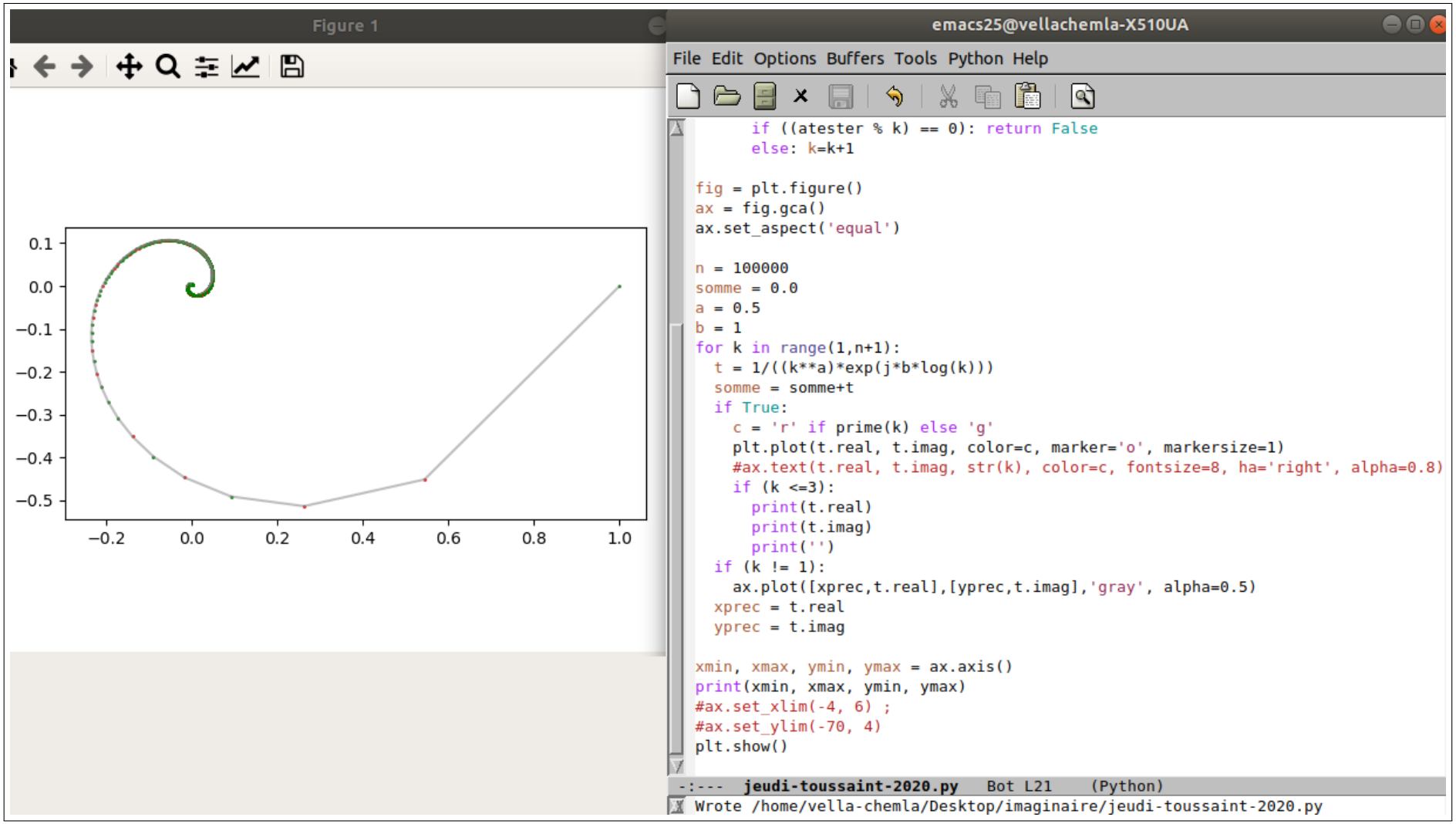


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 0.5
b = 1
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-4, 6) ;
#ax.set_ylim(-70, 4)
plt.show()
-:--- jeudi-toussaint-2020.py Bot L21 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

$n = 100000, a = 0.5, b = 1.$



$n = 1000, a = 0.5, b = 5.$

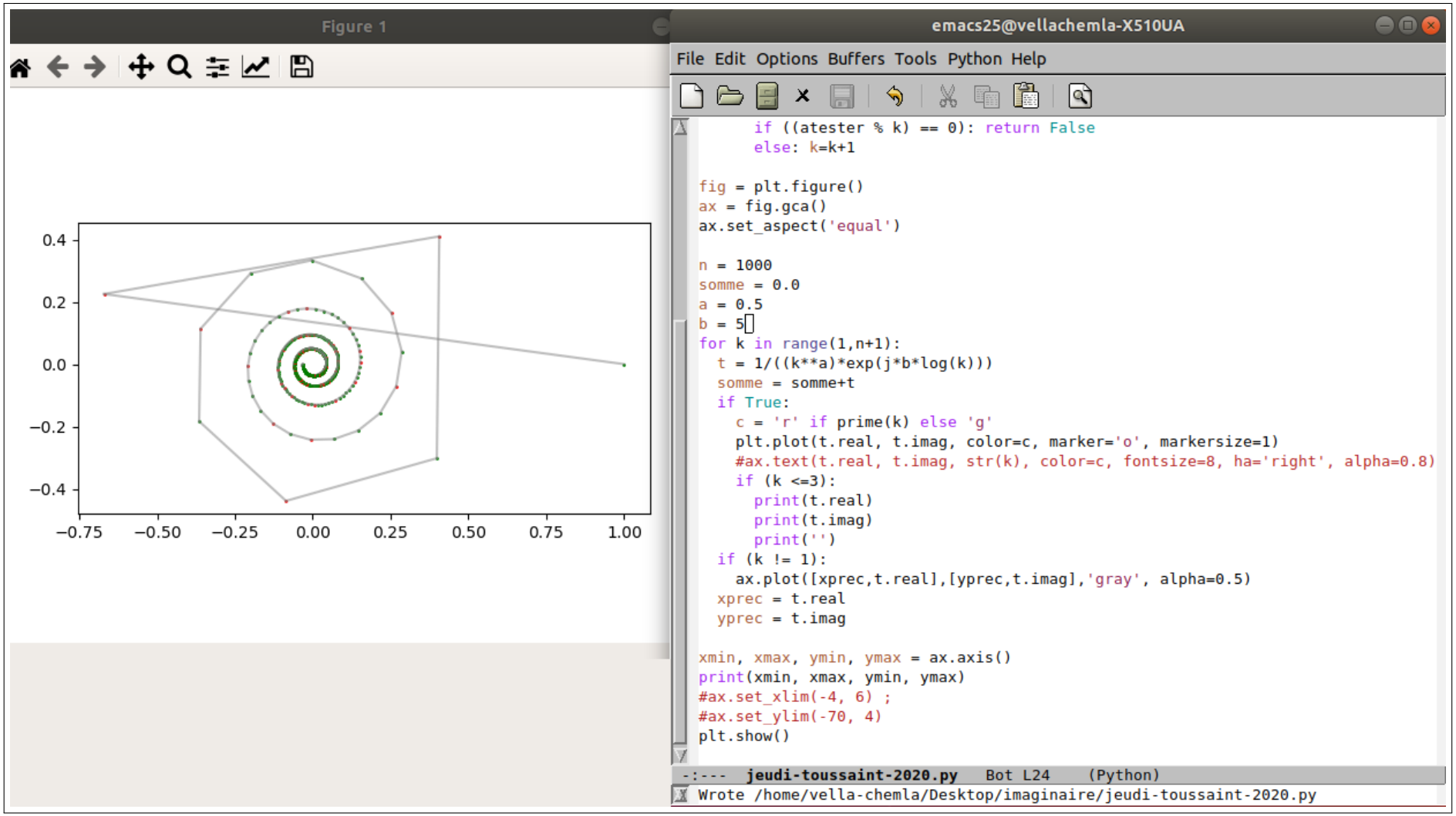
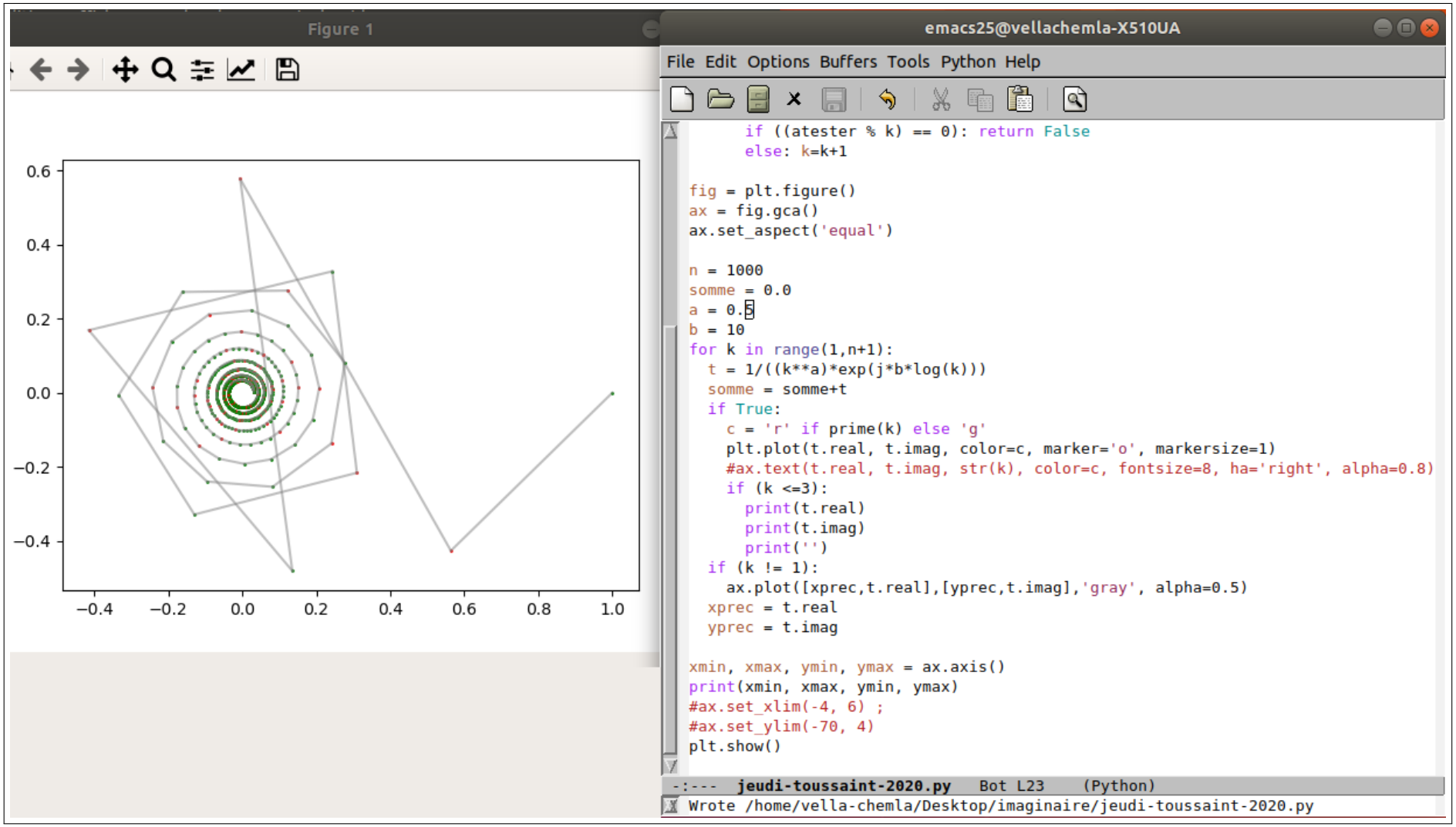
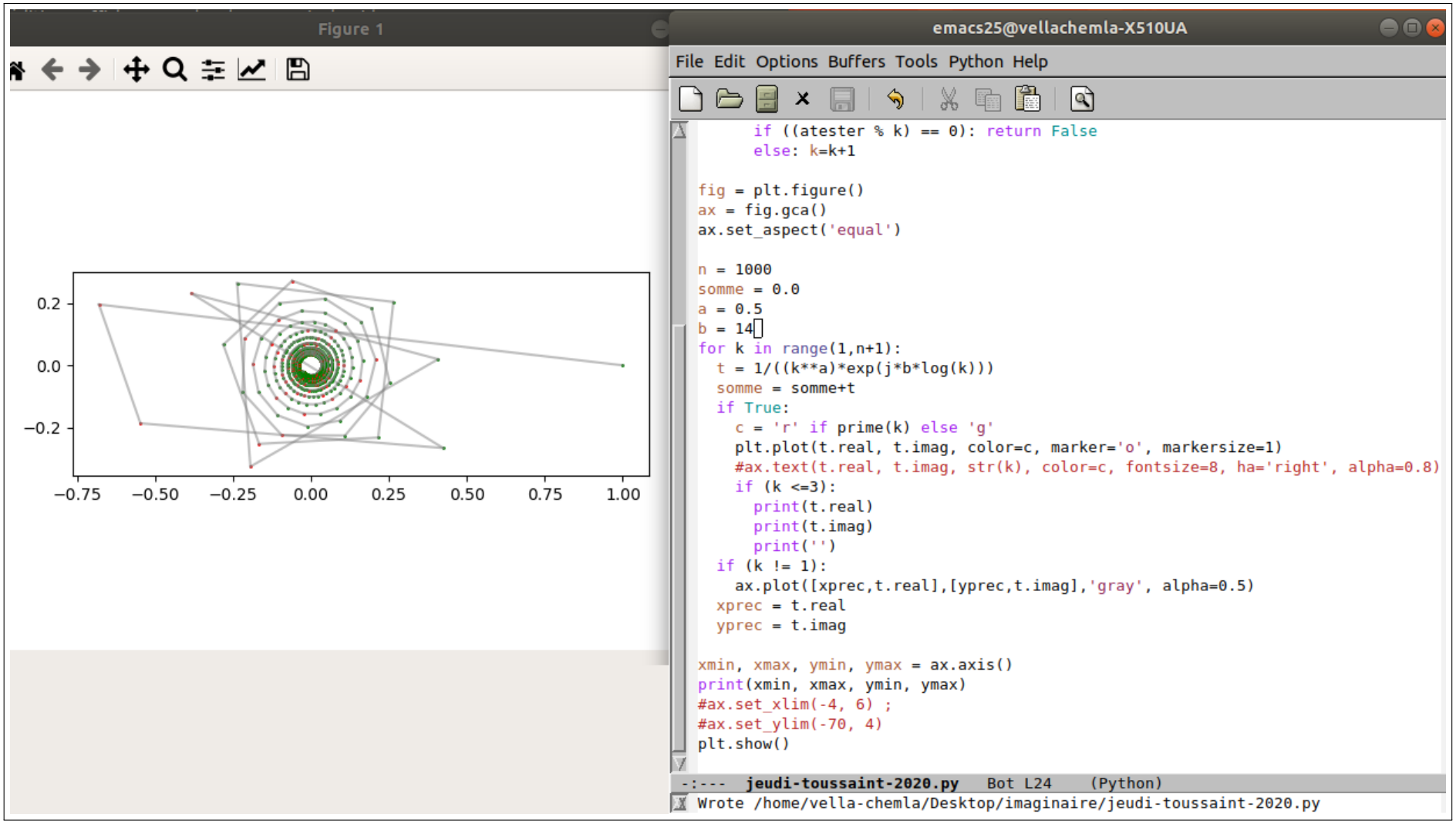


Figure 1
0.4
0.2
0.0
−0.2
−0.4
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 0.5
b = 5
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-4, 6) ;
#ax.set_ylim(-70, 4)
plt.show()
-:--- jeudi-toussaint-2020.py Bot L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

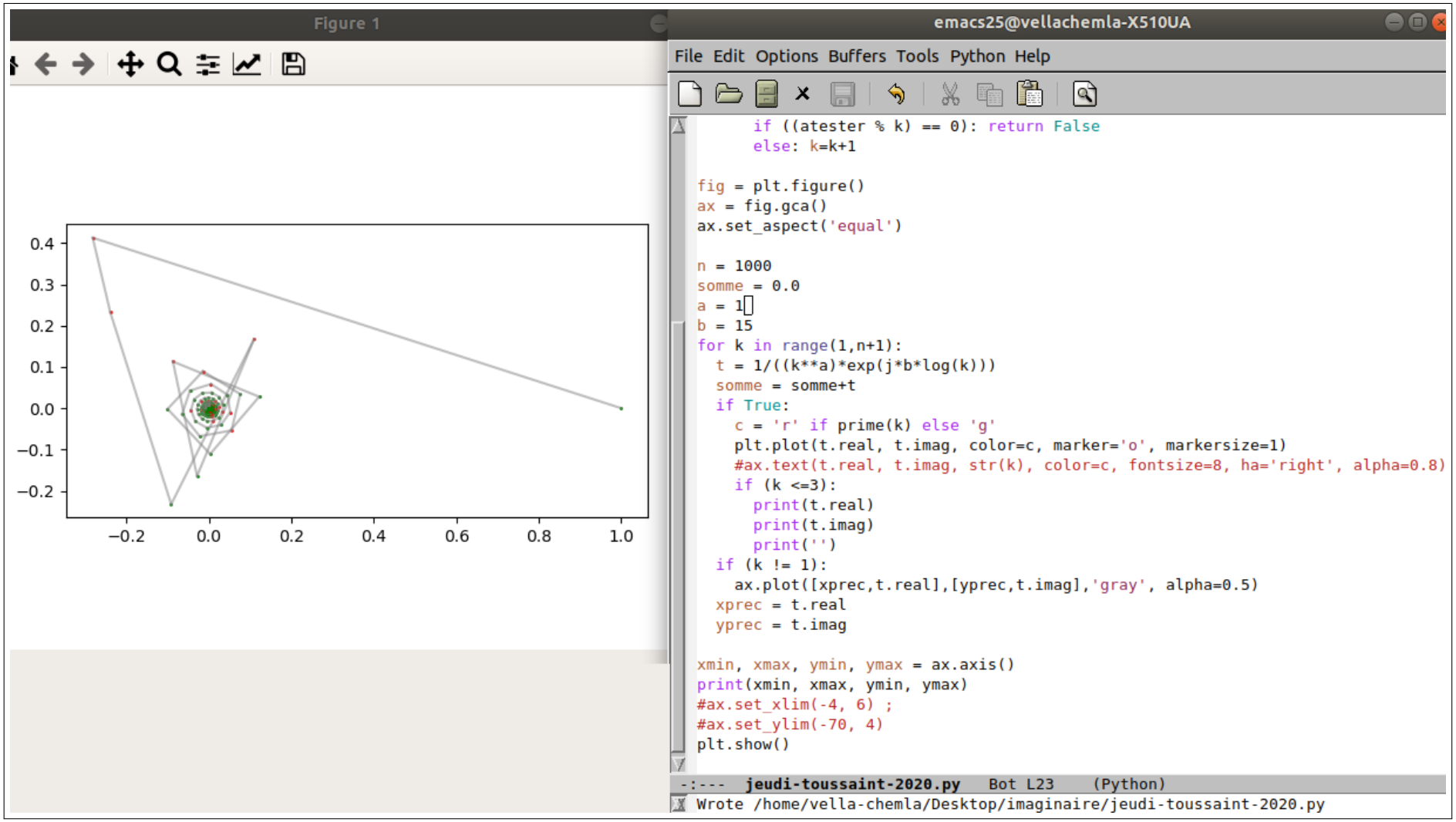
$n = 1000, a = 0.5, b = 10.$



$n = 1000, a = 0.5, b = 14.$



$n = 1000, a = 0.5, b = 15.$



$n = 1000, a = 3, b = 15.$

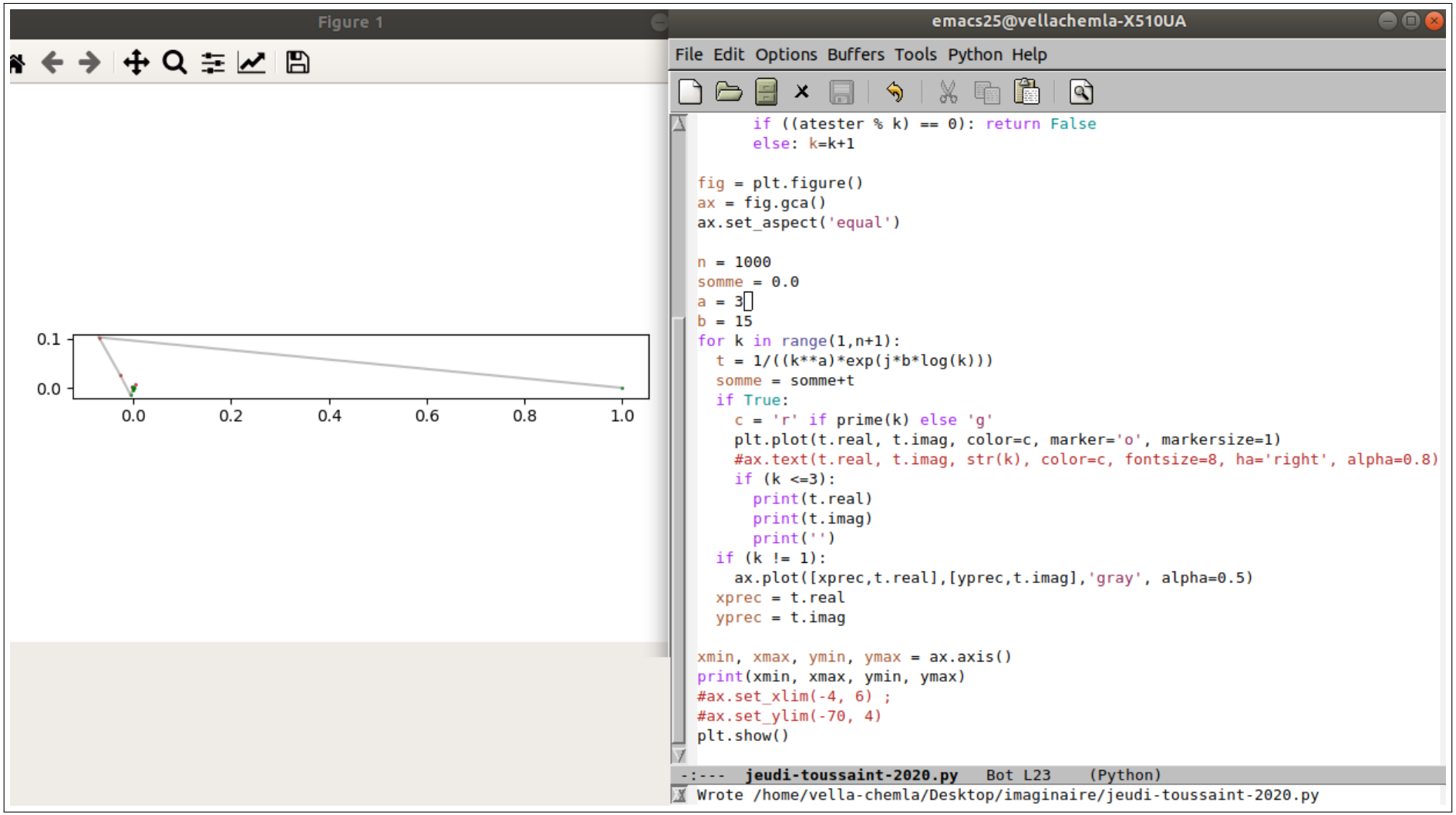


Figure 1
0.1
0.0
0.0
0.2
0.4
0.6
0.8
1.0
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 3
b = 15
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-4, 6) ;
#ax.set_ylim(-70, 4)
plt.show()
-:--- jeudi-toussaint-2020.py Bot L23 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

$n = 1000, a = 4, b = 15.$

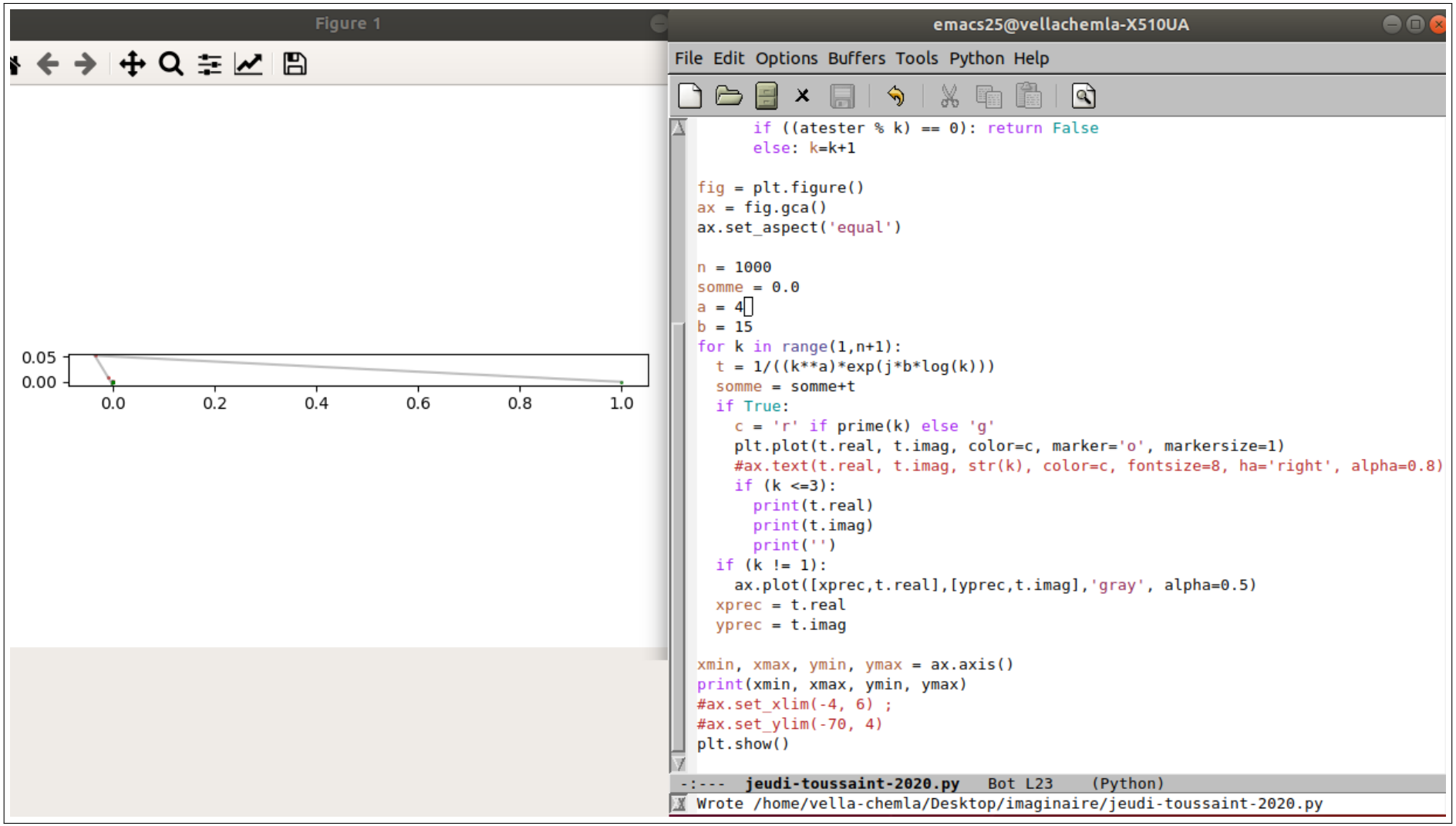


Figure 1
0.05
0.00
0.0
0.2
0.4
0.6
0.8
1.0
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 4
b = 15
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-4, 6) ;
#ax.set_ylim(-70, 4)
plt.show()
-:--- jeudi-toussaint-2020.py Bot L23 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

$n = 1000, a = 5, b = 15.$

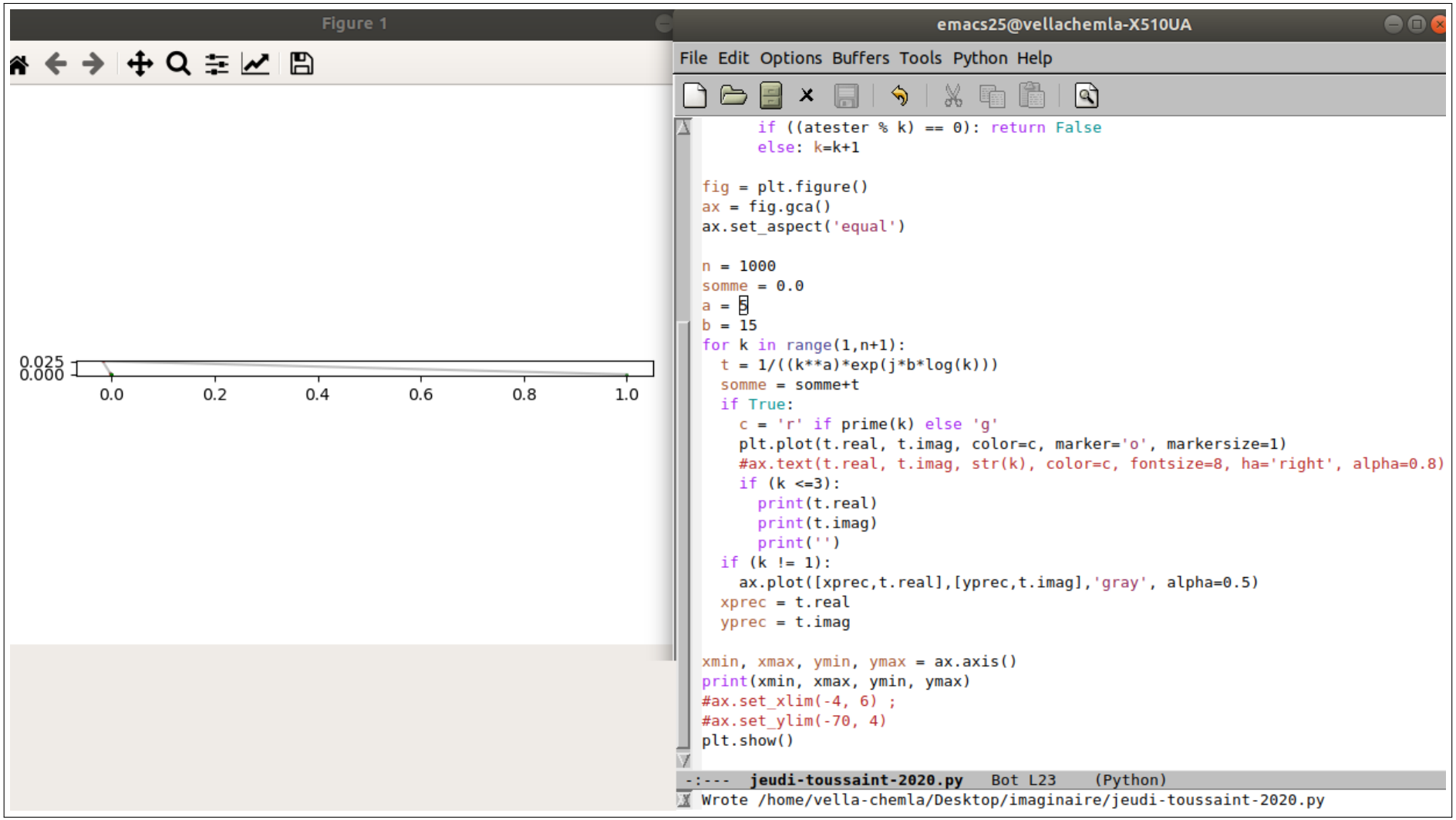


Figure 1
0.025
0.000
0.0
0.2
0.4
0.6
0.8
1.0
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 5
b = 15
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-4, 6) ;
#ax.set_ylim(-70, 4)
plt.show()
-:--- jeudi-toussaint-2020.py Bot L23 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

$n = 1000, a = 7, b = 15.$

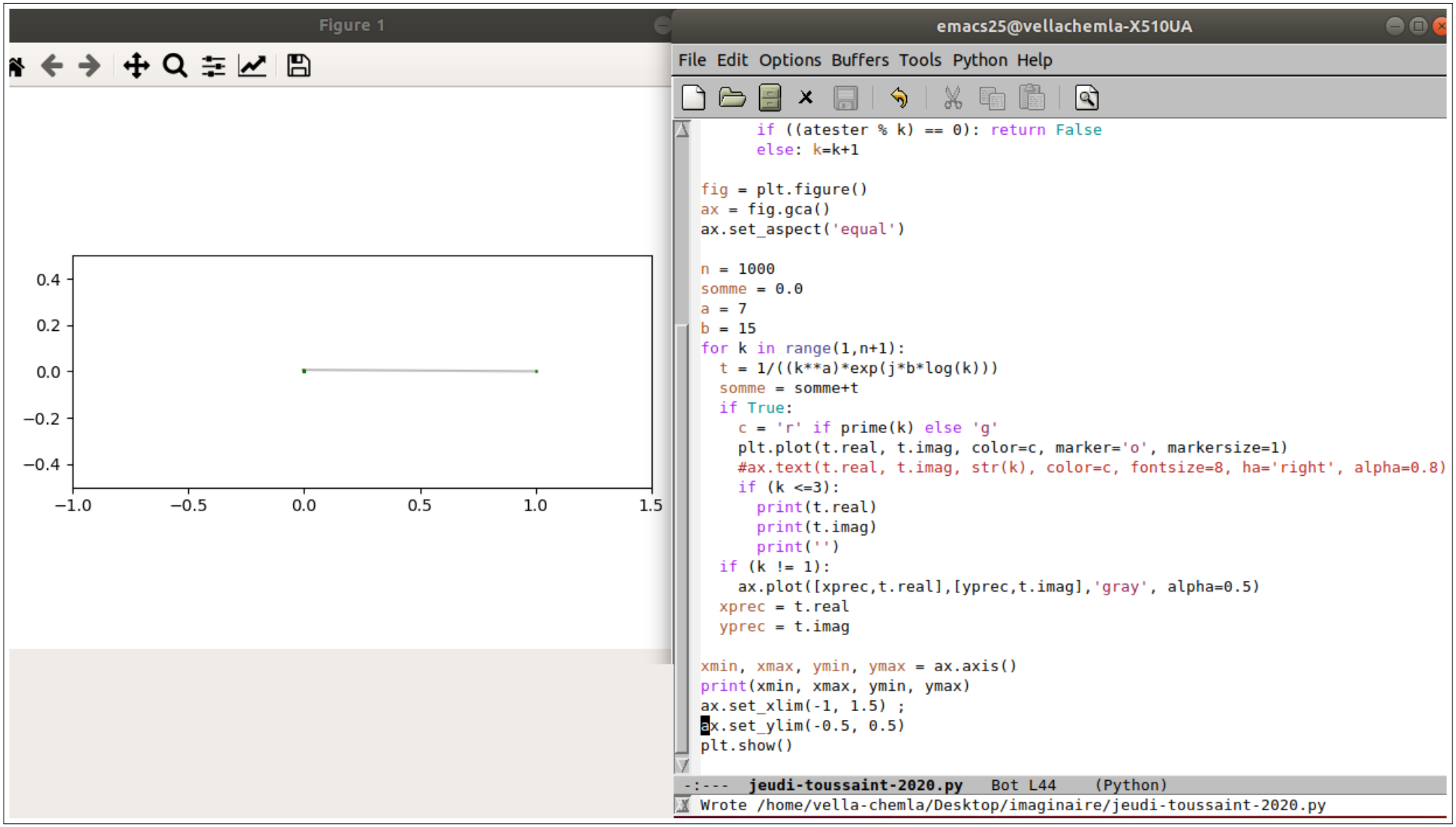
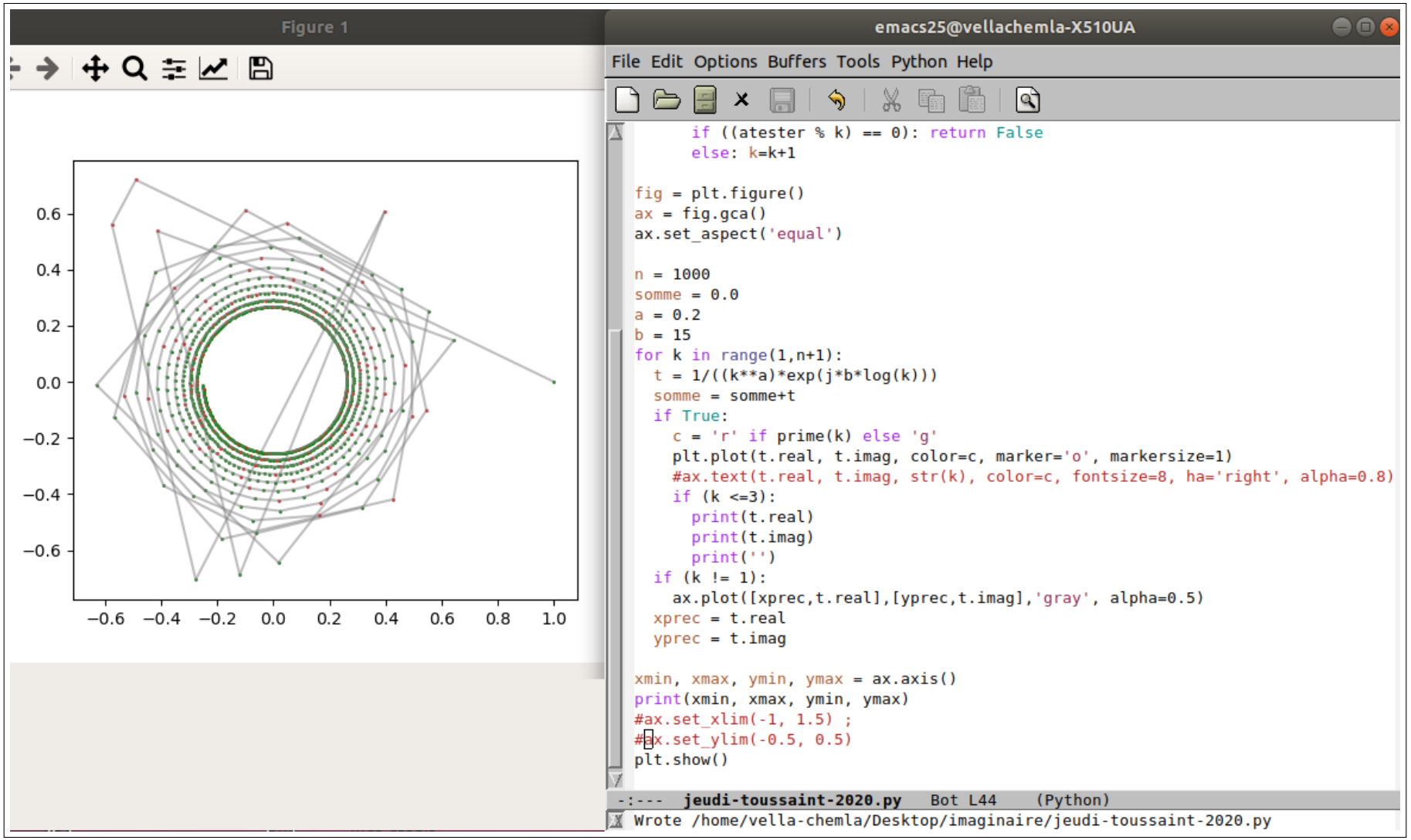
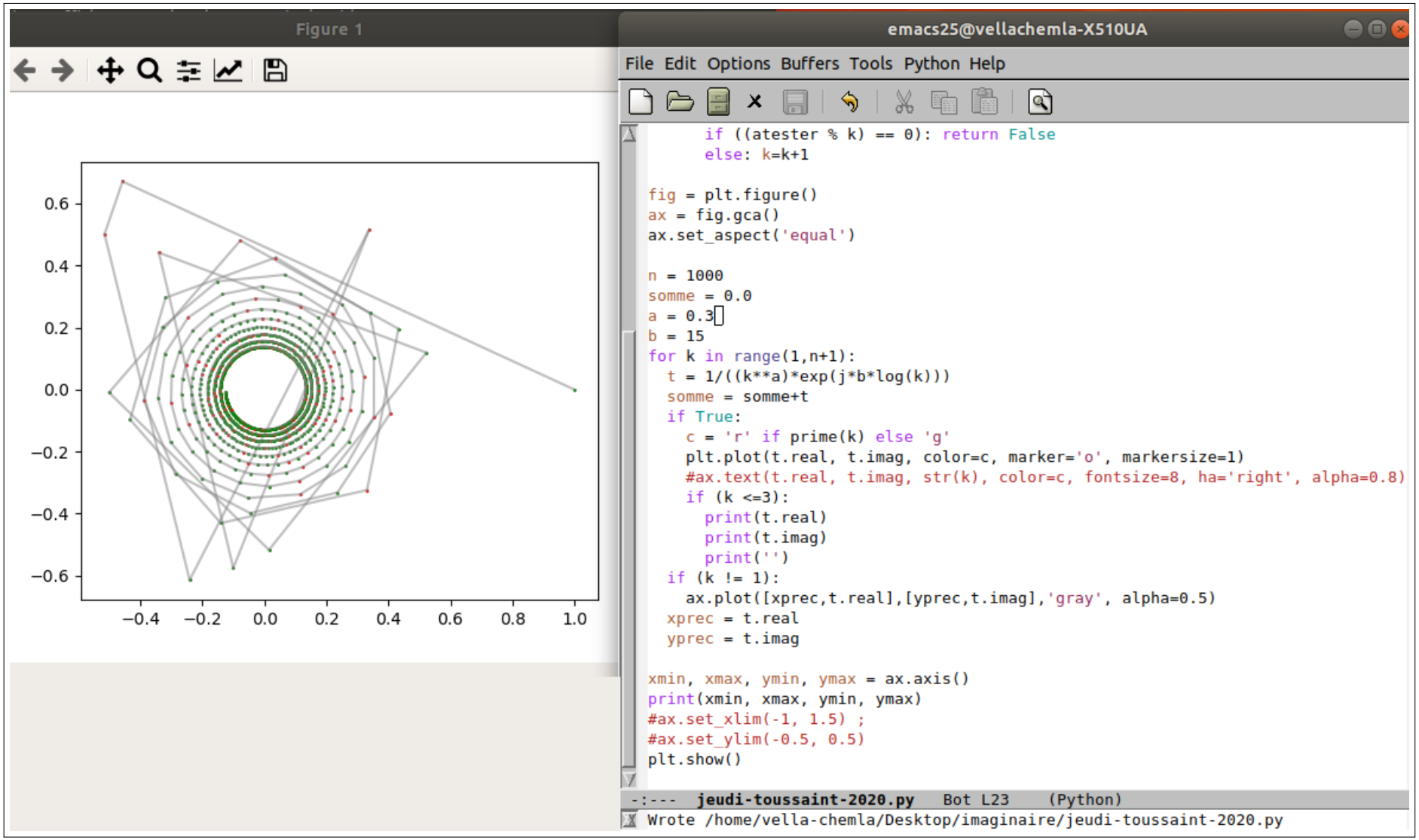


Figure 1
0.4
0.2
0.0
−0.2
−0.4
−1.0
−0.5
0.0
0.5
1.0
1.5
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 7
b = 15
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
ax.set_xlim(-1, 1.5) ;
ax.set_ylim(-0.5, 0.5)
plt.show()
-:--- jeudi-toussaint-2020.py Bot L44 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

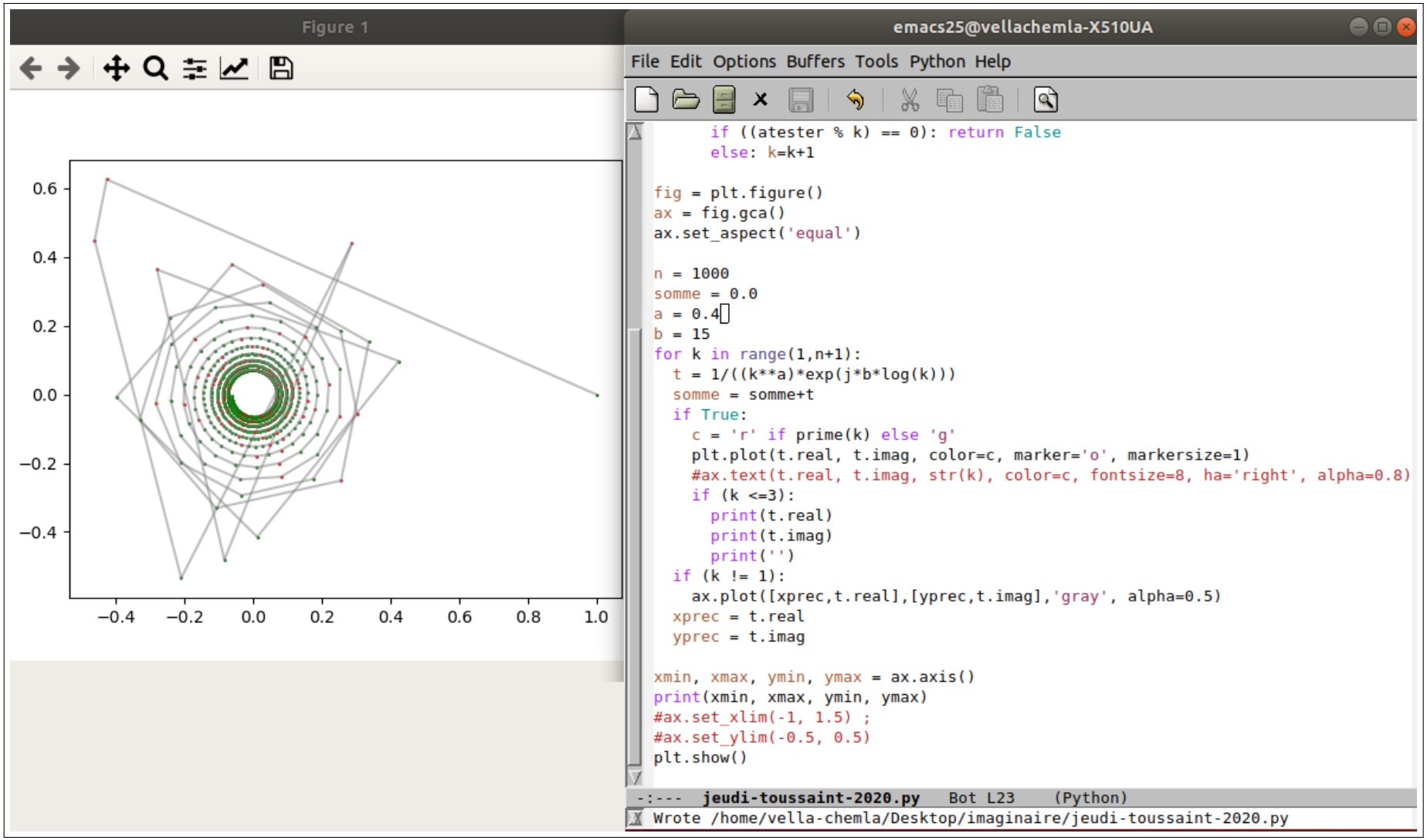
$n = 1000, a = 0.2, b = 15.$



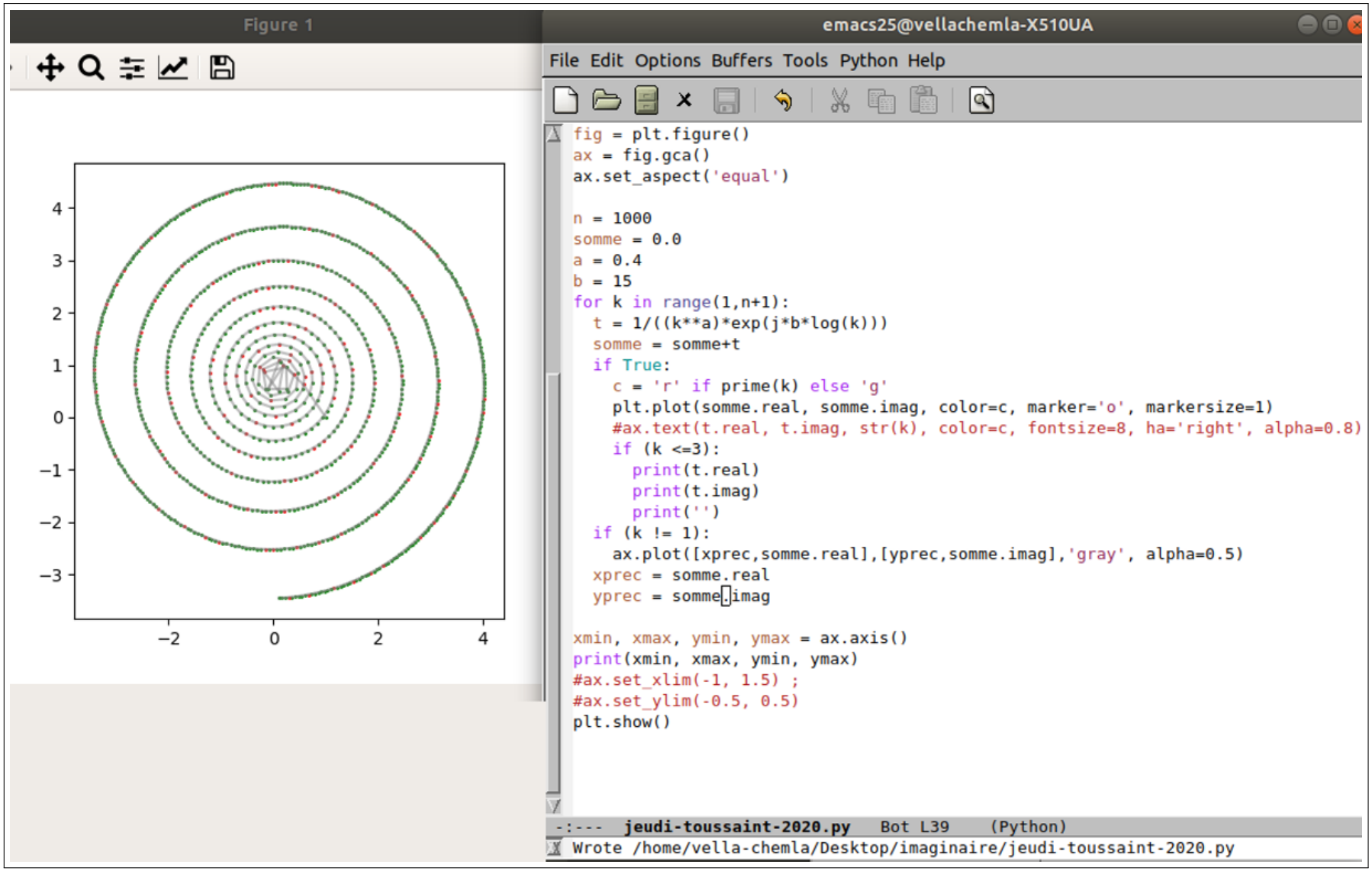
$n = 1000, a = 0.3, b = 15.$



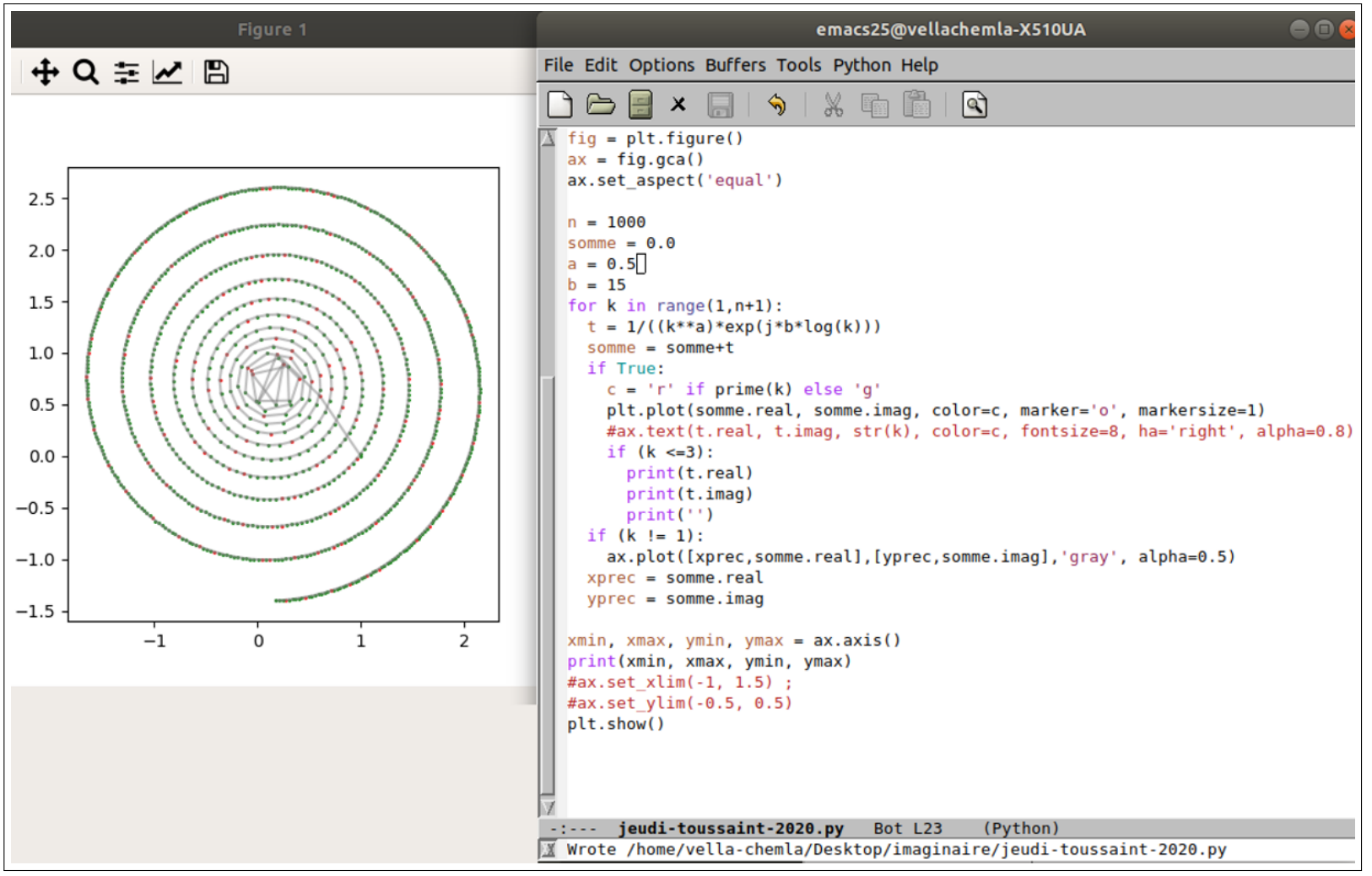
$n = 1000, a = 0.4, b = 15.$



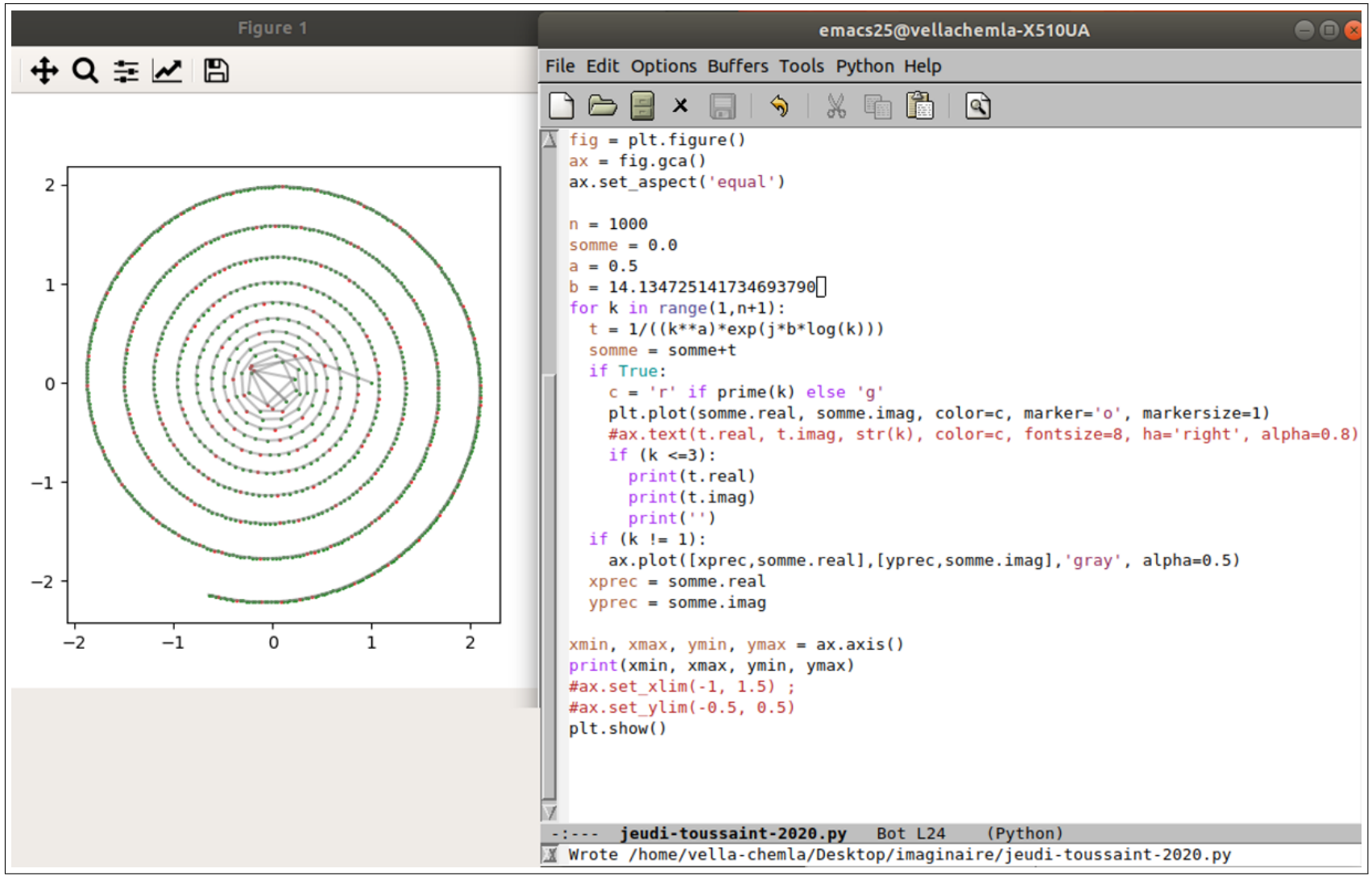
$n = 1000, a = 0.4, b = 15$. On “plotte” la somme au lieu de “plotter” chacun de ses termes.



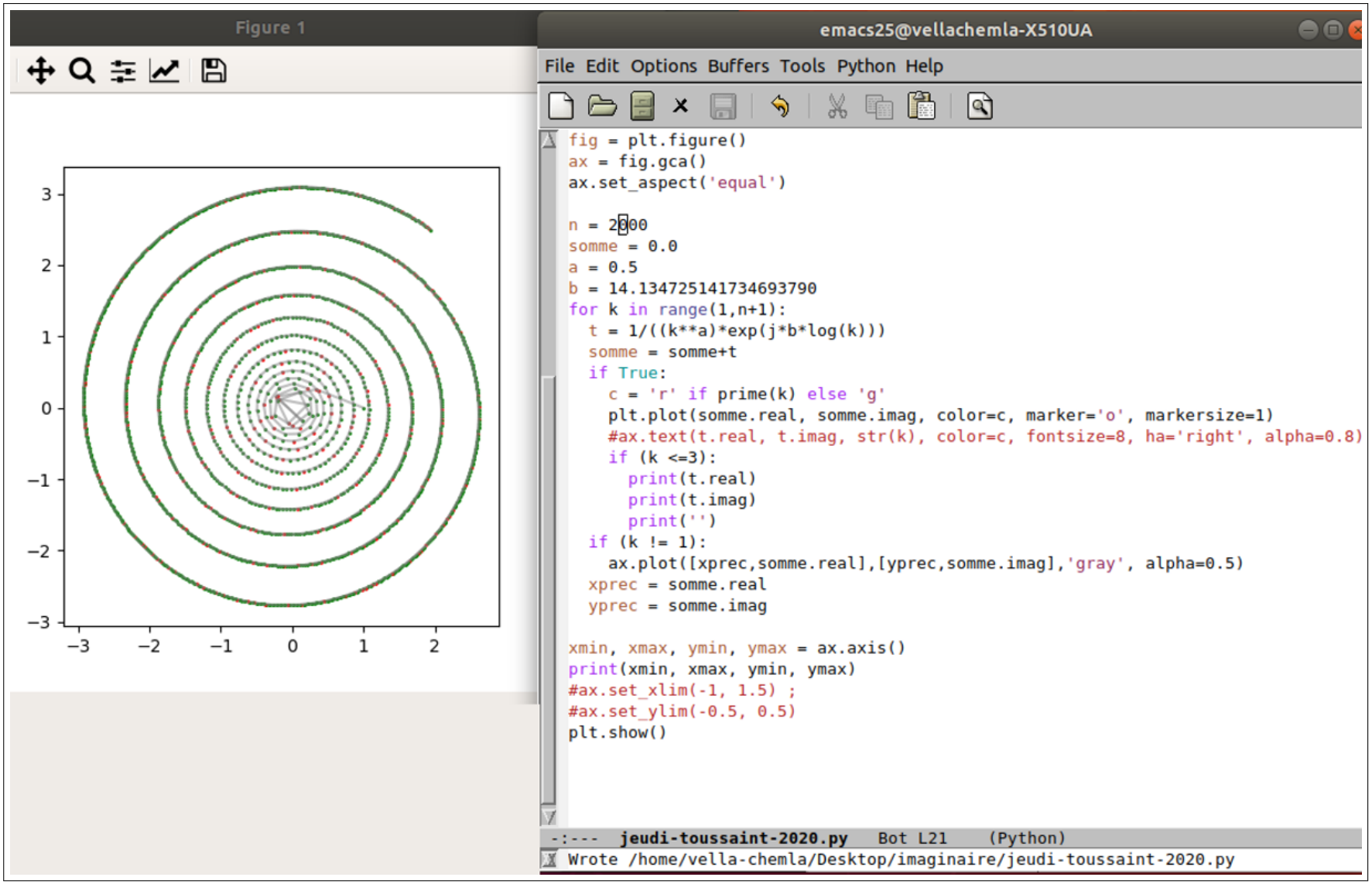
$n = 1000, a = 0.5, b = 15$. On “plotte” la somme au lieu de “plotter” chacun de ses termes.



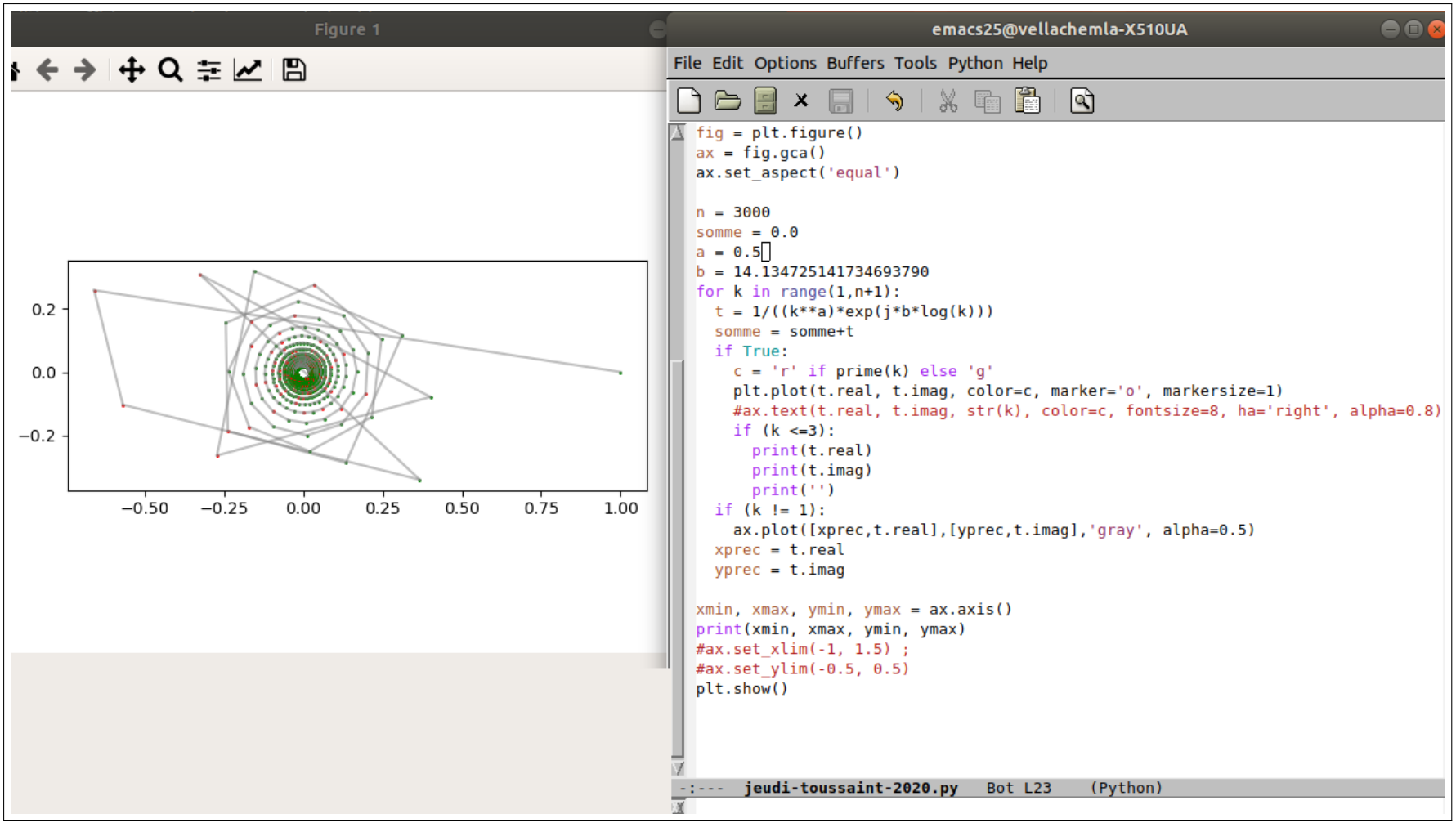
$n = 1000, a = 0.5, b = 14.134725141734693790$. On “plotte” la somme au lieu de “plotter” chacun de ses termes.



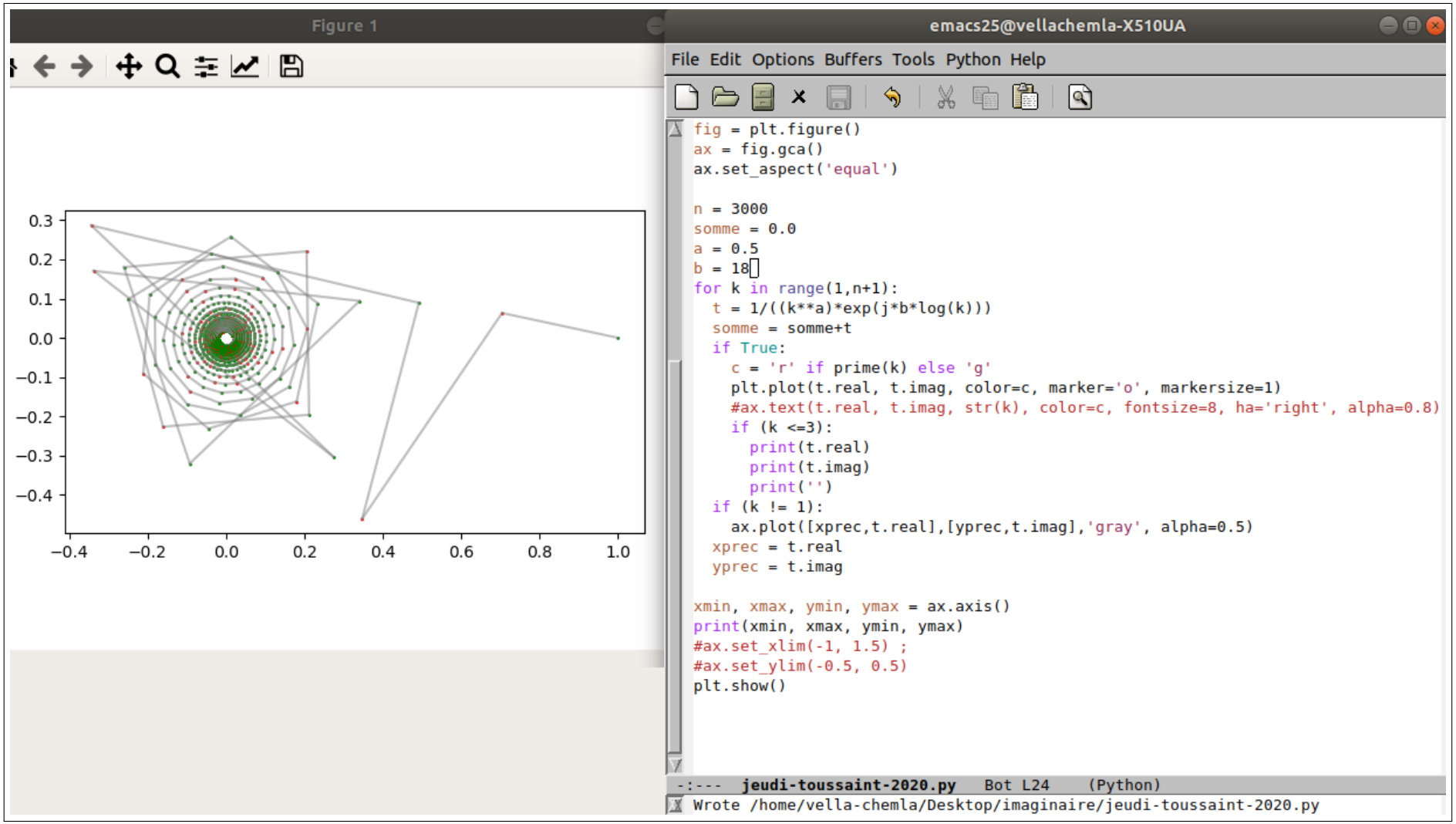
$n = 2000, a = 0.5, b = 14.134725141734693790$. On “plotte” la somme au lieu de “plotter” chacun de ses termes.



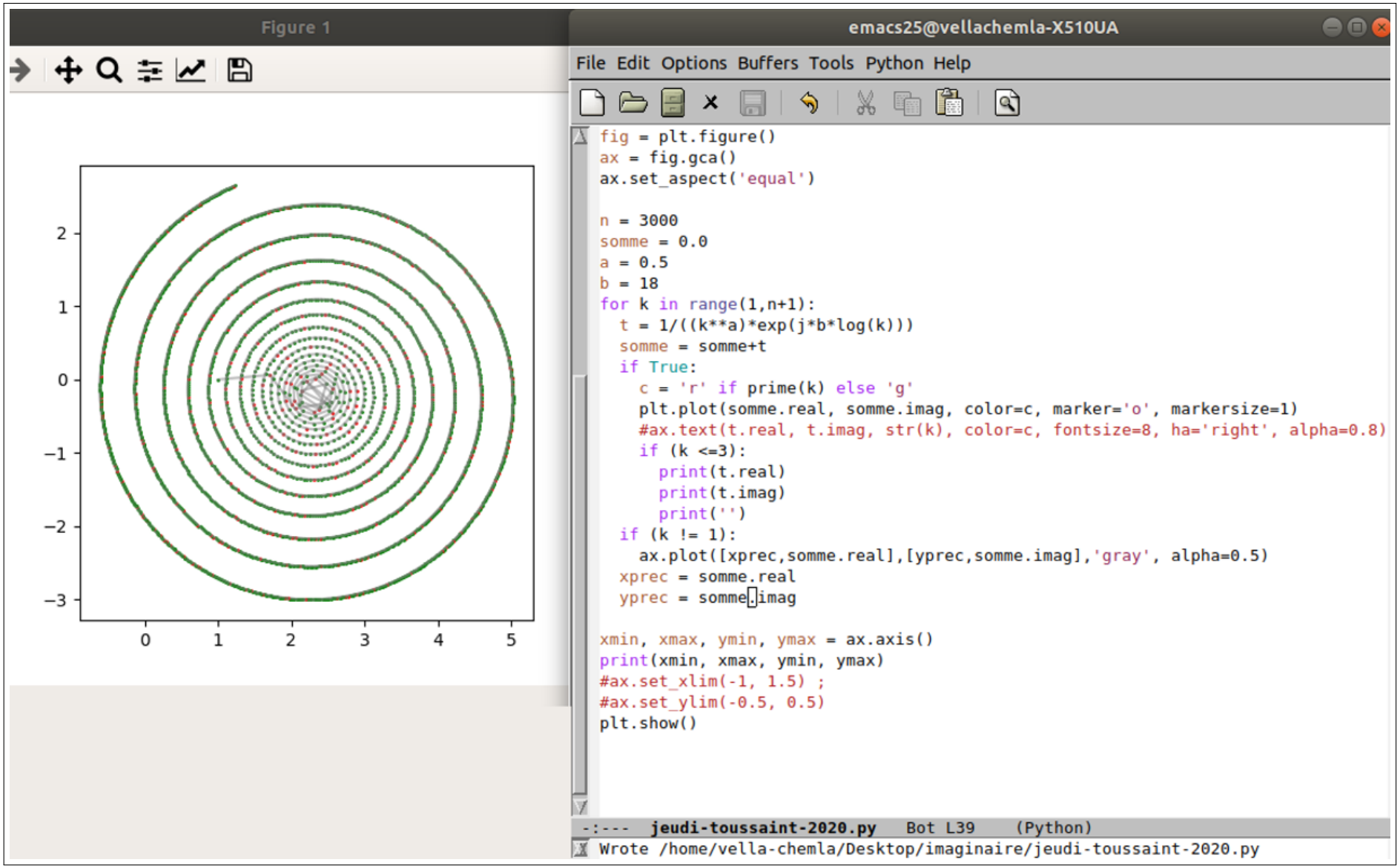
$n = 3000, a = 0.5, b = 14.134725141734693790.$



$n = 3000, a = 0.5, b = 18.$



$n = 3000, a = 0.5, b = 18$. On “plotte” la somme au lieu de “plotter” chacun de ses termes.



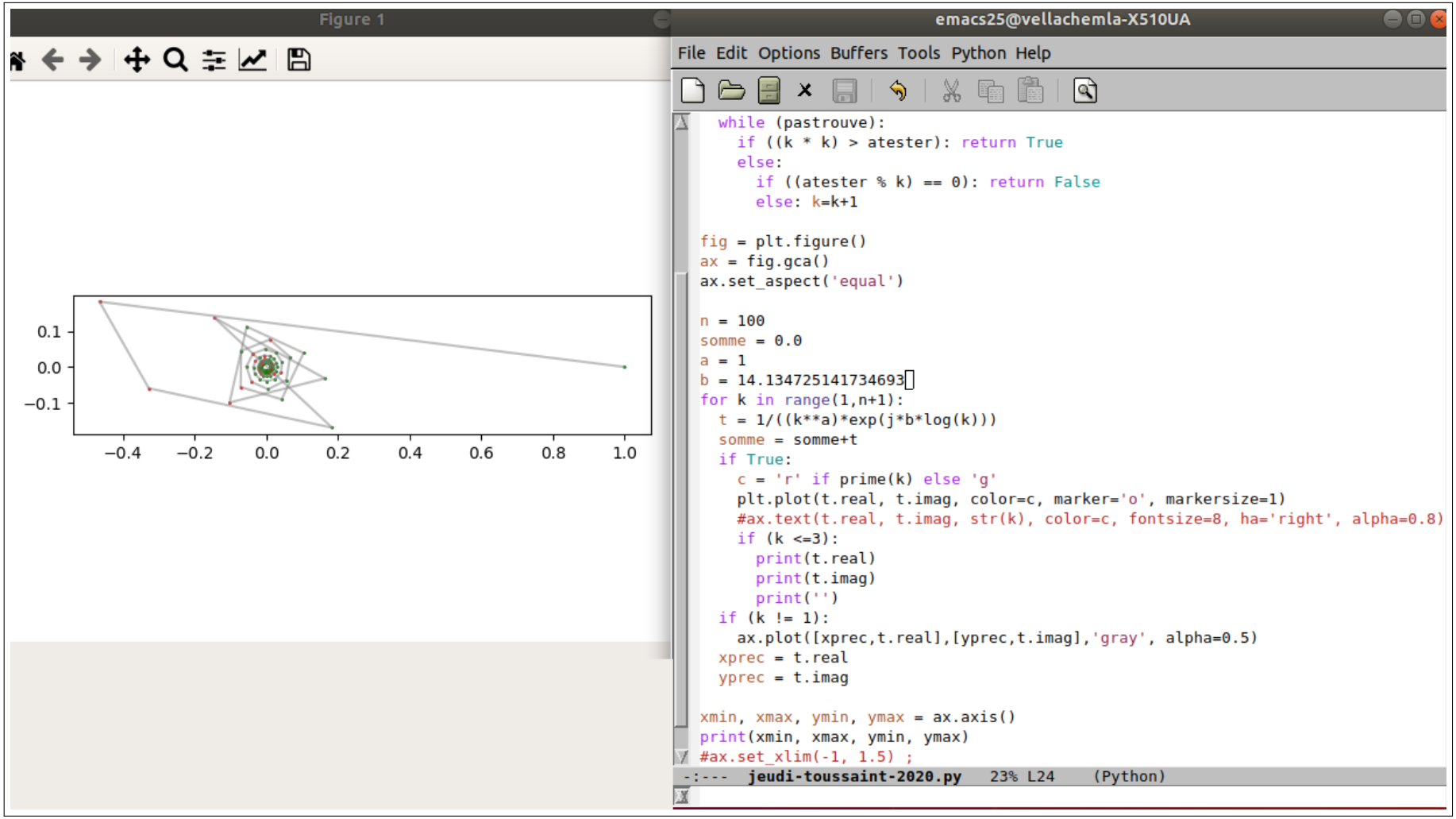


Figure 1
0.1
0.0
−0.1
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
somme = 0.0
a = 1
b = 14.134725141734693
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)

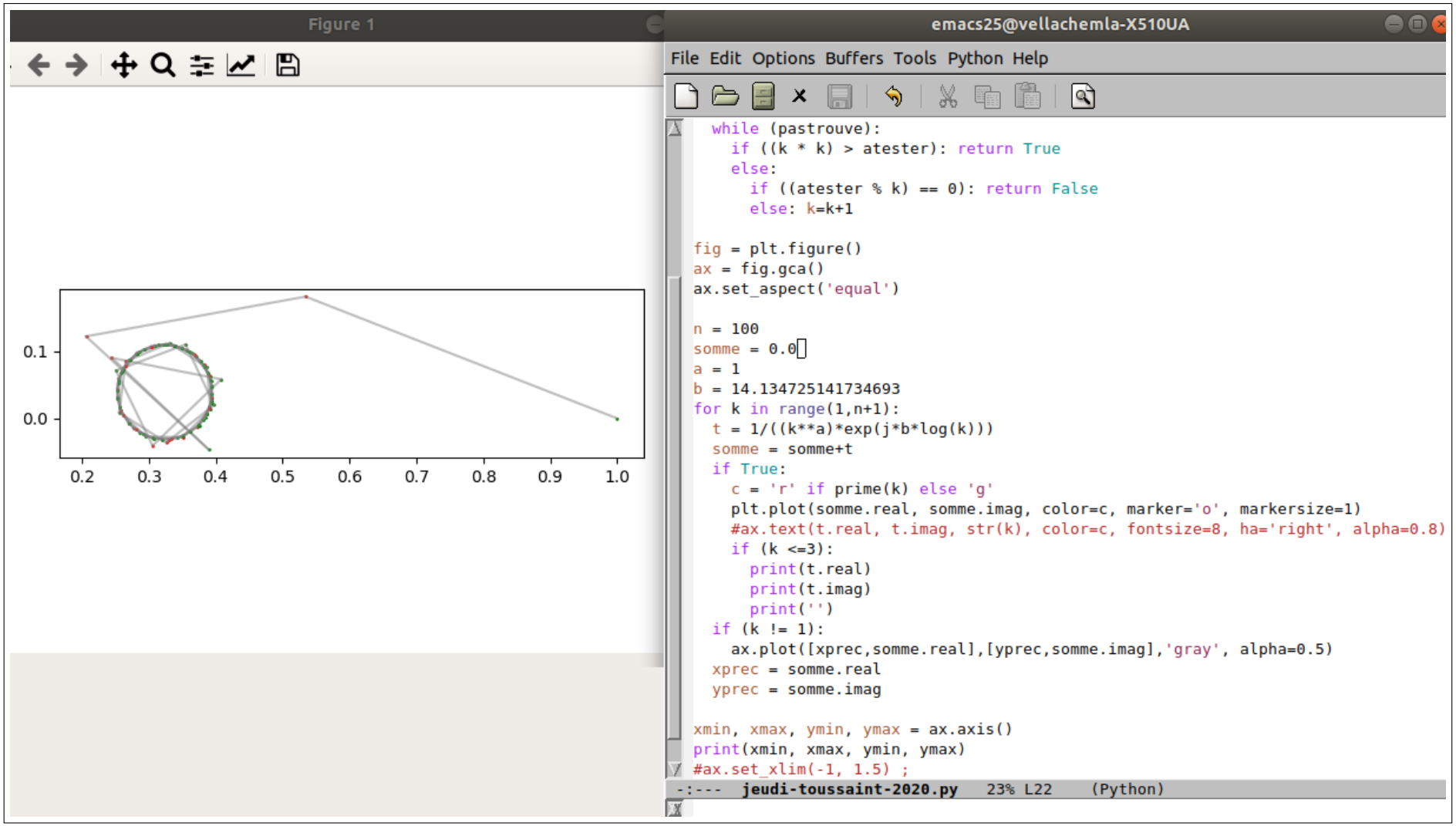


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
0.1
0.0
0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
somme = 0.0
a = 1
b = 14.134725141734693
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L22 (Python)

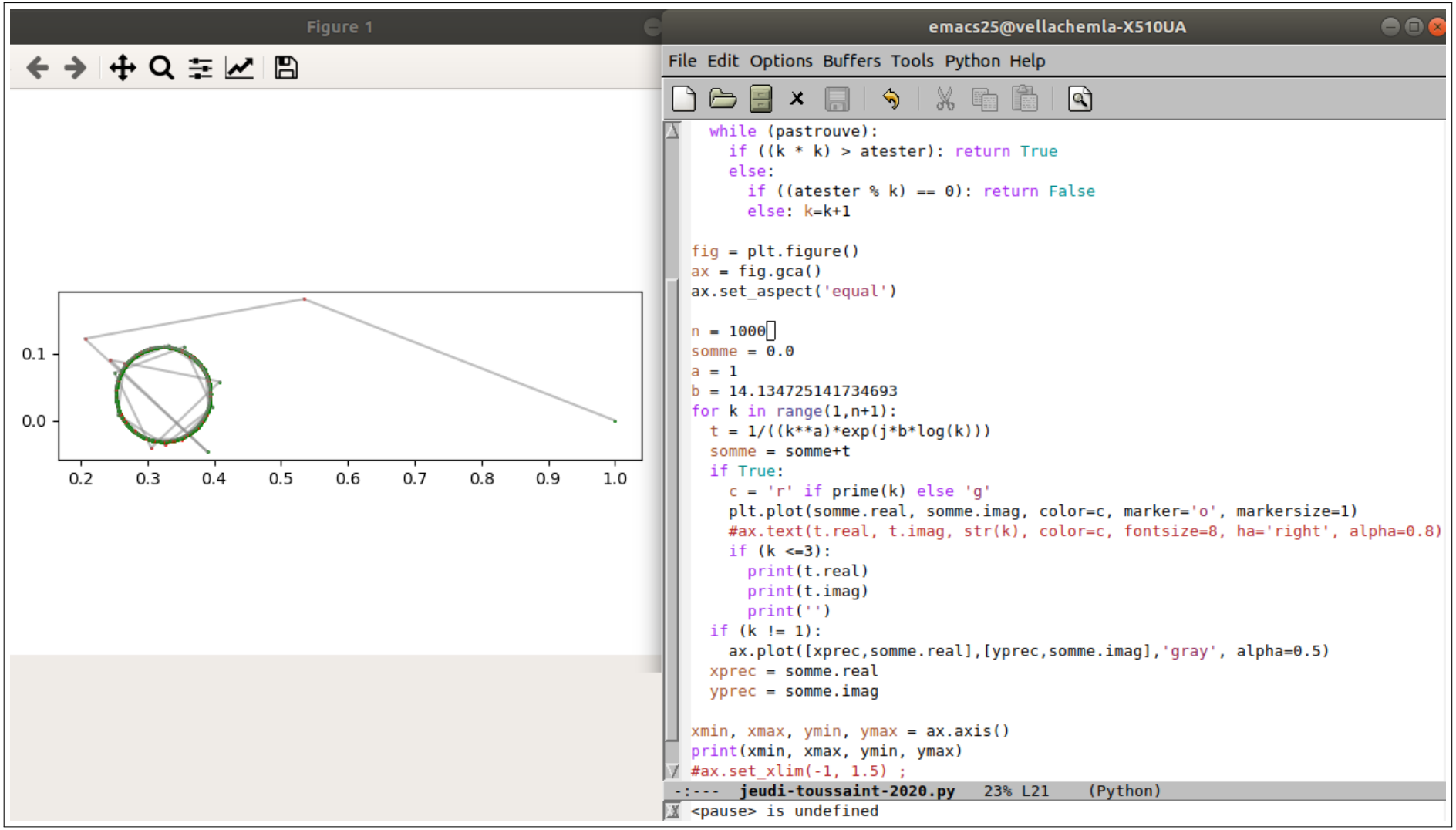


Figure 1
0.1
0.0
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 1
b = 14.134725141734693
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L21 (Python)
<pause> is undefined

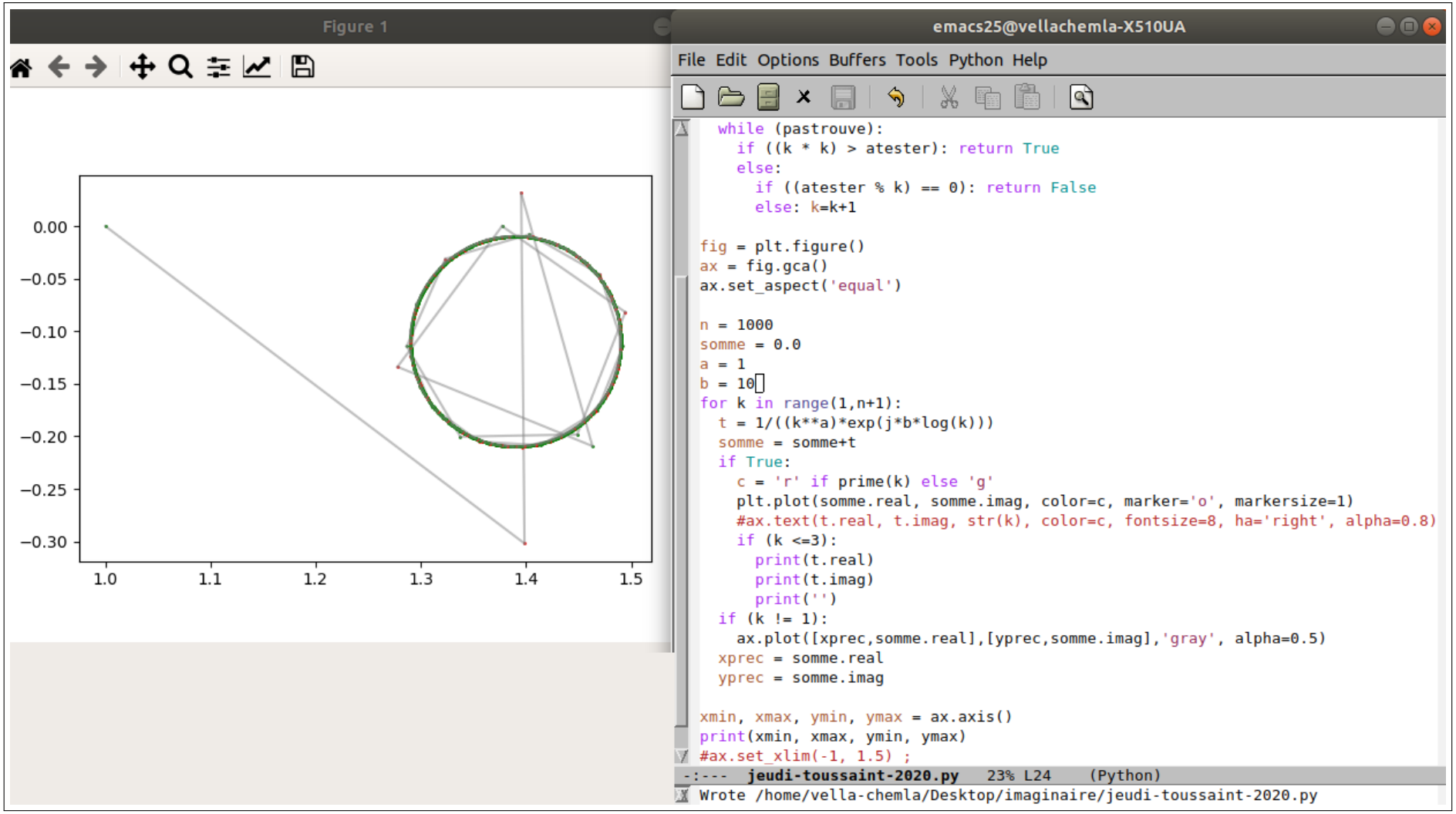


Figure 1
0.00
−0.05
−0.10
−0.15
−0.20
−0.25
−0.30
1.0
1.1
1.2
1.3
1.4
1.5
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 1
b = 10
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

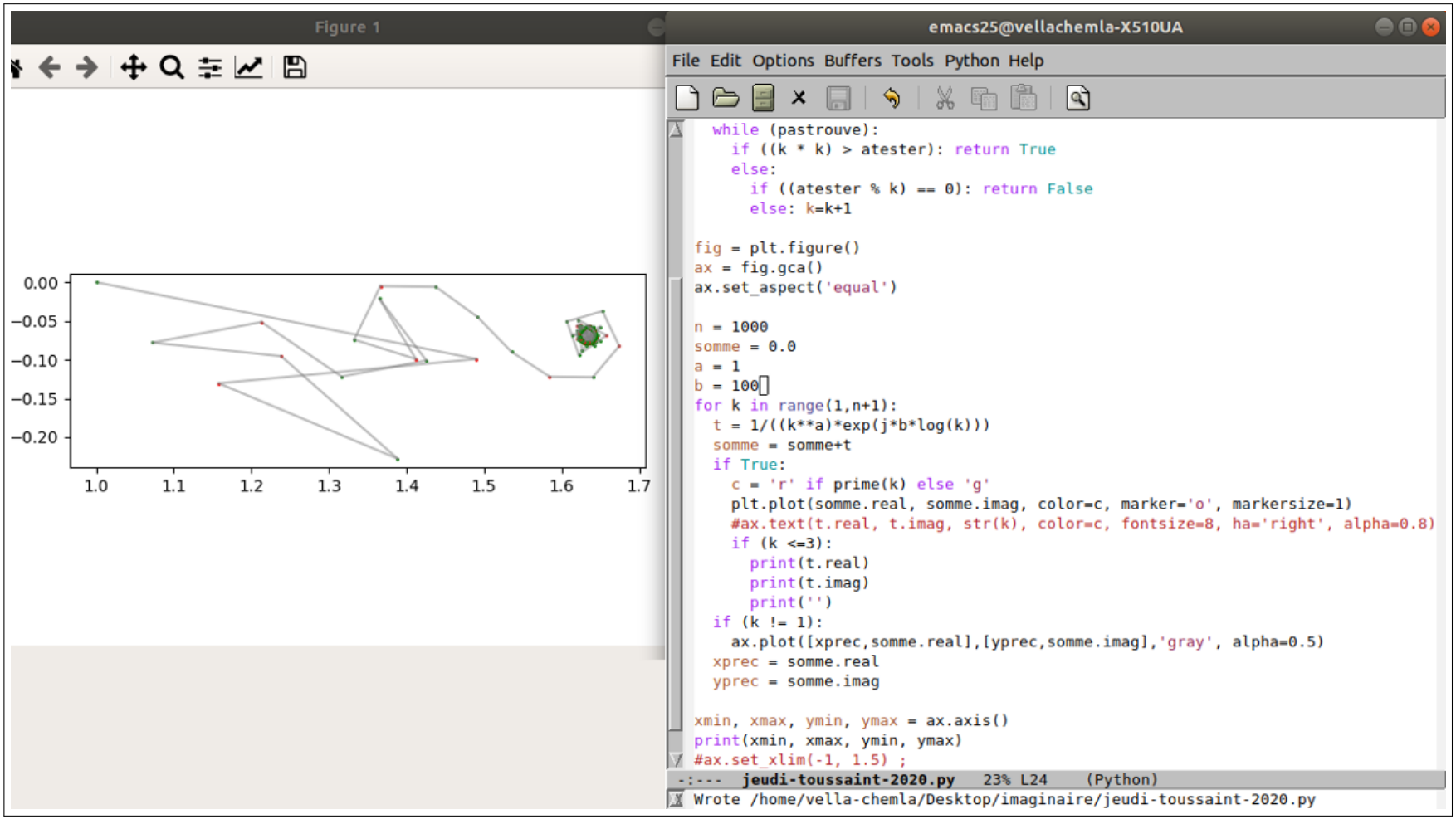


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
0.00
−0.05
−0.10
−0.15
−0.20
1.0
1.1
1.2
1.3
1.4
1.5
1.6
1.7
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 1
b = 100
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

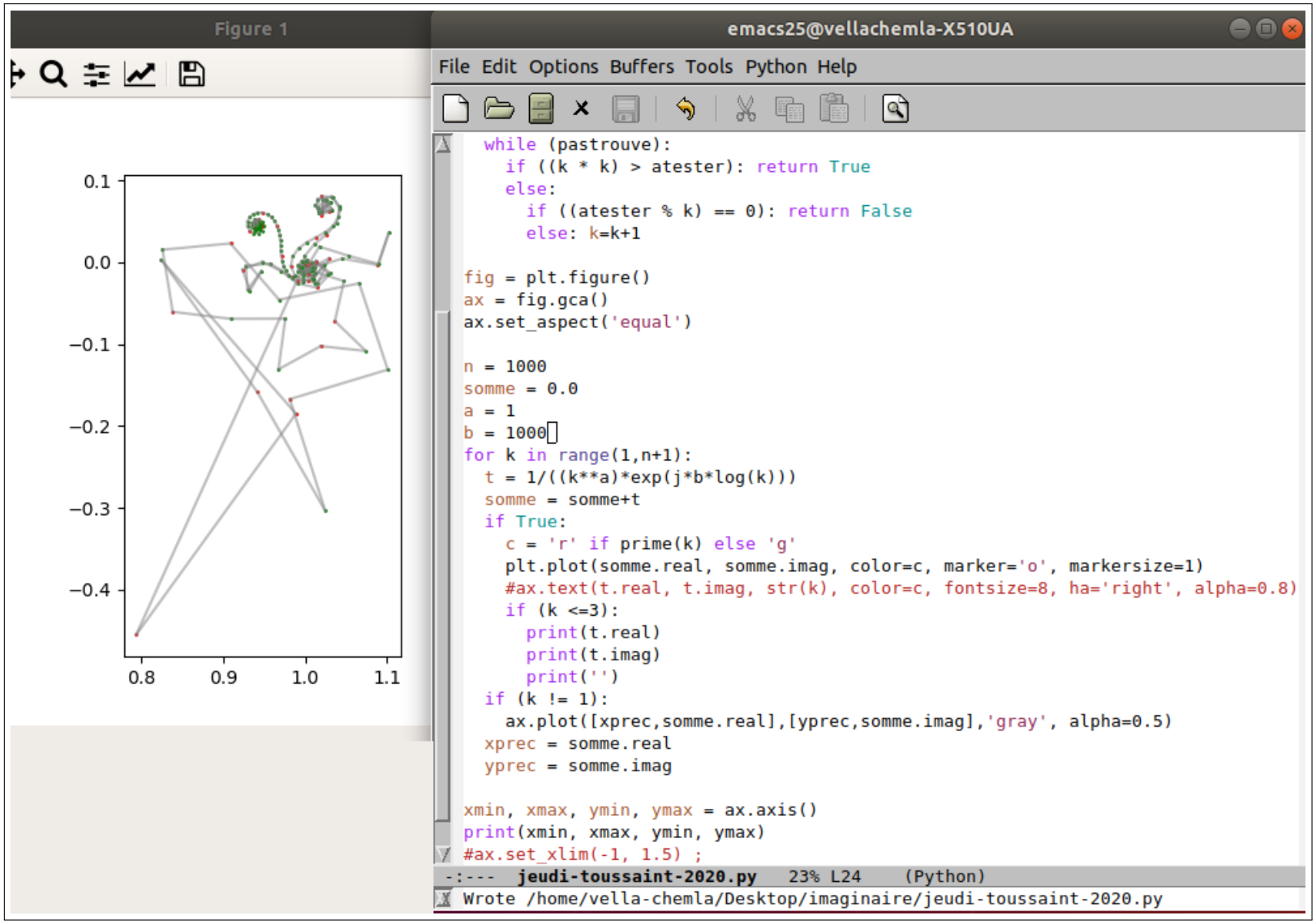


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
0.1
0.0
−0.1
−0.2
−0.3
−0.4
0.8
0.9
1.0
1.1
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 1
b = 1000
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

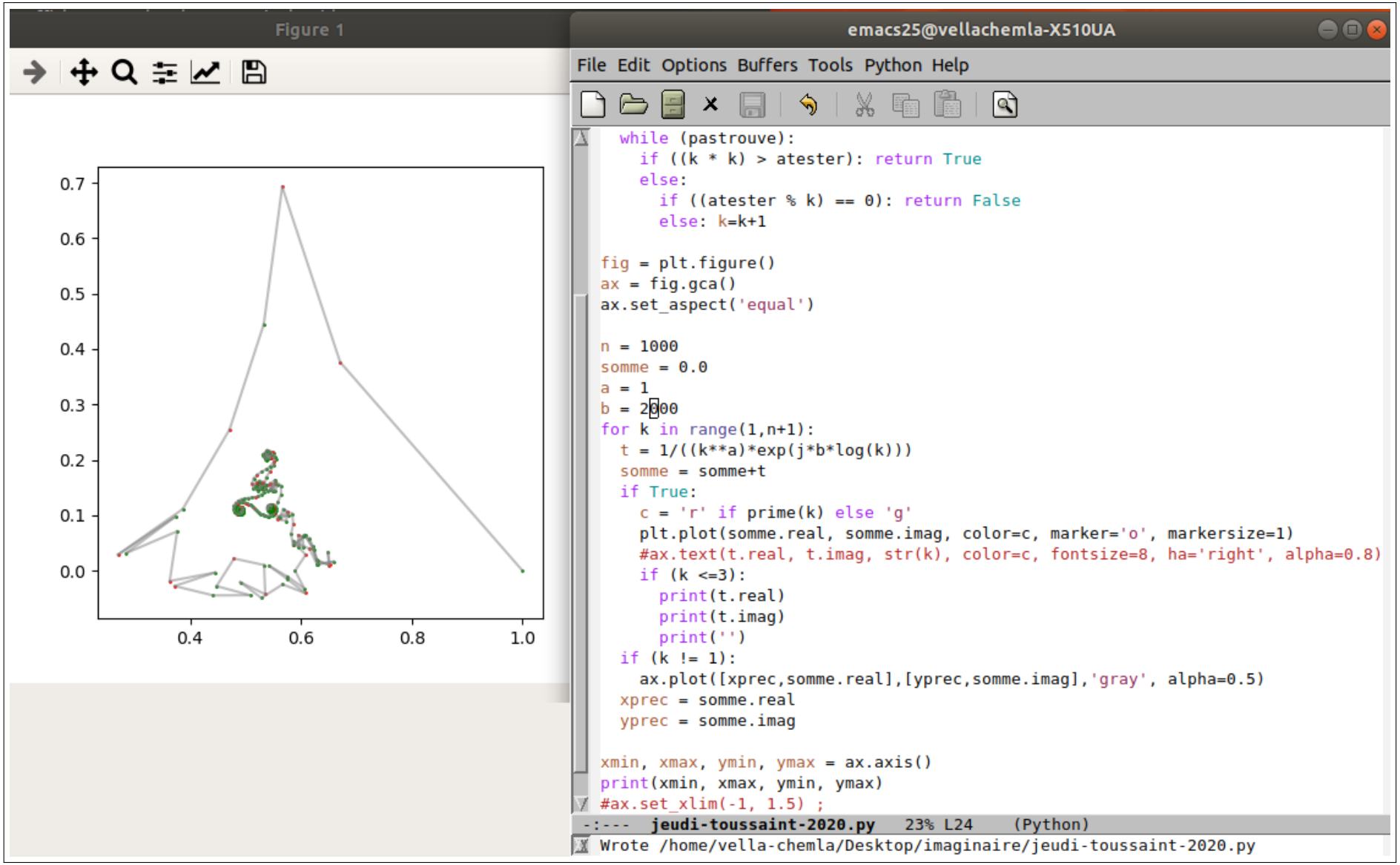


Figure 1
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0.4
0.6
0.8
1.0
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 1
b = 2000
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

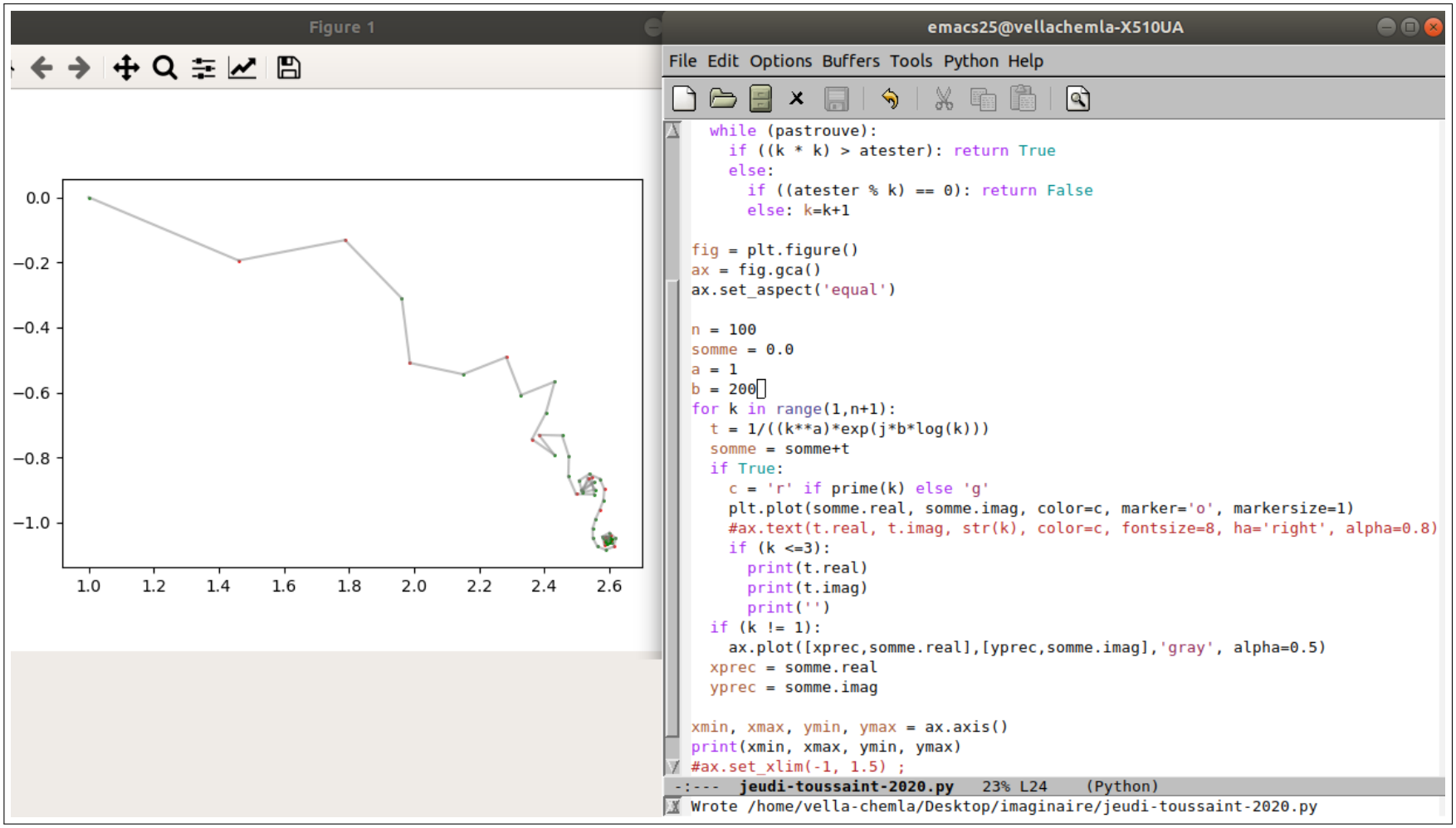


Figure 1
0.0
−0.2
−0.4
−0.6
−0.8
−1.0
1.0
1.2
1.4
1.6
1.8
2.0
2.2
2.4
2.6
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
somme = 0.0
a = 1
b = 200
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

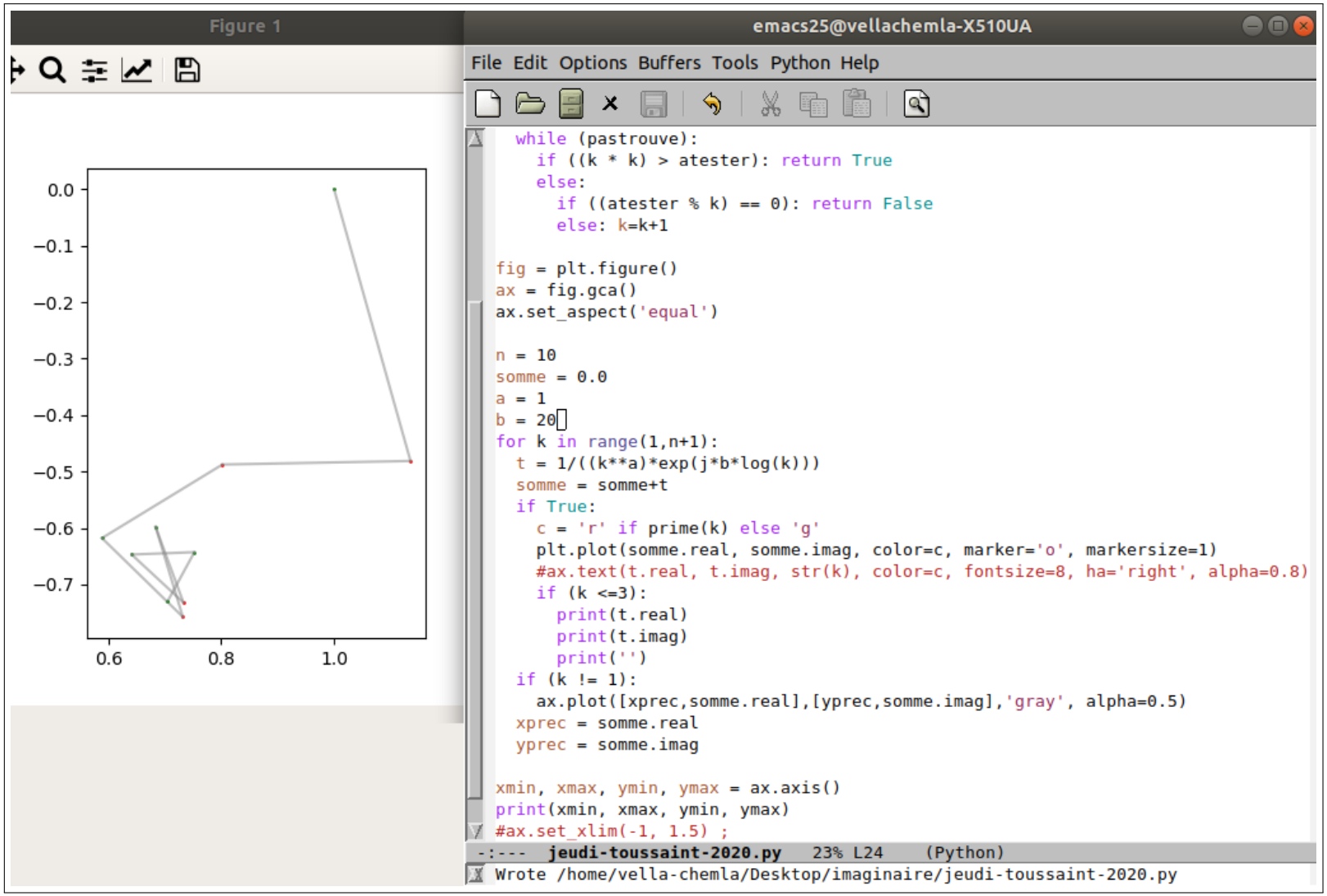


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
0.0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7
0.6
0.8
1.0
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 10
somme = 0.0
a = 1
b = 20
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

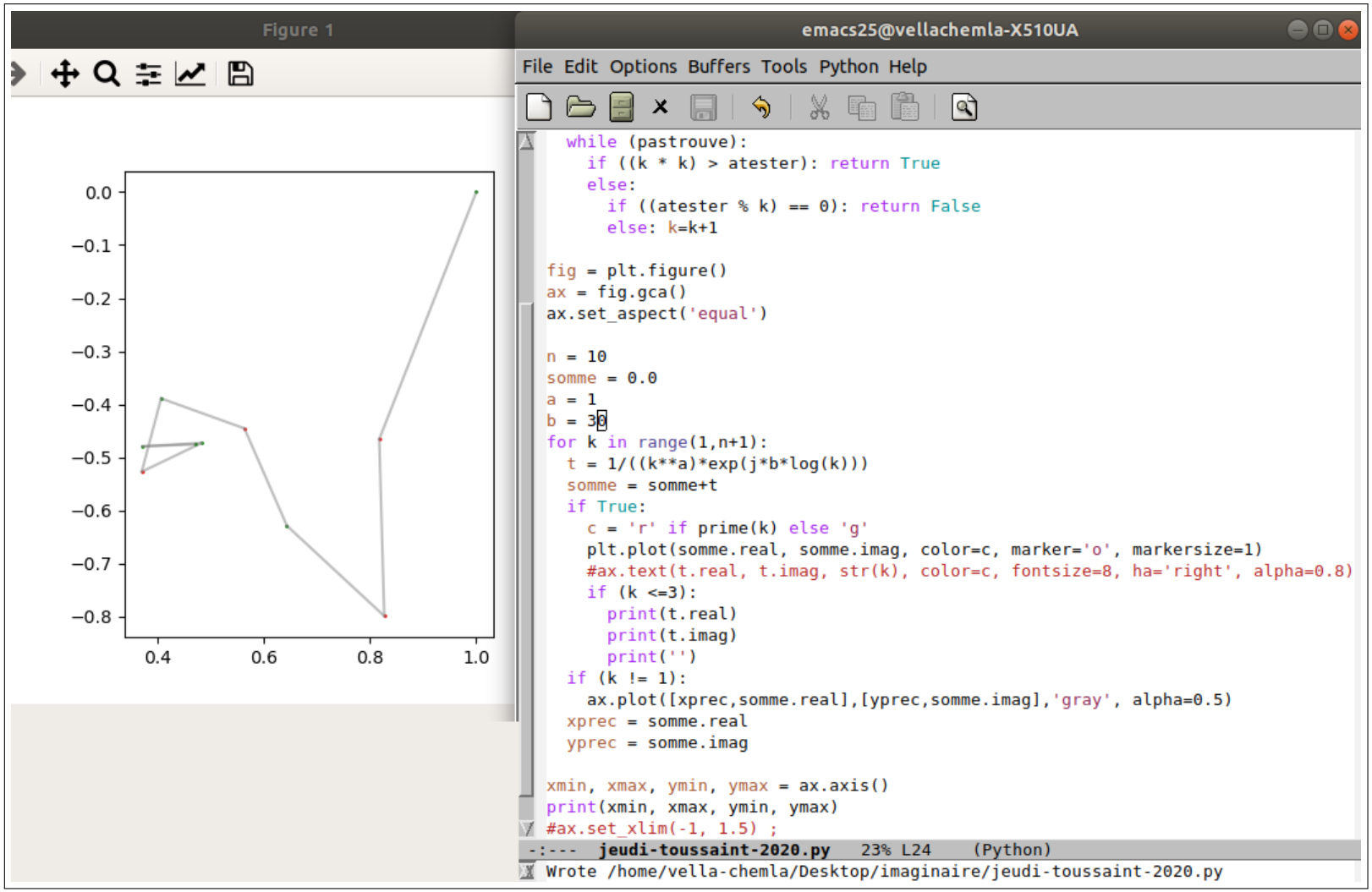


Figure 1
0.0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7
−0.8
0.4
0.6
0.8
1.0
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 10
somme = 0.0
a = 1
b = 30
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

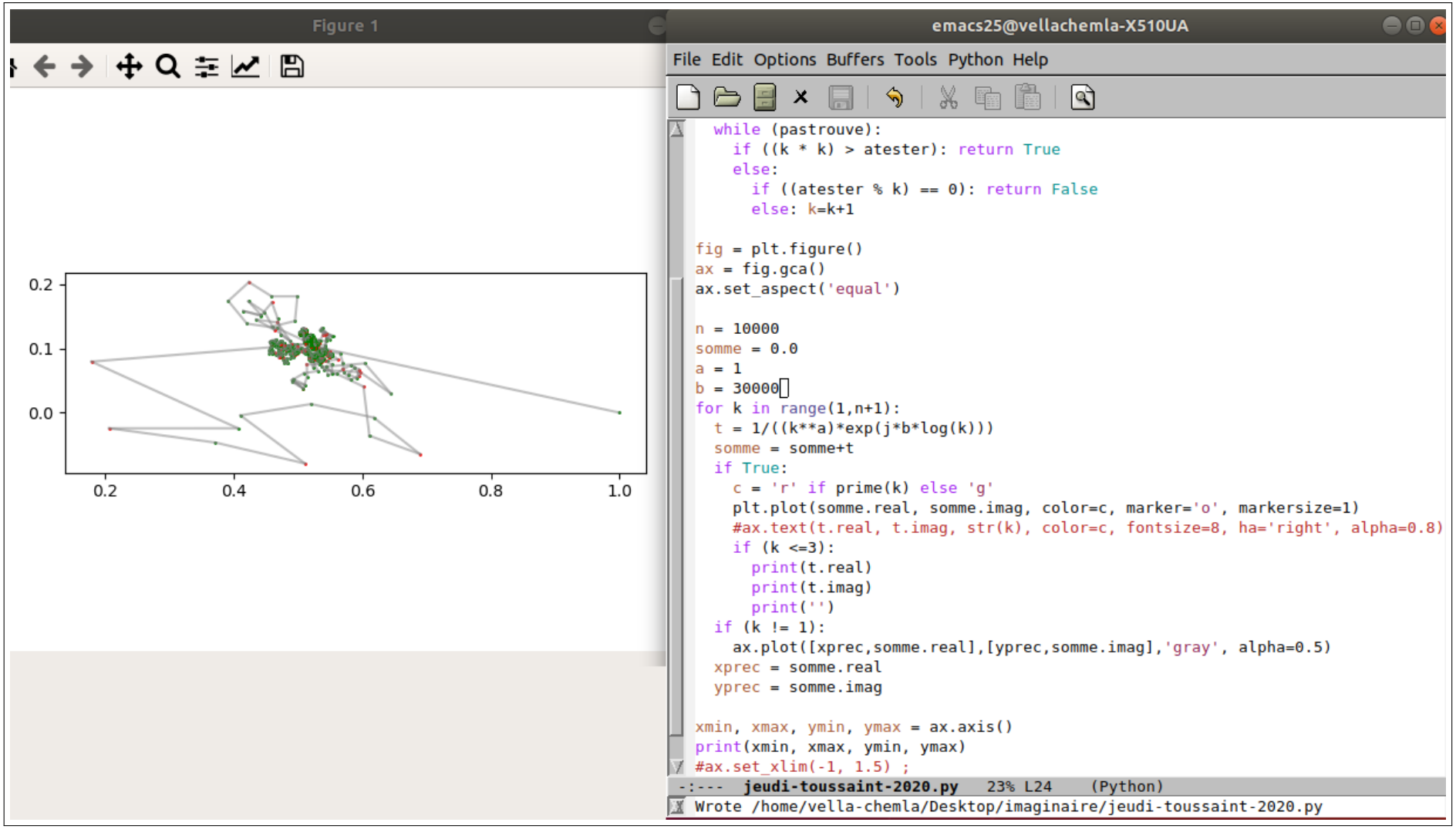


Figure 1
0.2
0.1
0.0
0.2
0.4
0.6
0.8
1.0
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 10000
somme = 0.0
a = 1
b = 30000
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

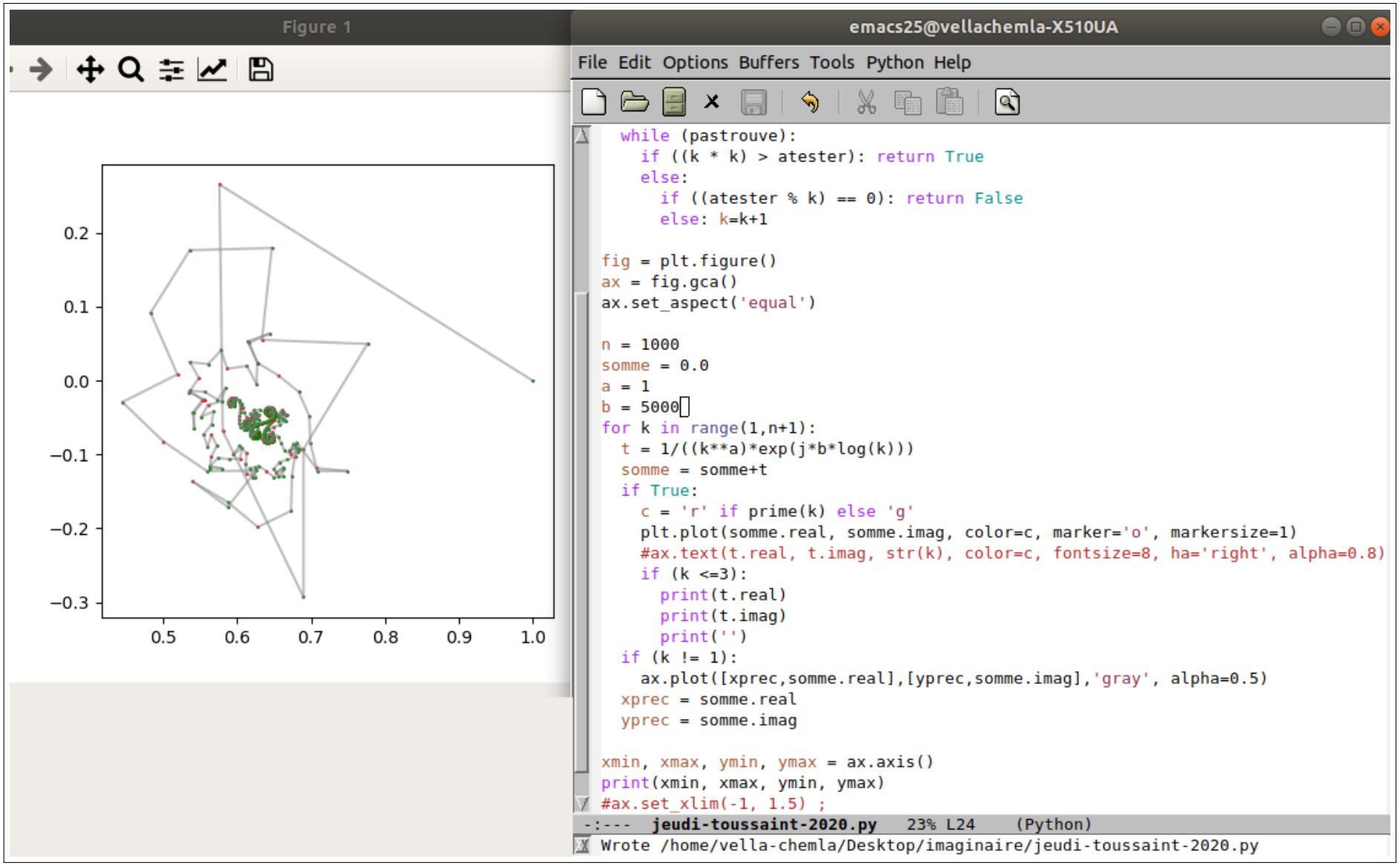


Figure 1
0.2
0.1
0.0
−0.1
−0.2
−0.3
0.5
0.6
0.7
0.8
0.9
1.0
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 1
b = 5000
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

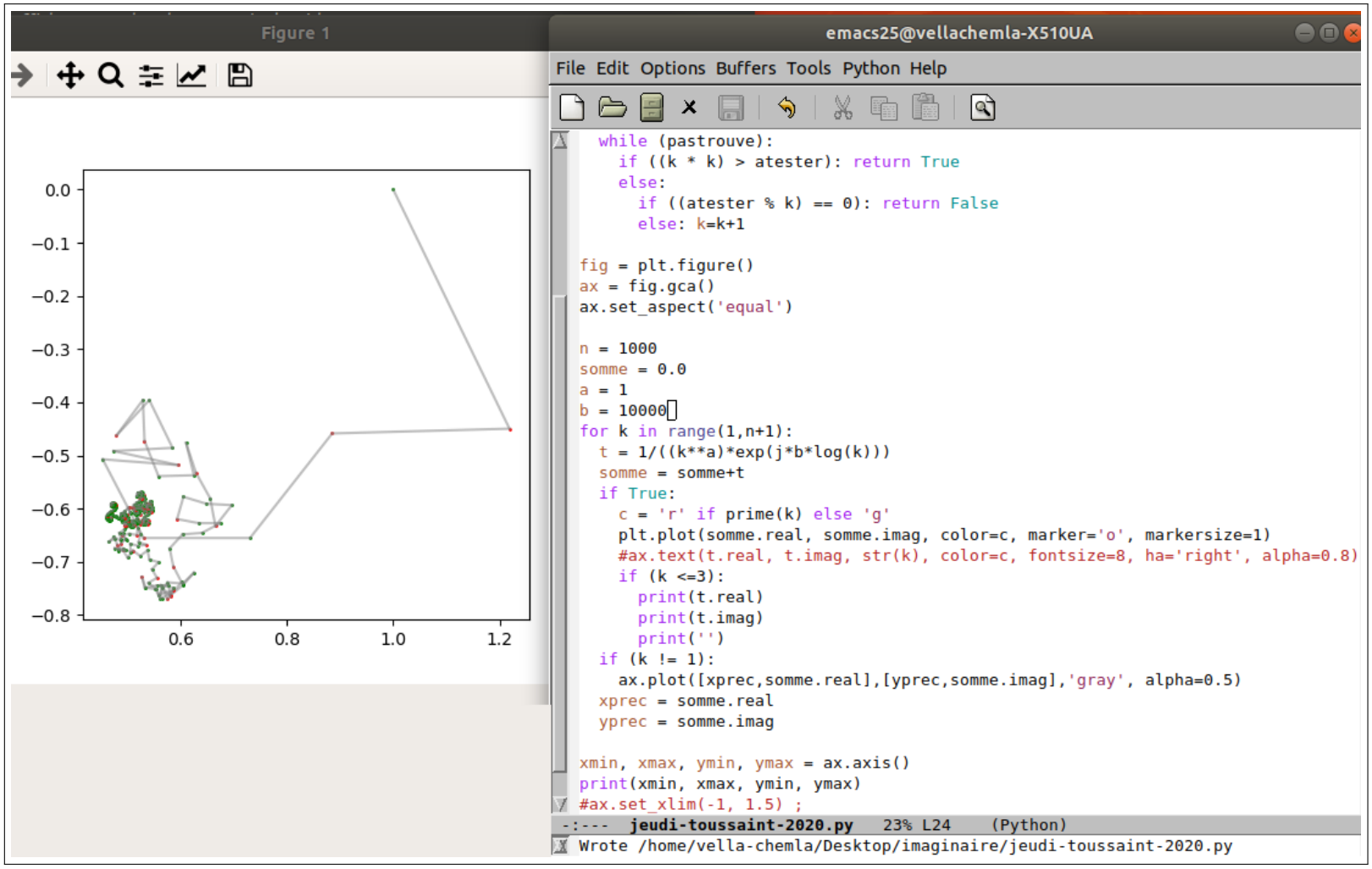


Figure 1
0.0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7
−0.8
0.6
0.8
1.0
1.2
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 1
b = 10000
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

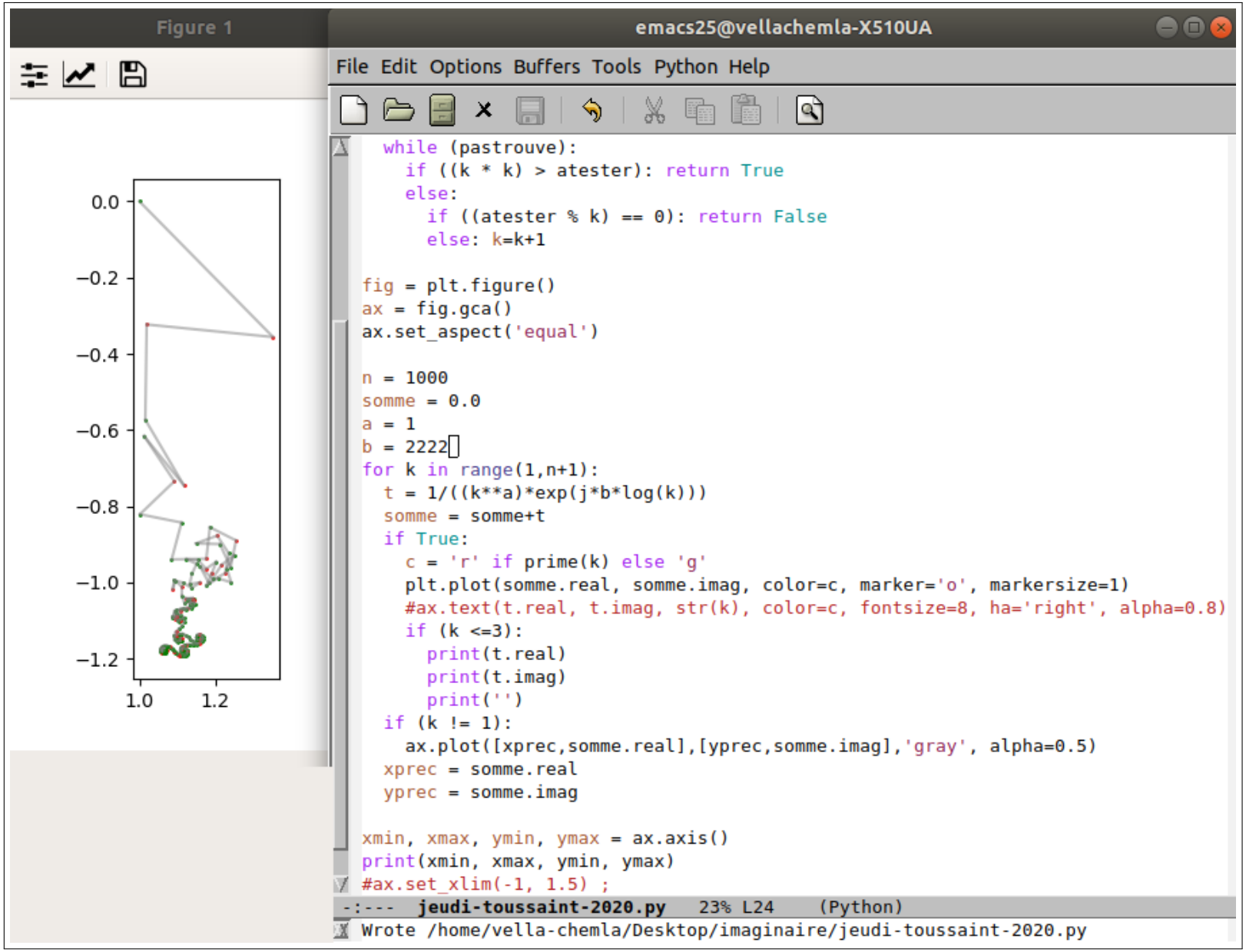


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
0.0
−0.2
−0.4
−0.6
−0.8
−1.0
−1.2
1.0
1.2
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 1
b = 2222
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

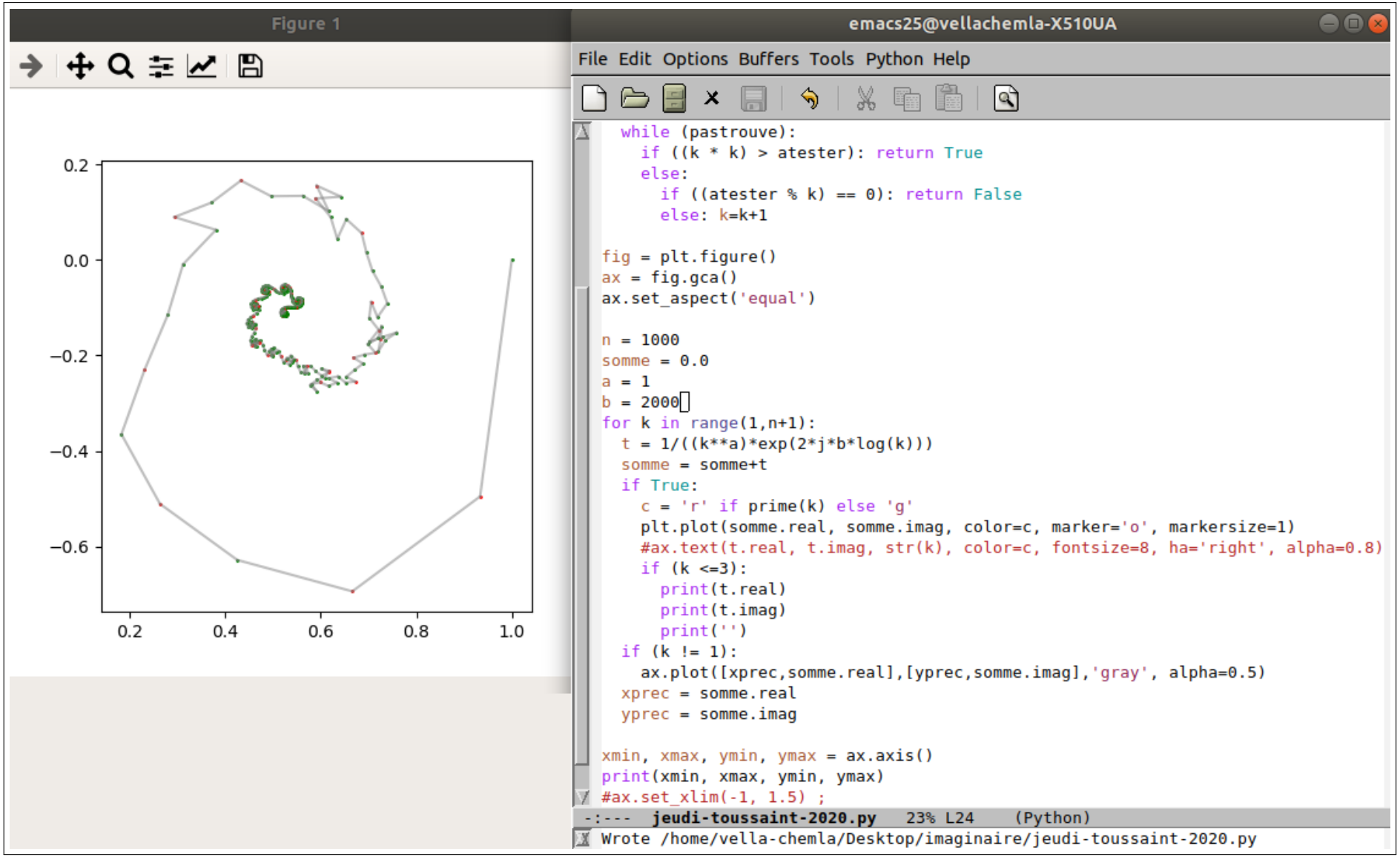


Figure 1
0.2
0.0
−0.2
−0.4
−0.6
0.2
0.4
0.6
0.8
1.0
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 1
b = 2000
for k in range(1,n+1):
t = 1/((k**a)*exp(2*j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

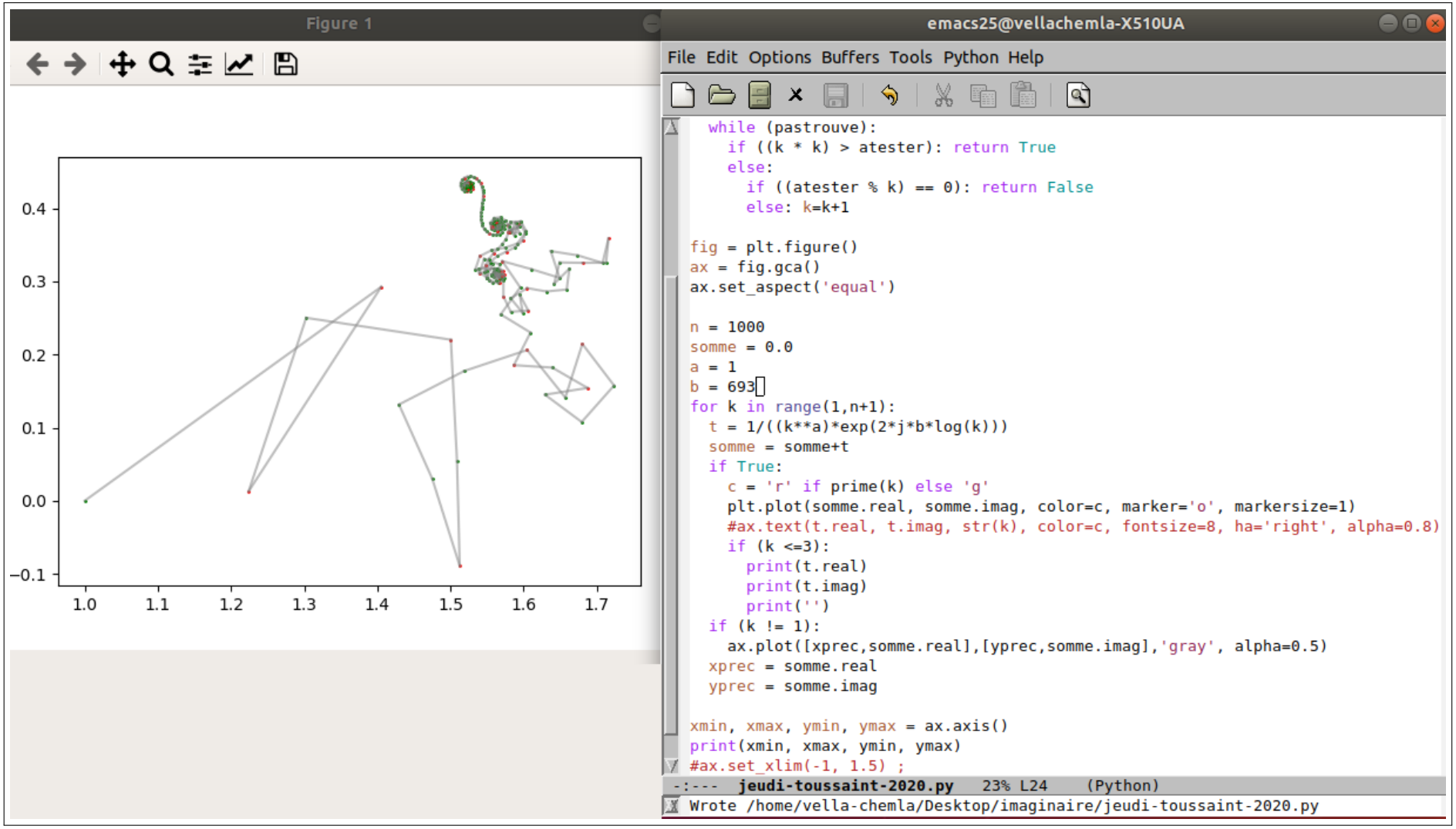


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
0.4
0.3
0.2
0.1
0.0
−0.1
1.0
1.1
1.2
1.3
1.4
1.5
1.6
1.7
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 1
b = 693
for k in range(1,n+1):
t = 1/((k**a)*exp(2*j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,somme.real],[yprec,somme.imag],'gray', alpha=0.5)
xprec = somme.real
yprec = somme.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

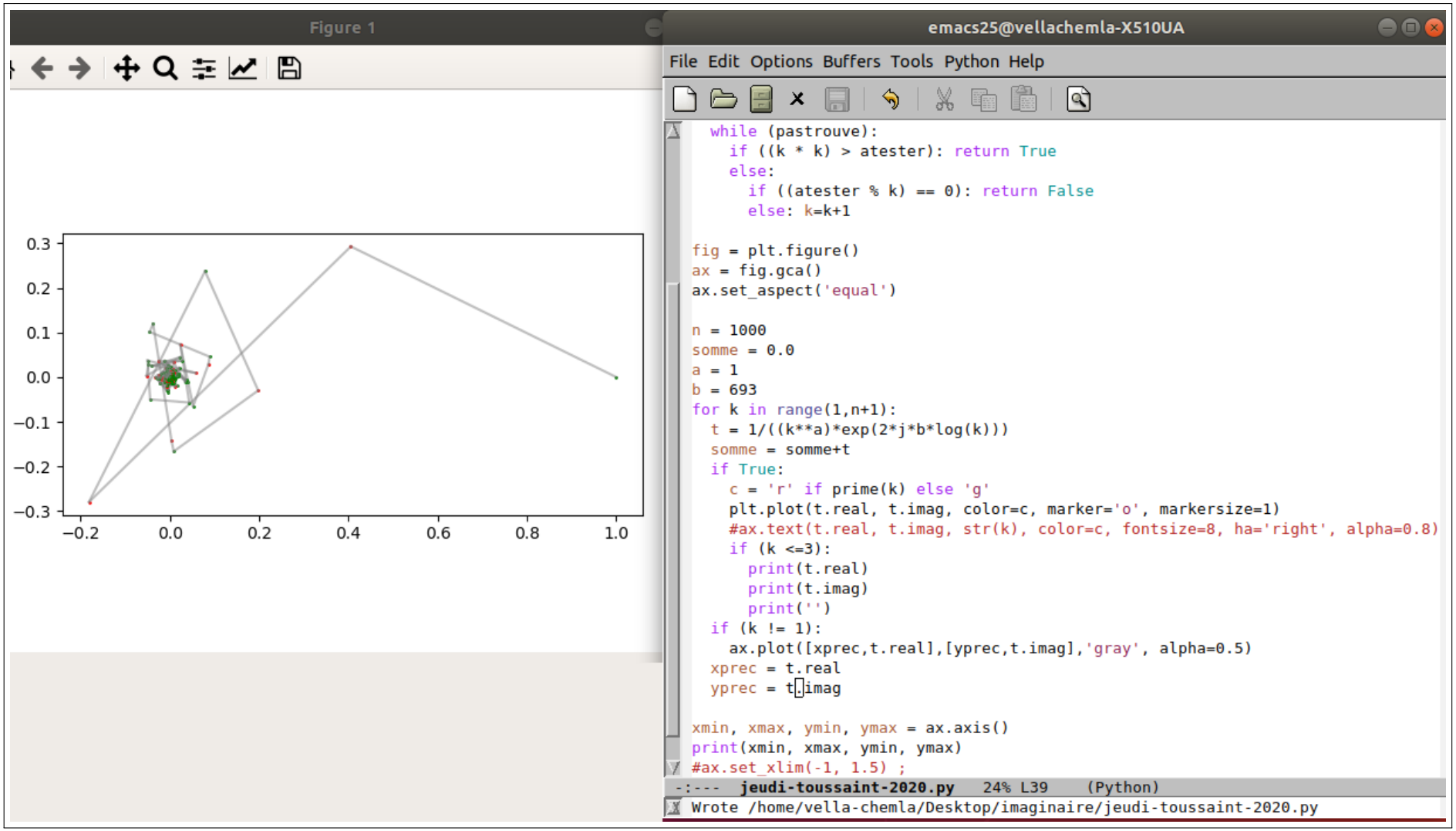


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 1
b = 693
for k in range(1,n+1):
t = 1/((k**a)*exp(2*j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
if (k != 1):
ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 24% L39 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

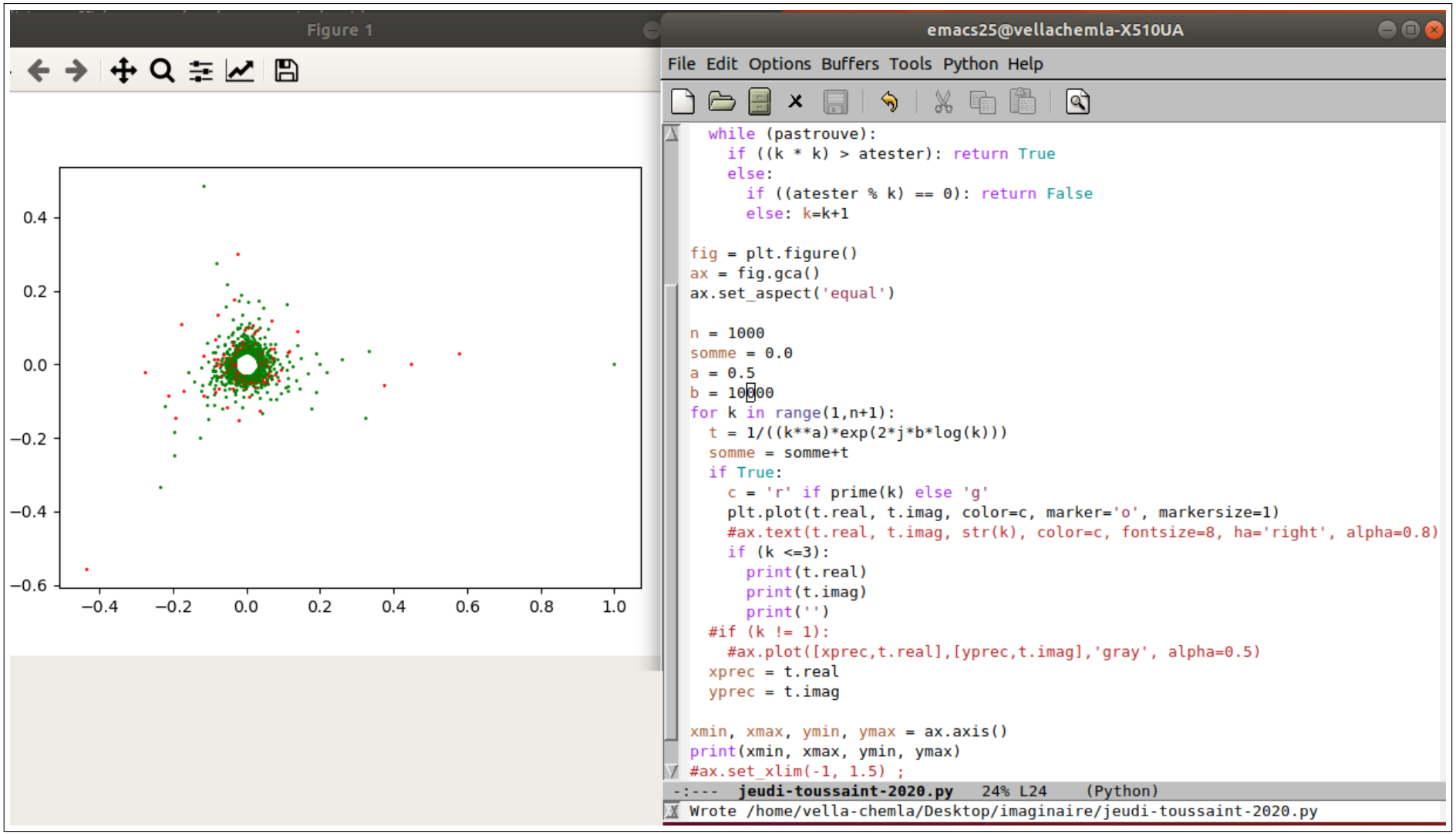


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 0.5
b = 10000
for k in range(1,n+1):
t = 1/((k**a)*exp(2*j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
#if (k != 1):
#ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 24% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
1.0

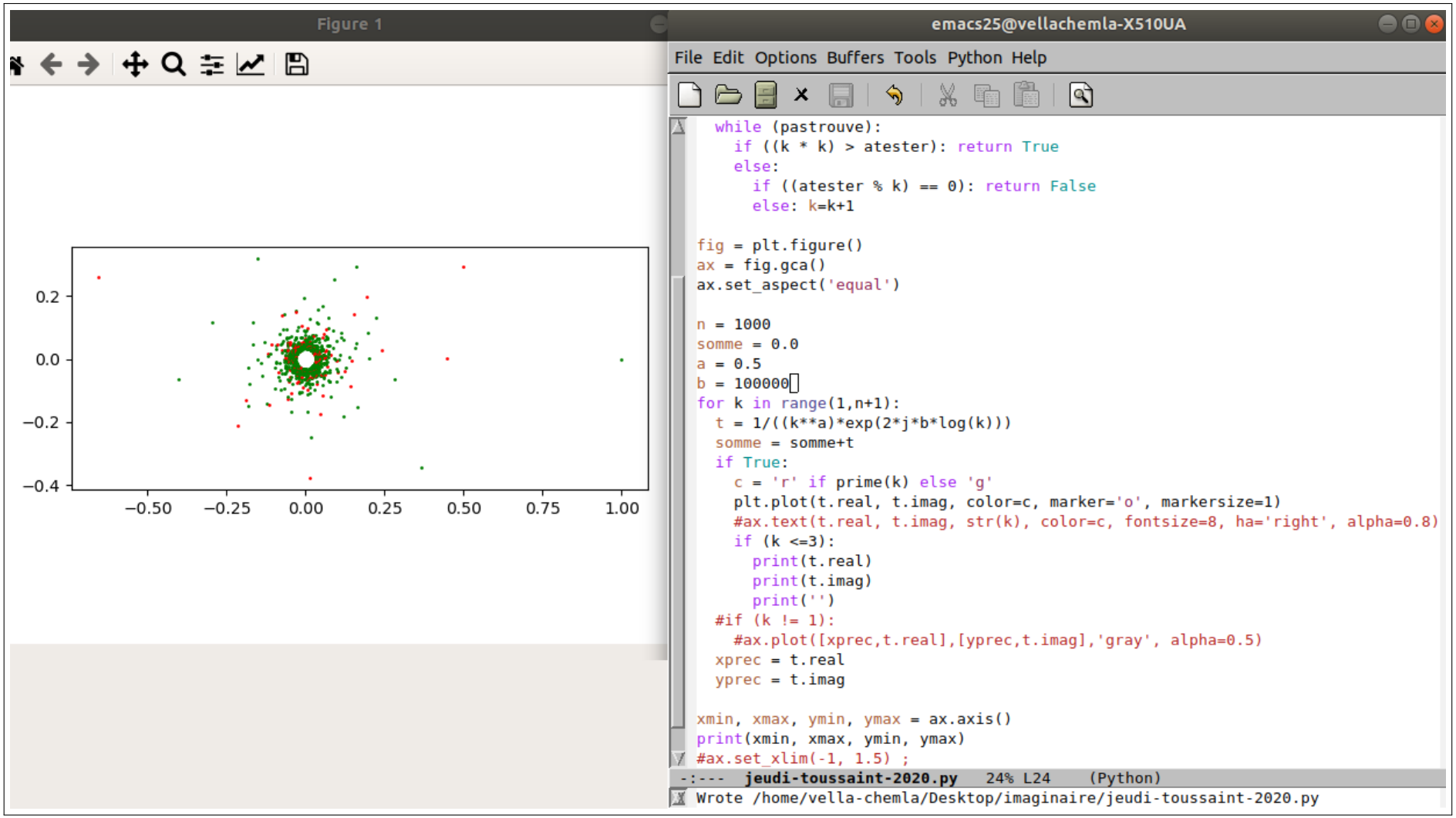


Figure 1
0.2
0.0
−0.2
−0.4
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 0.5
b = 100000
for k in range(1,n+1):
t = 1/((k**a)*exp(2*j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
#if (k != 1):
#ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 24% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

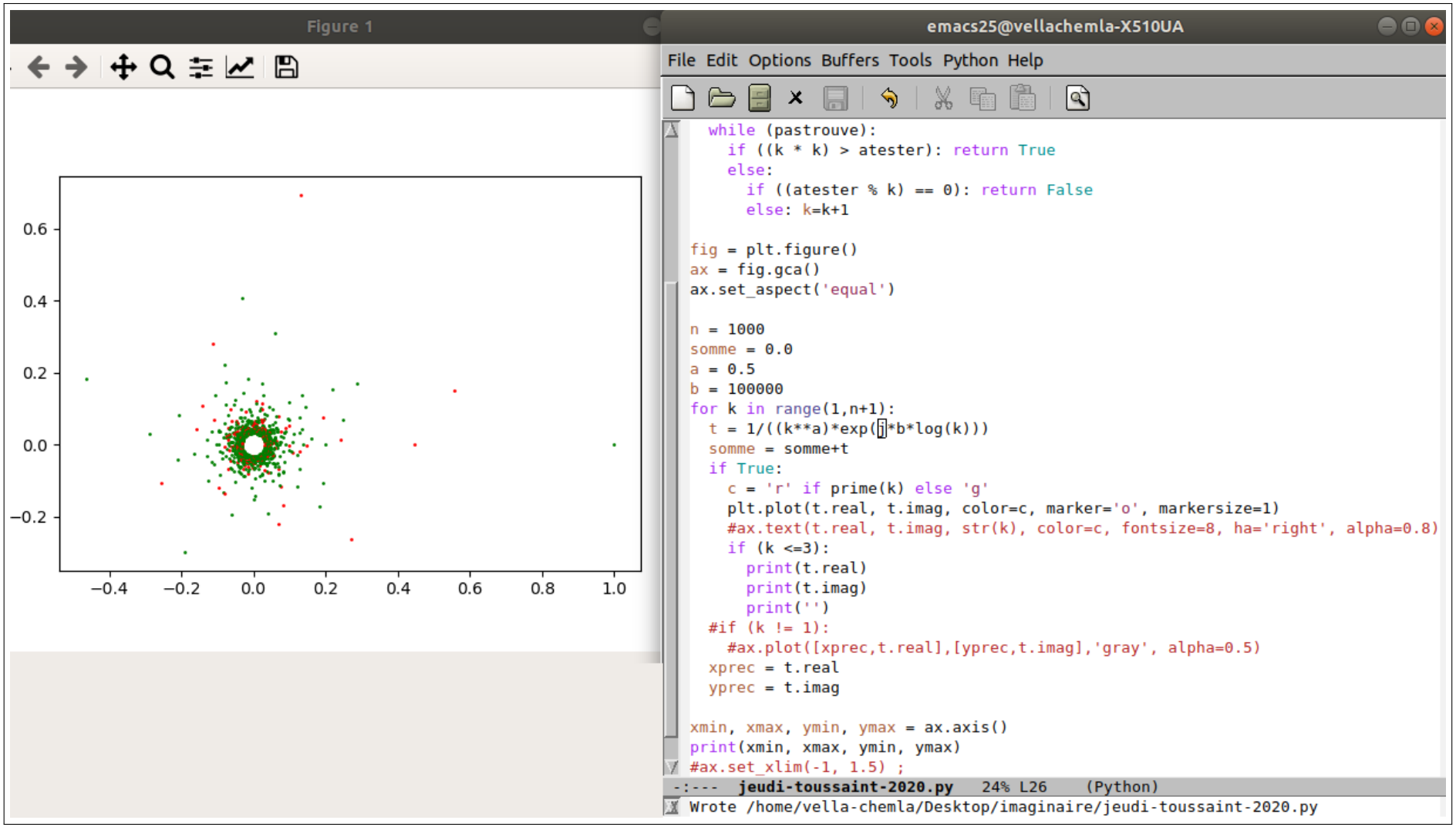


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 0.5
b = 100000
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
#if (k != 1):
#ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 24% L26 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

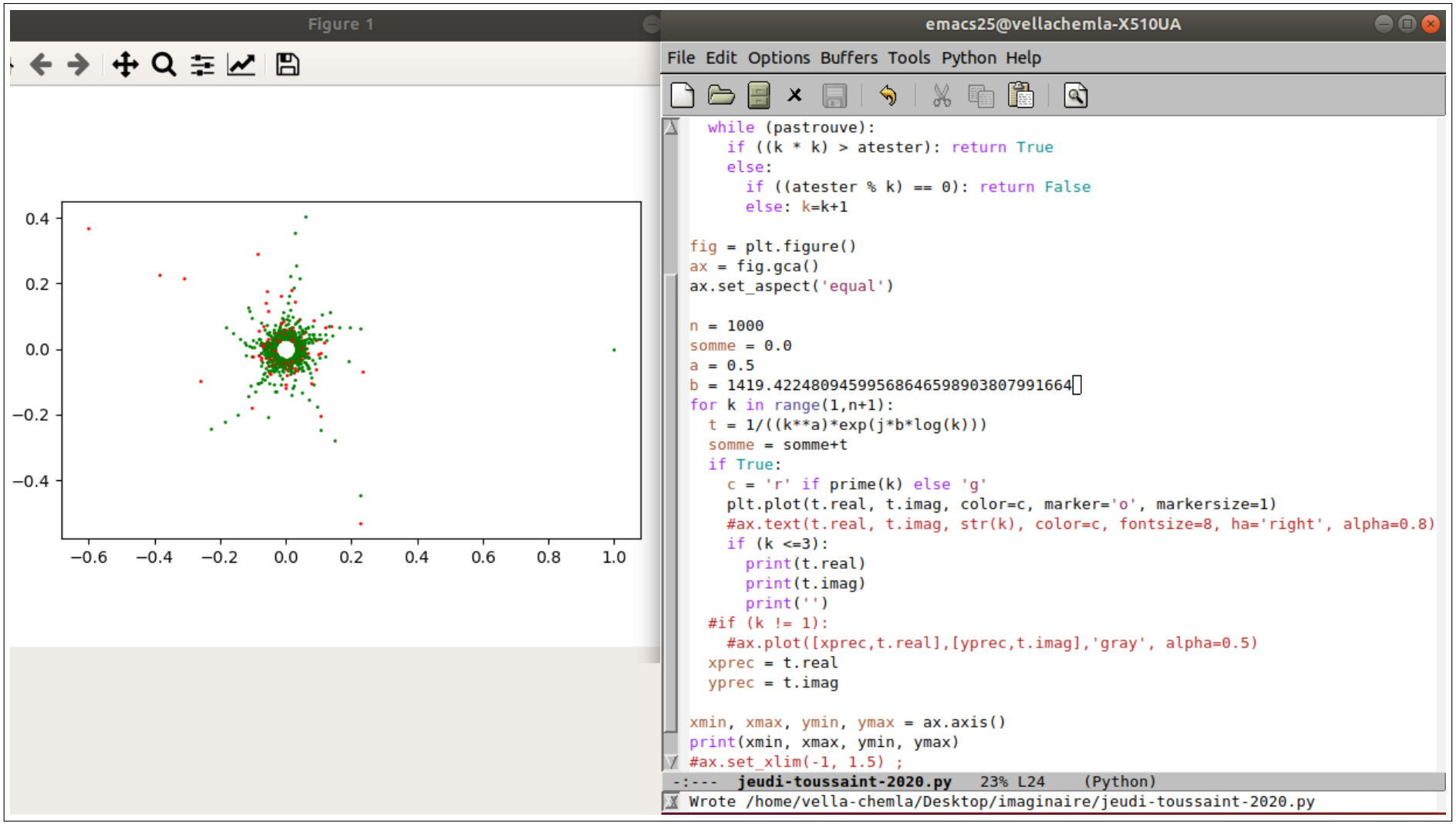


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
0.4
0.2
0.0
−0.2
−0.4
−0.6 −0.4 −0.2 0.0 0.2 0.4 0.6 0.8 1.0
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 0.5
b = 1419.42248094599568646598903807991664
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(t.real, t.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
#if (k != 1):
#ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L24 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

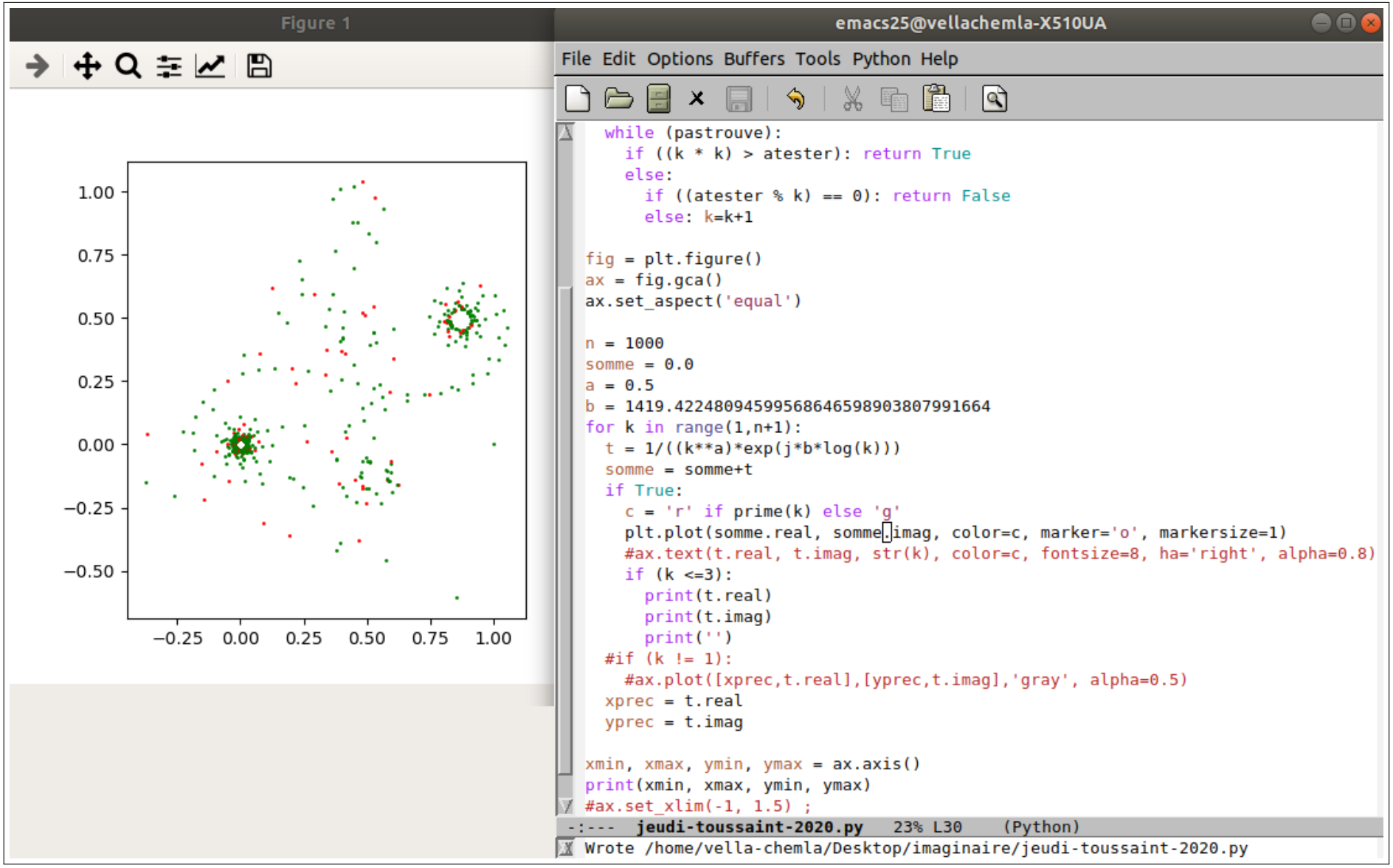


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.25 0.00 0.25 0.50 0.75 1.00
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 1000
somme = 0.0
a = 0.5
b = 1419.42248094599568646598903807991664
for k in range(1,n+1):
t = 1/((k**a)*exp(j*b*log(k)))
somme = somme+t
if True:
c = 'r' if prime(k) else 'g'
plt.plot(somme.real, somme.imag, color=c, marker='o', markersize=1)
#ax.text(t.real, t.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k <=3):
print(t.real)
print(t.imag)
print('')
#if (k != 1):
#ax.plot([xprec,t.real],[yprec,t.imag],'gray', alpha=0.5)
xprec = t.real
yprec = t.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
#ax.set_xlim(-1, 1.5) ;
-:--- jeudi-toussaint-2020.py 23% L30 (Python)
Wrote /home/vella-chemla/Desktop/imaginaire/jeudi-toussaint-2020.py

Différents programmes (Denise Vella-Chemla, 21.11.2020)

Cette note pour garder mémoire de divers programmes trouvés au gré de lectures.

1) En lisant littéralement l'expression de ζ et en l'imaginant géométriquement comme une spirale [1], aux complexes annulant ζ correspondraient des spirales dont les côtés successifs sont de longueur $\frac{1}{\sqrt{n}}$ pour n les entiers successifs. On a trouvé cette formule d'égalité à l'inverse d'une racine carrée :

$$\frac{1+2\psi(x)}{1+2\psi(\frac{1}{x})} = \frac{1}{\sqrt{x}}$$

avec $\psi(x)$ la fonction [2] définie par $\psi(x) = \sum_{n=1}^{\infty} e^{-n^2\pi x}$. Le programme de vérification rapide est :

```
import mpmath
from mpmath import jtheta, sqrt, exp, pi
from scipy import integrate

def sommeinfinie(x):
    somme = 0.0
    n = 1
    res = 1.0/mpmath.exp(n*n*mpmath.pi*x)
    while (res > 0.00000000000001):
        somme = somme+res
        n=n+1
        res = 1.0/exp(n*n*mpmath.pi*x)
    return somme

for x in range(1,100):
    print(str(x))
    print("1/racine de x  "+str(1.0/sqrt(float(x))))
    res = (1.0+2.0*sommeinfinie(x))/(1.0+2.0*sommeinfinie(1.0/float(x)))
    print("1+2 psi(x) / 1+2 psi(1/x) "+str(res))
    print('')
```

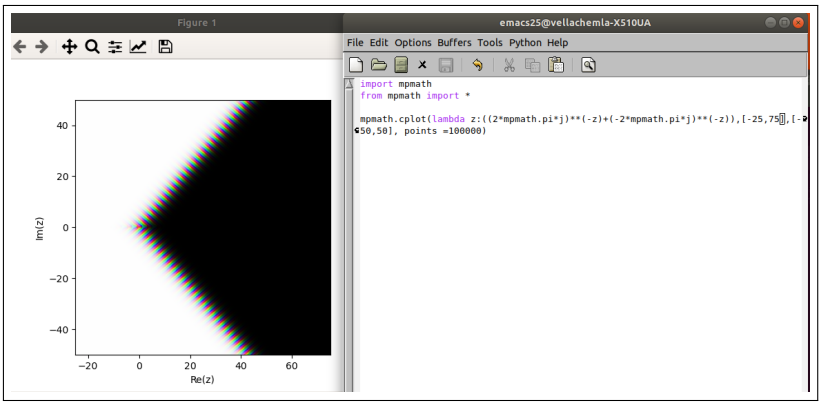
On modélise la convergence à l'infini comme on peut. Les résultats sont en fin de note.

2) On a trouvé dans un article récent de Peter Luschny cette expression au sujet de ζ :

$$\zeta(1-s) = \zeta(s)\tau(s)\Gamma(s)$$

avec $\tau(s) = (2\pi\, i)^{-s} + (-2\pi\, i)^{-s}$. Voici le programme et la visualisation de la fonction τ juste après :

```
import mpmath
from mpmath import *

mpmath.cplot(lambda z:((2*mpmath.pi*j)**(-z)+(-2*mpmath.pi*j)**(-z)),[-25,75],[-50,50], points =100000)
```

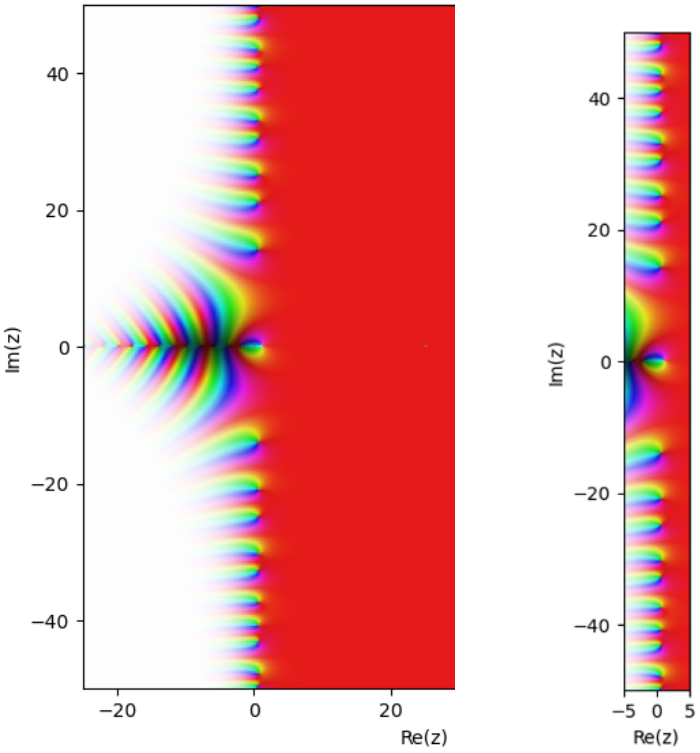


1. On avait pensé à ça dans http://denisevellachemla.eu/spirR.pdf.
2. Ce serait une fonction theta de Jacobi notée psi ?

On vérifie par ce programme

```
import mpmath
from mpmath import *

def tautt(z):
    return ((2*mpmath.pi*j)**(-z))+((-2*mpmath.pi*j)**(-z))

    mpmath.cplot(lambda z:mpmath.zeta(1-z)/(tautt(z)*mpmath.gamma(z)),[-25,75],[-50,50], points =100000)
```

qu'on obtient bien la visualisation de ζ "habituelle".



3) On peut également vérifier que la fonction en partie droite de la formule ci-dessous pour ζ fournit la même visualisation que ζ par le programme fourni.

$$\zeta(s) = \zeta(1-s)\Gamma(1-s)2^s\pi^{s-1}\sin\tfrac{1}{2}\pi\, s$$

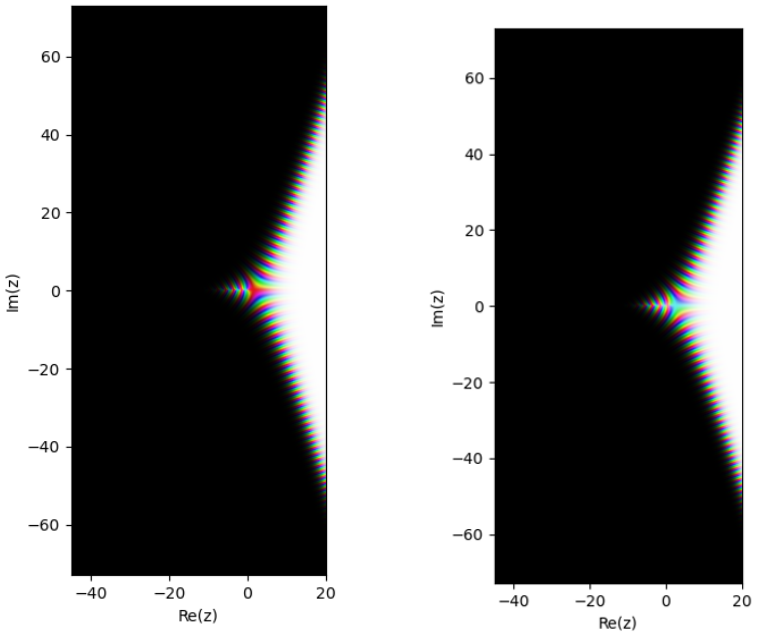
```
import mpmath
from mpmath import zeta,gamma, sin,pi

mpmath.cplot(lambda z:zeta(1-z)*gamma(1-z)*(2**z)*(pi**(z-1))*sin(0.5*pi*z),[-50,50],[-50,50], points = 100000)
```

4) On vérifie en python qu'une autre expression de Gamma visualise la fonction comme habituellement.

$$\Gamma(s) = 2^s \Gamma(\frac{s}{2})\Gamma(\frac{s-1}{2})\pi^{-\frac{1}{2}}$$

```
import mpmath
from mpmath import *
import math
from math import *

mpmath.cplot(lambda z:(2**z)*mpmath.gamma(z/2)*mpmath.gamma((z-1)/2)*(pi*(-0.5)),[-45,20],[-73,73], points =202000)
```

On n'obtient cependant pas les mêmes couleurs, ci-dessous.



Annexe : calcul de l'inverse de la racine en utilisant une fonction theta de Jacobi

1
1/racine de x 1.0
1+2 psi(x) / 1+2 psi(1/x) 1.0

2
1/racine de x 0.707106781186547
1+2 psi(x) / 1+2 psi(1/x) 0.707106781186547

3
1/racine de x 0.577350269189626
1+2 psi(x) / 1+2 psi(1/x) 0.577350269189626

4
1/racine de x 0.5
1+2 psi(x) / 1+2 psi(1/x) 0.5

5
1/racine de x 0.447213595499958
1+2 psi(x) / 1+2 psi(1/x) 0.447213595499958

6
1/racine de x 0.408248290463863
1+2 psi(x) / 1+2 psi(1/x) 0.408248290463864

7
1/racine de x 0.377964473009227
1+2 psi(x) / 1+2 psi(1/x) 0.377964473009227

8
1/racine de x 0.353553390593274
1+2 psi(x) / 1+2 psi(1/x) 0.353553390593274

9
1/racine de x 0.333333333333333
1+2 psi(x) / 1+2 psi(1/x) 0.333333333333333

10
1/racine de x 0.316227766016838
1+2 psi(x) / 1+2 psi(1/x) 0.316227766016838

11
1/racine de x 0.301511344577764
1+2 psi(x) / 1+2 psi(1/x) 0.301511344577763

12
1/racine de x 0.288675134594813
1+2 psi(x) / 1+2 psi(1/x) 0.288675134594813

13
1/racine de x 0.277350098112615
1+2 psi(x) / 1+2 psi(1/x) 0.277350098112615

14
1/racine de x 0.267261241912424
1+2 psi(x) / 1+2 psi(1/x) 0.267261241912426

15
1/racine de x 0.258198889747161
1+2 psi(x) / 1+2 psi(1/x) 0.258198889747161

16
1/racine de x 0.25
1+2 psi(x) / 1+2 psi(1/x) 0.25

17
1/racine de x 0.242535625036333
1+2 psi(x) / 1+2 psi(1/x) 0.242535625036333

18
1/racine de x 0.235702260395516
1+2 psi(x) / 1+2 psi(1/x) 0.235702260395516

19
1/racine de x 0.229415733870562
1+2 psi(x) / 1+2 psi(1/x) 0.229415733870563

20
1/racine de x 0.223606797749979
1+2 psi(x) / 1+2 psi(1/x) 0.223606797749979

21
1/racine de x 0.218217890235992
1+2 psi(x) / 1+2 psi(1/x) 0.218217890235993

22
1/racine de x 0.21320071635561
1+2 psi(x) / 1+2 psi(1/x) 0.21320071635561

23
1/racine de x 0.208514414057075
1+2 psi(x) / 1+2 psi(1/x) 0.208514414057075

24
1/racine de x 0.204124145231932
1+2 psi(x) / 1+2 psi(1/x) 0.204124145231932

25
1/racine de x 0.2
1+2 psi(x) / 1+2 psi(1/x) 0.2

26
1/racine de x 0.196116135138184
1+2 psi(x) / 1+2 psi(1/x) 0.196116135138184

27
1/racine de x 0.192450089729875
1+2 psi(x) / 1+2 psi(1/x) 0.192450089729875

28
1/racine de x 0.188982236504614
1+2 psi(x) / 1+2 psi(1/x) 0.188982236504614

29
1/racine de x 0.185695338177052
1+2 psi(x) / 1+2 psi(1/x) 0.185695338177052

30
1/racine de x 0.182574185835055
1+2 psi(x) / 1+2 psi(1/x) 0.182574185835055

31
1/racine de x 0.179605302026775
1+2 psi(x) / 1+2 psi(1/x) 0.179605302026775

32
1/racine de x 0.176776695296637
1+2 psi(x) / 1+2 psi(1/x) 0.176776695296637

33
1/racine de x 0.174077655955698
1+2 psi(x) / 1+2 psi(1/x) 0.174077655955698

34
1/racine de x 0.171498585142509
1+2 psi(x) / 1+2 psi(1/x) 0.171498585142509

35
1/racine de x 0.169030850945703
1+2 psi(x) / 1+2 psi(1/x) 0.169030850945704

36
1/racine de x 0.166666666666667
1+2 psi(x) / 1+2 psi(1/x) 0.166666666666667

37
1/racine de x 0.164398987305357
1+2 psi(x) / 1+2 psi(1/x) 0.164398987305357

38
1/racine de x 0.162221421130763
1+2 psi(x) / 1+2 psi(1/x) 0.162221421130763

39
1/racine de x 0.160128153805087
1+2 psi(x) / 1+2 psi(1/x) 0.160128153805087

40
1/racine de x 0.158113883008419
1+2 psi(x) / 1+2 psi(1/x) 0.158113883008419

41
1/racine de x 0.156173761888606
1+2 psi(x) / 1+2 psi(1/x) 0.156173761888606

42
1/racine de x 0.154303349962092
1+2 psi(x) / 1+2 psi(1/x) 0.154303349962092

43
1/racine de x 0.152498570332605
1+2 psi(x) / 1+2 psi(1/x) 0.152498570332605

44
1/racine de x 0.150755672288882
1+2 psi(x) / 1+2 psi(1/x) 0.150755672288882

45
1/racine de x 0.149071198499986
1+2 psi(x) / 1+2 psi(1/x) 0.149071198499986

46
1/racine de x 0.147441956154897
1+2 psi(x) / 1+2 psi(1/x) 0.147441956154897

47
1/racine de x 0.145864991497895
1+2 psi(x) / 1+2 psi(1/x) 0.145864991497895

48
1/racine de x 0.144337567297406
1+2 psi(x) / 1+2 psi(1/x) 0.144337567297406

49
1/racine de x 0.142857142857143
1+2 psi(x) / 1+2 psi(1/x) 0.142857142857143

50
1/racine de x 0.14142135623731
1+2 psi(x) / 1+2 psi(1/x) 0.14142135623731

51
1/racine de x 0.140028008402801
1+2 psi(x) / 1+2 psi(1/x) 0.140028008402801

52
1/racine de x 0.138675049056307
1+2 psi(x) / 1+2 psi(1/x) 0.138675049056307

53
1/racine de x 0.137360563948689
1+2 psi(x) / 1+2 psi(1/x) 0.137360563948689

54
1/racine de x 0.136082763487954
1+2 psi(x) / 1+2 psi(1/x) 0.136082763487954

55
1/racine de x 0.134839972492648
1+2 psi(x) / 1+2 psi(1/x) 0.134839972492649

56
1/racine de x 0.133630620956212
1+2 psi(x) / 1+2 psi(1/x) 0.133630620956213

57
1/racine de x 0.132453235706504
1+2 psi(x) / 1+2 psi(1/x) 0.132453235706504

58
1/racine de x 0.131306432859723
1+2 psi(x) / 1+2 psi(1/x) 0.131306432859723

59
1/racine de x 0.130188910980824
1+2 psi(x) / 1+2 psi(1/x) 0.130188910980824

60
1/racine de x 0.129099444873581
1+2 psi(x) / 1+2 psi(1/x) 0.129099444873581

61
1/racine de x 0.128036879932896
1+2 psi(x) / 1+2 psi(1/x) 0.128036879932896

62
1/racine de x 0.127000127000191
1+2 psi(x) / 1+2 psi(1/x) 0.127000127000191

63
1/racine de x 0.125988157669742
1+2 psi(x) / 1+2 psi(1/x) 0.125988157669743

64
1/racine de x 0.125
1+2 psi(x) / 1+2 psi(1/x) 0.125

65
1/racine de x 0.124034734589208
1+2 psi(x) / 1+2 psi(1/x) 0.124034734589209

66
1/racine de x 0.123091490979333
1+2 psi(x) / 1+2 psi(1/x) 0.123091490979333

67
1/racine de x 0.122169444356305
1+2 psi(x) / 1+2 psi(1/x) 0.122169444356305

68
1/racine de x 0.121267812518166
1+2 psi(x) / 1+2 psi(1/x) 0.121267812518167

69
1/racine de x 0.120385853085769
1+2 psi(x) / 1+2 psi(1/x) 0.120385853085769

70
1/racine de x 0.119522860933439
1+2 psi(x) / 1+2 psi(1/x) 0.11952286093344

71
1/racine de x 0.118678165819385
1+2 psi(x) / 1+2 psi(1/x) 0.118678165819386

72
1/racine de x 0.117851130197758
1+2 psi(x) / 1+2 psi(1/x) 0.117851130197758

73
1/racine de x 0.117041147196131
1+2 psi(x) / 1+2 psi(1/x) 0.117041147196131

74
1/racine de x 0.116247638743819
1+2 psi(x) / 1+2 psi(1/x) 0.116247638743819

75
1/racine de x 0.115470053837925
1+2 psi(x) / 1+2 psi(1/x) 0.115470053837925

76
1/racine de x 0.114707866935281
1+2 psi(x) / 1+2 psi(1/x) 0.114707866935281

77
1/racine de x 0.113960576459638
1+2 psi(x) / 1+2 psi(1/x) 0.113960576459638

78
1/racine de x 0.11322770341446
1+2 psi(x) / 1+2 psi(1/x) 0.11322770341446

79
1/racine de x 0.112508790092602
1+2 psi(x) / 1+2 psi(1/x) 0.112508790092602

80
1/racine de x 0.111803398874989
1+2 psi(x) / 1+2 psi(1/x) 0.11180339887499

81
1/racine de x 0.111111111111111
1+2 psi(x) / 1+2 psi(1/x) 0.111111111111111

82
1/racine de x 0.110431526074847
1+2 psi(x) / 1+2 psi(1/x) 0.110431526074847

83
1/racine de x 0.10976425998969
1+2 psi(x) / 1+2 psi(1/x) 0.10976425998969

84
1/racine de x 0.109108945117996
1+2 psi(x) / 1+2 psi(1/x) 0.109108945117996

85
1/racine de x 0.108465228909328
1+2 psi(x) / 1+2 psi(1/x) 0.108465228909328

86
1/racine de x 0.107832773203438
1+2 psi(x) / 1+2 psi(1/x) 0.107832773203439

87
1/racine de x 0.107211253483779
1+2 psi(x) / 1+2 psi(1/x) 0.10721125348378

88
1/racine de x 0.106600358177805
1+2 psi(x) / 1+2 psi(1/x) 0.106600358177805

89
1/racine de x 0.105999788000636
1+2 psi(x) / 1+2 psi(1/x) 0.105999788000636

90
1/racine de x 0.105409255338946
1+2 psi(x) / 1+2 psi(1/x) 0.105409255338946

91
1/racine de x 0.104828483672192
1+2 psi(x) / 1+2 psi(1/x) 0.104828483672192

92
1/racine de x 0.104257207028537
1+2 psi(x) / 1+2 psi(1/x) 0.104257207028537

93
1/racine de x 0.103695169473043
1+2 psi(x) / 1+2 psi(1/x) 0.103695169473043

94
1/racine de x 0.103142124625879
1+2 psi(x) / 1+2 psi(1/x) 0.103142124625879

95
1/racine de x 0.102597835208515
1+2 psi(x) / 1+2 psi(1/x) 0.102597835208515

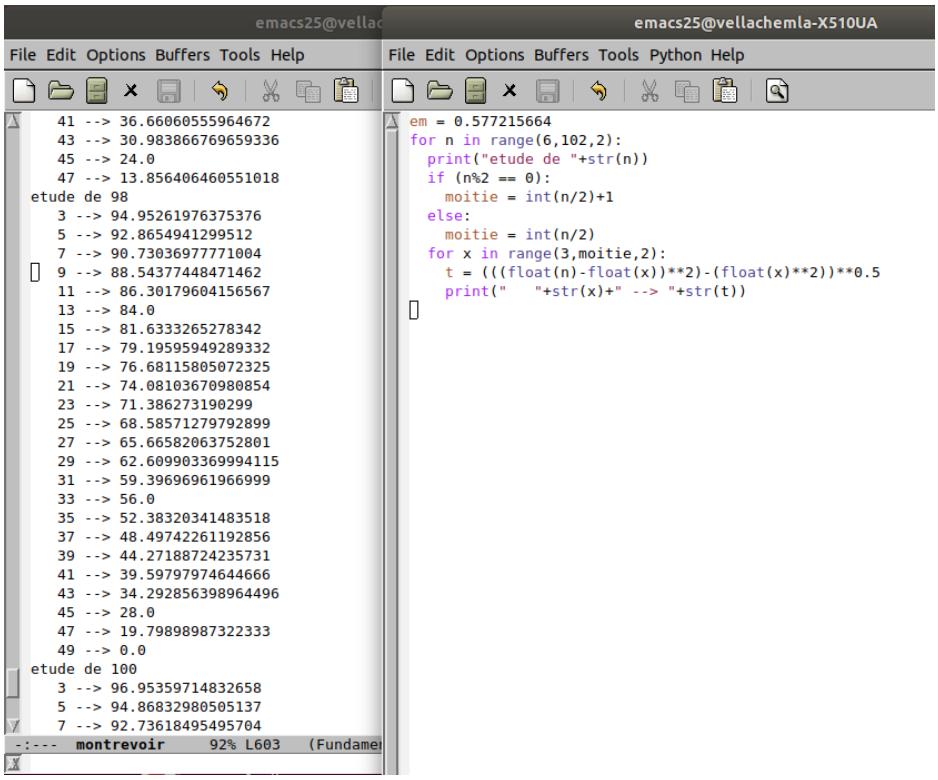
96
1/racine de x 0.102062072615966
1+2 psi(x) / 1+2 psi(1/x) 0.102062072615966

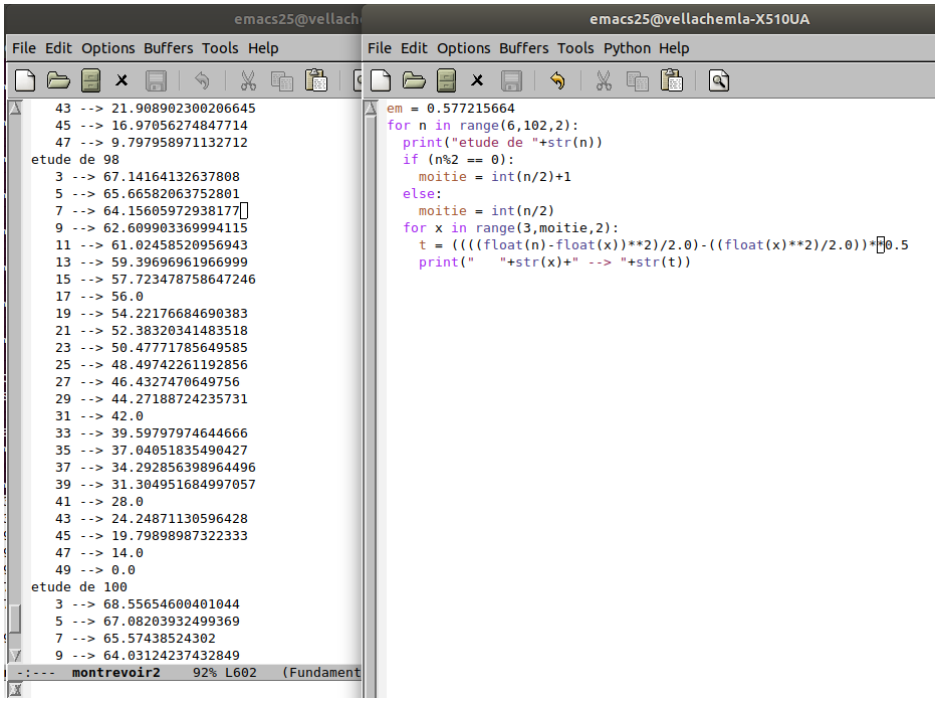
97
1/racine de x 0.101534616513362
1+2 psi(x) / 1+2 psi(1/x) 0.101534616513362

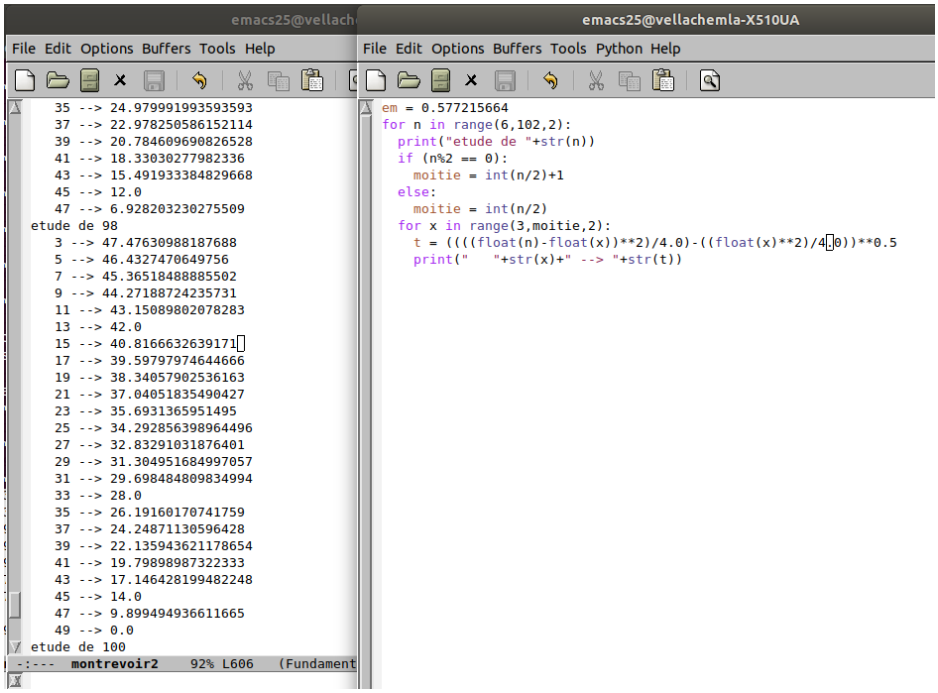
98
1/racine de x 0.101015254455221
1+2 psi(x) / 1+2 psi(1/x) 0.101015254455221

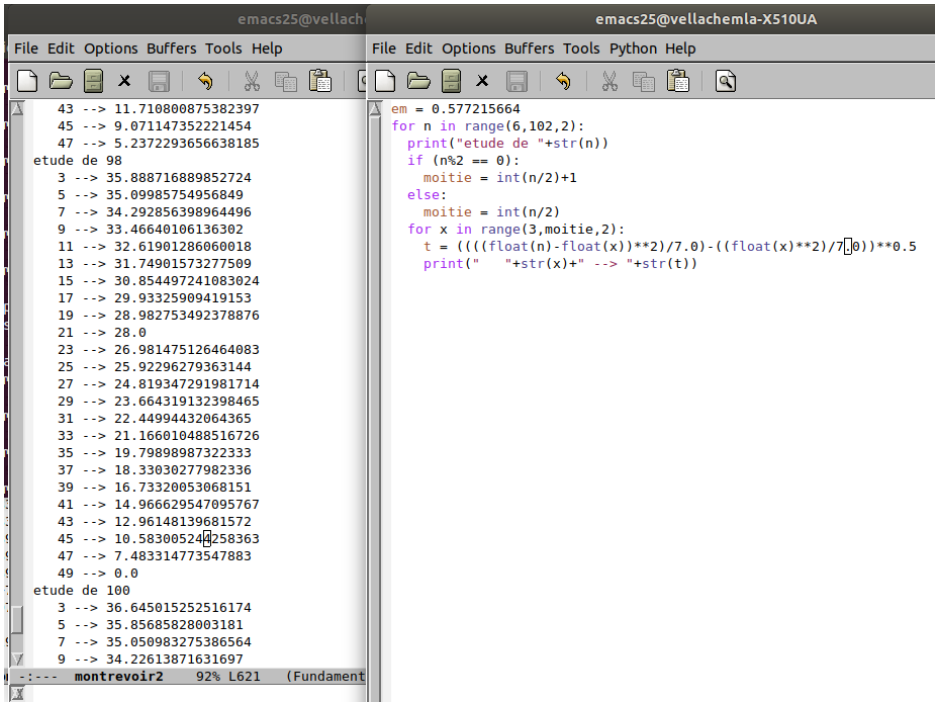
99
1/racine de x 0.100503781525921
1+2 psi(x) / 1+2 psi(1/x) 0.100503781525921

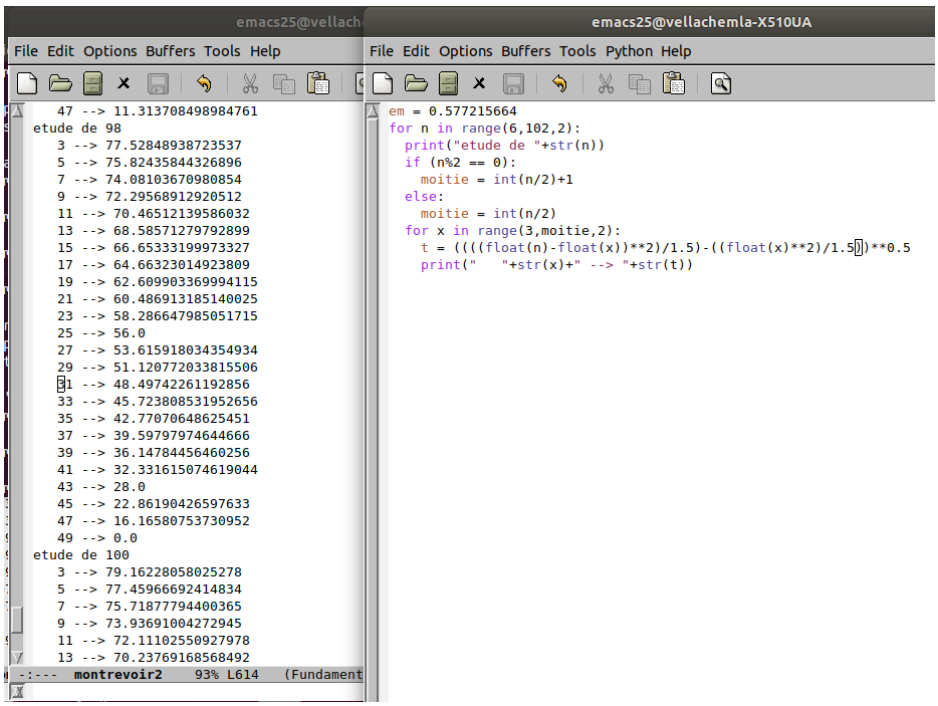
Essais de formules pour voir des entiers parmi les réels (Denise Vella-Chemla, décembre 2020)

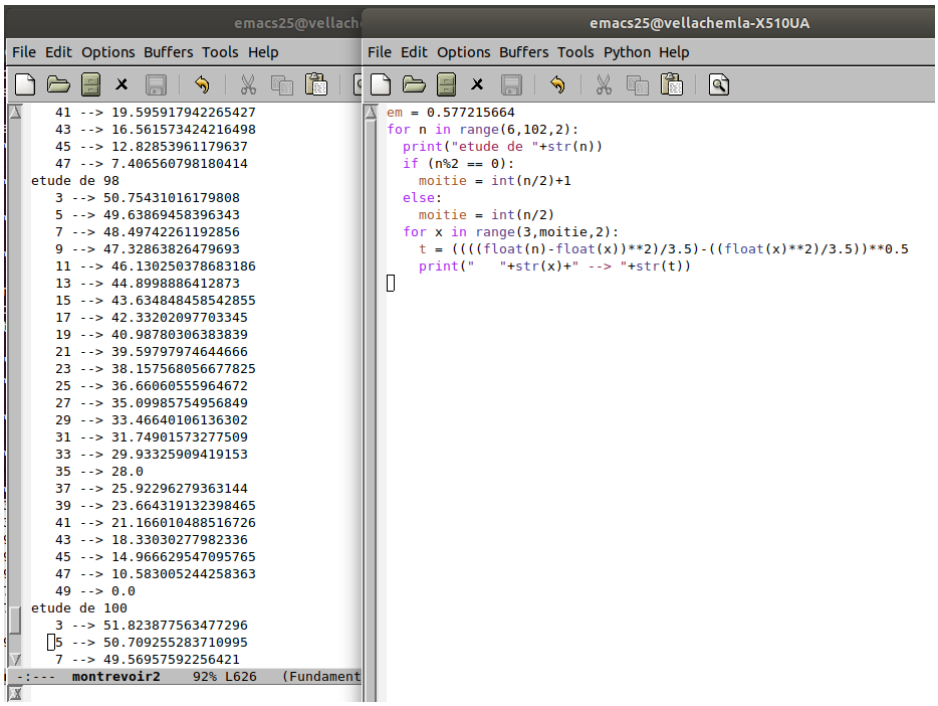


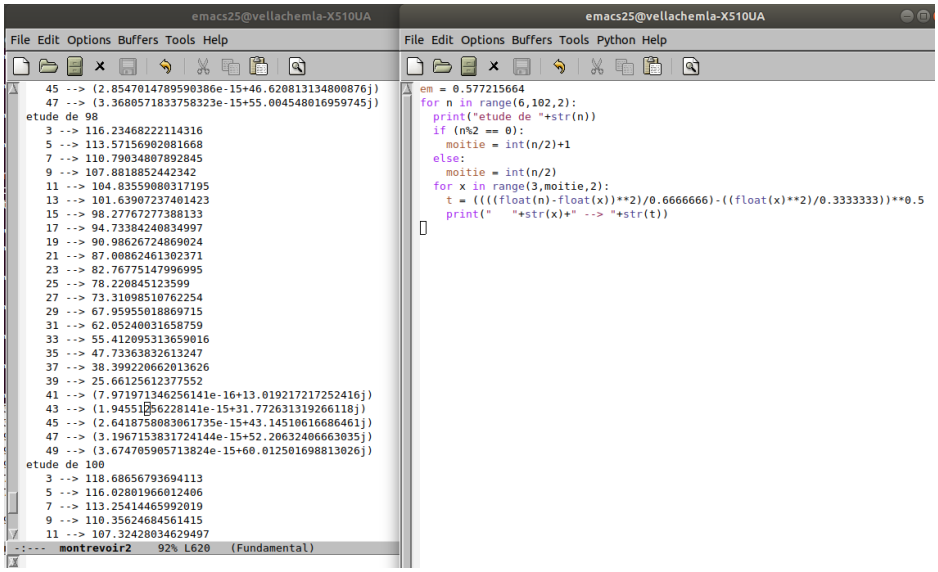


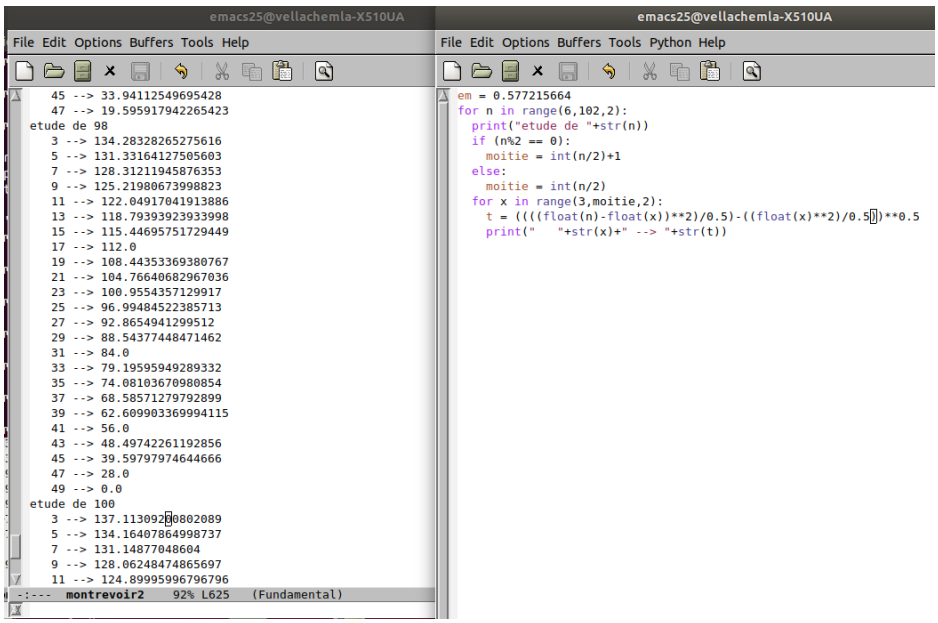

emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
35 --> 24.979991993593593
37 --> 22.978250586152114
39 --> 20.784609690826528
41 --> 18.33030277982336
43 --> 15.491933384829668
45 --> 12.0
47 --> 6.928203230275509
etude de 98
3 --> 47.47630988187688
5 --> 46.4327470649756
7 --> 45.36518488885502
9 --> 44.27188724235731
11 --> 43.15089802078283
13 --> 42.0
15 --> 40.8166632639171
17 --> 39.59797974644666
19 --> 38.34057902536163
21 --> 37.04051835490427
23 --> 35.6931365951495
25 --> 34.292856398964496
27 --> 32.83291031876401
29 --> 31.304951684997057
31 --> 29.698484809834994
33 --> 28.0
35 --> 26.19160170741759
37 --> 24.24871130596428
39 --> 22.135943621178654
41 --> 19.79898987322333
43 --> 17.146428199482248
45 --> 14.0
47 --> 9.899494936611665
49 --> 0.0
etude de 100
-:--- montrevoir2 92% L606 (Fundament
em = 0.577215664
for n in range(6,102,2):
print("etude de "+str(n))
if (n%2 == 0):
moitie = int(n/2)+1
else:
moitie = int(n/2)
for x in range(3,moitie,2):
t = ((((float(n)-float(x))**2)/4.0)-((float(x)**2)/4.0))**0.5
print(" "+str(x)+" --> "+str(t))

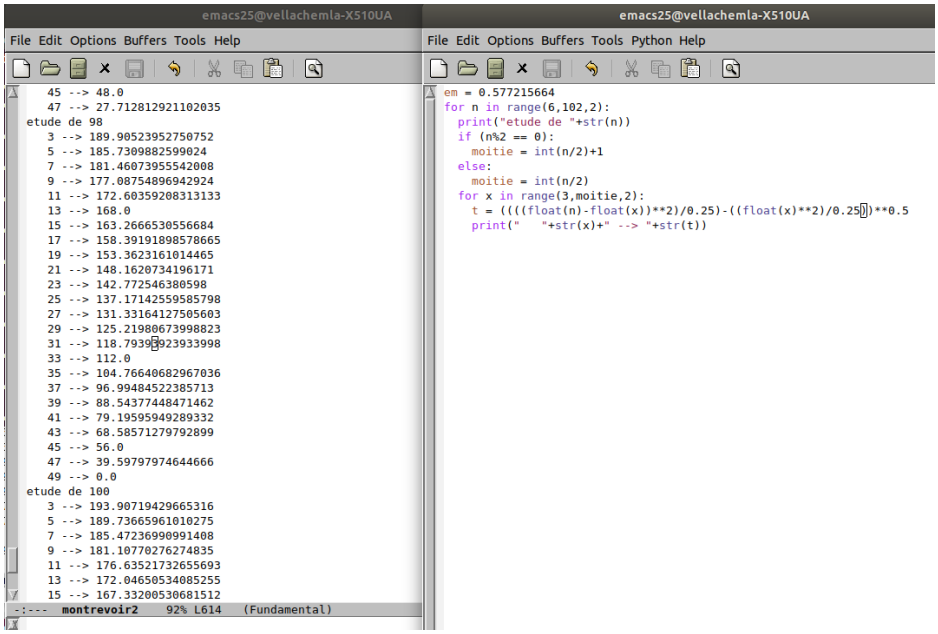

emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
43 --> 11.710800875382397
45 --> 9.071147352221454
47 --> 5.2372293656638185
etude de 98
3 --> 35.888716889852724
5 --> 35.09985754956849
7 --> 34.292856398964496
9 --> 33.46640106136302
11 --> 32.61901286060018
13 --> 31.74901573277509
15 --> 30.854497241083024
17 --> 29.93325909419153
19 --> 28.982753492378876
21 --> 28.0
23 --> 26.981475126464083
25 --> 25.92296279363144
27 --> 24.819347291981714
29 --> 23.664319132398465
31 --> 22.44994432064365
33 --> 21.166010488516726
35 --> 19.79898987322333
37 --> 18.33030277982336
39 --> 16.73320053068151
41 --> 14.966629547095767
43 --> 12.96148139681572
45 --> 10.583005244258363
47 --> 7.483314773547883
49 --> 0.0
etude de 100
3 --> 36.645015252516174
5 --> 35.85685828003181
7 --> 35.050983275386564
9 --> 34.22613871631697
-:--- montrevoir2 92% L621 (Fundament
em = 0.577215664
for n in range(6,102,2):
print("etude de "+str(n))
if (n%2 == 0):
moitie = int(n/2)+1
else:
moitie = int(n/2)
for x in range(3,moitie,2):
t = ((((float(n)-float(x))**2)/7.0)-((float(x)**2)/7.0))**0.5
print(" "+str(x)+" --> "+str(t))

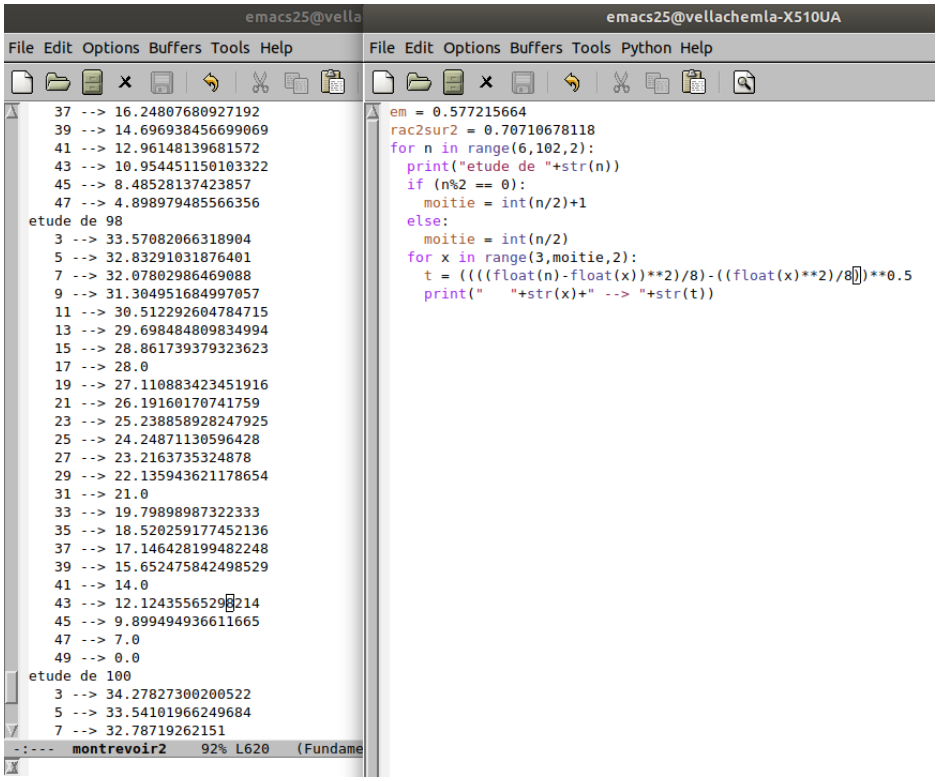

emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
47 --> 11.313708498984761
etude de 98
3 --> 77.52848938723537
5 --> 75.82435844326896
7 --> 74.08103670980854
9 --> 72.29568912920512
11 --> 70.46512139586032
13 --> 68.58571279792899
15 --> 66.65333199973327
17 --> 64.66323014923809
19 --> 62.609903369994115
21 --> 60.486913185140025
23 --> 58.286647985051715
25 --> 56.0
27 --> 53.615918034354934
29 --> 51.120772033815506
31 --> 48.49742261192856
33 --> 45.723808531952656
35 --> 42.77070648625451
37 --> 39.59797974644666
39 --> 36.14784456460256
41 --> 32.331615074619044
43 --> 28.0
45 --> 22.86190426597633
47 --> 16.16580753730952
49 --> 0.0
etude de 100
3 --> 79.16228058025278
5 --> 77.45966692414834
7 --> 75.71877794400365
9 --> 73.93691004272945
11 --> 72.11102550927978
13 --> 70.23769168568492
-:--- montrevoir2 93% L614 (Fundament
em = 0.577215664
for n in range(6,102,2):
print("etude de "+str(n))
if (n%2 == 0):
moitie = int(n/2)+1
else:
moitie = int(n/2)
for x in range(3,moitie,2):
t = ((((float(n)-float(x))**2)/1.5)-(((float(x)**2)/1.5))**0.5
print(" "+str(x)+" --> "+str(t))

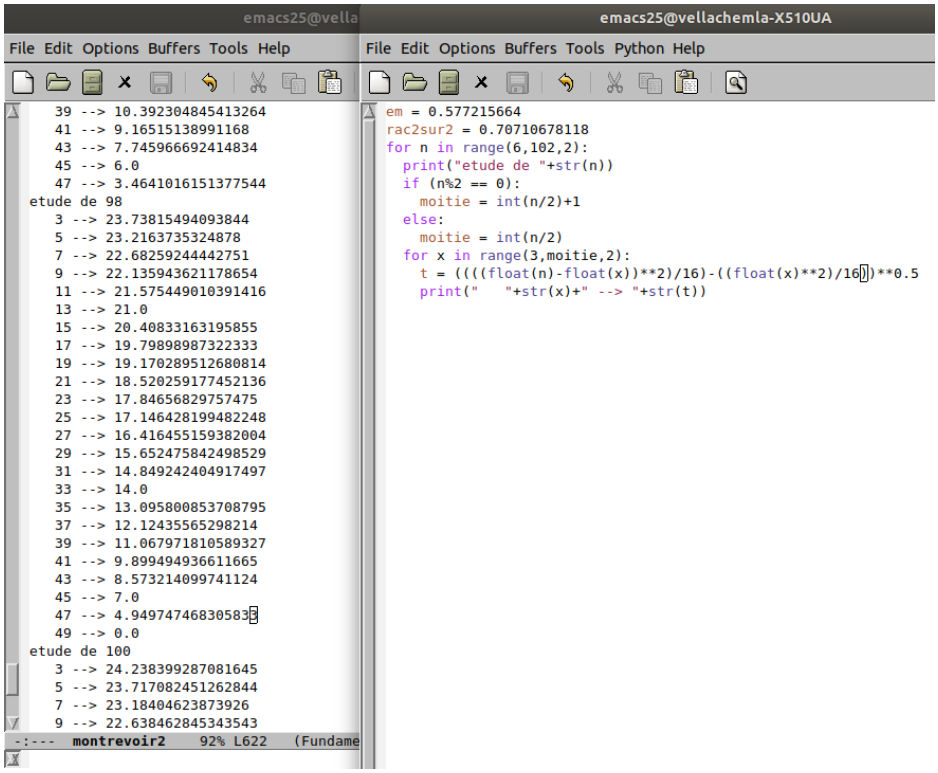

emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
41 --> 19.595917942265427
43 --> 16.561573424216498
45 --> 12.82853961179637
47 --> 7.406560798180414
etude de 98
3 --> 50.75431016179808
5 --> 49.63869458396343
7 --> 48.49742261192856
9 --> 47.32863826479693
11 --> 46.130250378683186
13 --> 44.8998886412873
15 --> 43.634848458542855
17 --> 42.33202097703345
19 --> 40.98780306383839
21 --> 39.59797974644666
23 --> 38.157568056677825
25 --> 36.66060555964672
27 --> 35.09985754956849
29 --> 33.46640106136302
31 --> 31.74901573277509
33 --> 29.93325909419153
35 --> 28.0
37 --> 25.92296279363144
39 --> 23.664319132398465
41 --> 21.166010488516726
43 --> 18.33030277982336
45 --> 14.966629547095765
47 --> 10.583005244258363
49 --> 0.0
etude de 100
3 --> 51.823877563477296
5 --> 50.709255283710995
7 --> 49.56957592256421
-:--- montrevoir2 92% L626 (Fundament
em = 0.577215664
for n in range(6,102,2):
print("etude de "+str(n))
if (n%2 == 0):
moitie = int(n/2)+1
else:
moitie = int(n/2)
for x in range(3,moitie,2):
t = ((((float(n)-float(x))**2)/3.5)-(((float(x)**2)/3.5))**0.5
print(" "+str(x)+" --> "+str(t))

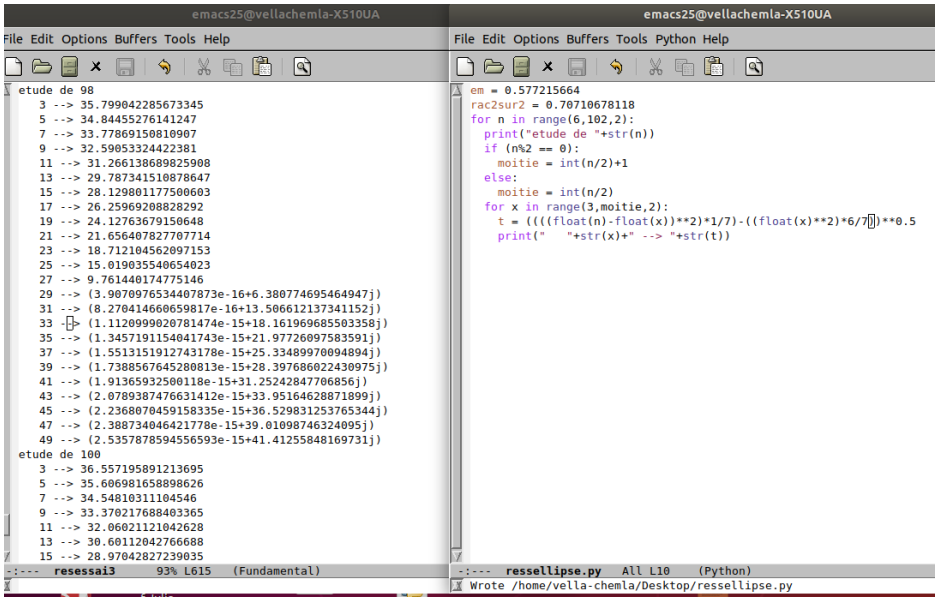

emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Help
45 --> (2.8547014789590386e-15+46.620813134800876j)
47 --> (3.3680571833758323e-15+55.004548016959745j)
etude de 98
3 --> 116.23468222114316
5 --> 113.57156902081668
7 --> 110.79034807892845
9 --> 107.8818852442342
11 --> 104.83559080317195
13 --> 101.63907237401423
15 --> 98.27767277388133
17 --> 94.73384240834997
19 --> 90.98626724869024
21 --> 87.00862461302371
23 --> 82.76775147996995
25 --> 78.220845123599
27 --> 73.31098510762254
29 --> 67.95955018869715
31 --> 62.05240031658759
33 --> 55.412095313659016
35 --> 47.73363832613247
37 --> 38.399220662013626
39 --> 25.66125612377552
41 --> (7.971971346256141e-16+13.019217217252416j)
43 --> (1.9455156228141e-15+31.772631319266118j)
45 --> (2.6418758083061735e-15+43.14510616686461j)
47 --> (3.1967153831724144e-15+52.20632406663035j)
49 --> (3.674705905713824e-15+60.012501698813026j)
etude de 100
3 --> 118.68656793694113
5 --> 116.02801966012406
7 --> 113.25414465992019
9 --> 110.35624684561415
11 --> 107.32428034629497
-:--- montrevoir2 92% L620 (Fundamental)
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
em = 0.577215664
for n in range(6,102,2):
print("etude de "+str(n))
if (n%2 == 0):
moitie = int(n/2)+1
else:
moitie = int(n/2)
for x in range(3,moitie,2):
t = ((((float(n)-float(x))**2)/0.6666666)-((float(x)**2)/0.3333333))**0.5
print(" "+str(x)+" --> "+str(t))


emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Help
45 --> 33.94112549695428
47 --> 19.595917942265423
etude de 98
3 --> 134.28328265275616
5 --> 131.33164127505603
7 --> 128.31211945876353
9 --> 125.21980673998823
11 --> 122.04917041913886
13 --> 118.79393923933998
15 --> 115.44695751729449
17 --> 112.0
19 --> 108.44353369380767
21 --> 104.76640682967036
23 --> 100.9554357129917
25 --> 96.99484522385713
27 --> 92.8654941299512
29 --> 88.54377448471462
31 --> 84.0
33 --> 79.19595949289332
35 --> 74.08103670980854
37 --> 68.58571279792899
39 --> 62.609903369994115
41 --> 56.0
43 --> 48.49742261192856
45 --> 39.59797974644666
47 --> 28.0
49 --> 0.0
etude de 100
3 --> 137.11309200802089
5 --> 134.16407864998737
7 --> 131.14877048604
9 --> 128.06248474865697
11 --> 124.89995996796796
-:--- montrevoir2 92% L625 (Fundamental)
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
em = 0.577215664
for n in range(6,102,2):
print("etude de "+str(n))
if (n%2 == 0):
moitie = int(n/2)+1
else:
moitie = int(n/2)
for x in range(3,moitie,2):
t = ((((float(n)-float(x))**2)/0.5)-((float(x)**2)/0.5))**0.5
print(" "+str(x)+" --> "+str(t))


emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Help
45 --> 48.0
47 --> 27.712812921102035
etude de 98
3 --> 189.90523952750752
5 --> 185.7309882599024
7 --> 181.46073955542008
9 --> 177.08754896942924
11 --> 172.60359208313133
13 --> 168.0
15 --> 163.2666530556684
17 --> 158.39191898578665
19 --> 153.3623161014465
21 --> 148.1620734196171
23 --> 142.772546380598
25 --> 137.17142559585798
27 --> 131.33164127505603
29 --> 125.21980673998823
31 --> 118.79393923933998
33 --> 112.0
35 --> 104.76640682967036
37 --> 96.99484522385713
39 --> 88.54377448471462
41 --> 79.19595949289332
43 --> 68.58571279792899
45 --> 56.0
47 --> 39.59797974644666
49 --> 0.0
etude de 100
3 --> 193.90719429665316
5 --> 189.73665961010275
7 --> 185.47236990991408
9 --> 181.10770276274835
11 --> 176.63521732655693
13 --> 172.04650534085255
15 --> 167.33200530681512
-:--- montrevoir2 92% L614 (Fundamental)
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
em = 0.577215664
for n in range(6,102,2):
print("etude de "+str(n))
if (n%2 == 0):
moitie = int(n/2)+1
else:
moitie = int(n/2)
for x in range(3,moitie,2):
t = ((((float(n)-float(x))**2)/0.25)-((float(x)**2)/0.25))**0.5
print(" "+str(x)+" --> "+str(t))


emacs25@vella
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Help
File Edit Options Buffers Tools Python Help
37 --> 16.24807680927192
39 --> 14.696938456699069
41 --> 12.96148139681572
43 --> 10.954451150103322
45 --> 8.48528137423857
47 --> 4.898979485566356
etude de 98
3 --> 33.57082066318904
5 --> 32.83291031876401
7 --> 32.07802986469088
9 --> 31.304951684997057
11 --> 30.512292604784715
13 --> 29.698484809834994
15 --> 28.861739379323623
17 --> 28.0
19 --> 27.110883423451916
21 --> 26.19160170741759
23 --> 25.238858928247925
25 --> 24.24871130596428
27 --> 23.2163735324878
29 --> 22.135943621178654
31 --> 21.0
33 --> 19.79898987322333
35 --> 18.520259177452136
37 --> 17.146428199482248
39 --> 15.652475842498529
41 --> 14.0
43 --> 12.124355652982214
45 --> 9.899494936611665
47 --> 7.0
49 --> 0.0
etude de 100
3 --> 34.27827300200522
5 --> 33.54101966249684
7 --> 32.78719262151
-:--- montrevoir2 92% L620 (Fundame
em = 0.577215664
rac2sur2 = 0.70710678118
for n in range(6,102,2):
print("etude de "+str(n))
if (n%2 == 0):
moitie = int(n/2)+1
else:
moitie = int(n/2)
for x in range(3,moitie,2):
t = ((((float(n)-float(x))**2)/8)-((float(x)**2)/8))**0.5
print(" "+str(x)+" --> "+str(t))


emacs25@vella
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Help
File Edit Options Buffers Tools Python Help
39 --> 10.392304845413264
41 --> 9.16515138991168
43 --> 7.745966692414834
45 --> 6.0
47 --> 3.4641016151377544
etude de 98
3 --> 23.73815494093844
5 --> 23.2163735324878
7 --> 22.68259244442751
9 --> 22.135943621178654
11 --> 21.575449010391416
13 --> 21.0
15 --> 20.40833163195855
17 --> 19.79898987322333
19 --> 19.170289512680814
21 --> 18.520259177452136
23 --> 17.84656829757475
25 --> 17.146428199482248
27 --> 16.416455159382004
29 --> 15.652475842498529
31 --> 14.849242404917497
33 --> 14.0
35 --> 13.095800853708795
37 --> 12.12435565298214
39 --> 11.067971810589327
41 --> 9.899494936611665
43 --> 8.573214099741124
45 --> 7.0
47 --> 4.949747468305833
49 --> 0.0
etude de 100
3 --> 24.238399287081645
5 --> 23.717082451262844
7 --> 23.18404623873926
9 --> 22.638462845343543
-:--- montrevoir2 92% L622 (Fundame
em = 0.577215664
rac2sur2 = 0.70710678118
for n in range(6,102,2):
print("etude de "+str(n))
if (n%2 == 0):
moitie = int(n/2)+1
else:
moitie = int(n/2)
for x in range(3,moitie,2):
t = ((((float(n)-float(x))**2)/16)-((float(x)**2)/16))**0.5
print(" "+str(x)+" --> "+str(t))


emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Help
etude de 98
3 --> 35.799042285673345
5 --> 34.84455276141247
7 --> 33.77869150810907
9 --> 32.59053324422381
11 --> 31.266138689825908
13 --> 29.787341510878647
15 --> 28.129801177500603
17 --> 26.25969208828292
19 --> 24.12763679150648
21 --> 21.656407827707714
23 --> 18.712104562097153
25 --> 15.019035540654023
27 --> 9.761440174775146
29 --> (3.9070976534407873e-16+6.380774695464947j)
31 --> (8.270414660659817e-16+13.506612137341152j)
33 --> (1.1120999020781474e-15+18.161969685503358j)
35 --> (1.3457191154041743e-15+21.97726097583591j)
37 --> (1.5513151912743178e-15+25.33489970094894j)
39 --> (1.7388567645280813e-15+28.397686022430975j)
41 --> (1.91365932500118e-15+31.25242847706856j)
43 --> (2.0789387476631412e-15+33.951646288711899j)
45 --> (2.2368070459158335e-15+36.529831253765344j)
47 --> (2.388734046421778e-15+39.01098746324095j)
49 --> (2.5357878594556593e-15+41.41255848169731j)
etude de 100
3 --> 36.557195891213695
5 --> 35.606981658898626
7 --> 34.54810311104546
9 --> 33.370217688403365
11 --> 32.06021121042628
13 --> 30.60112042766688
15 --> 28.9704282729035
-:--- resesai3 93% L615 (Fundamental)
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
em = 0.577215664
rac2sur2 = 0.70710678118
for n in range(6,102,2):
print("etude de "+str(n))
if (n%2 == 0):
moitie = int(n/2)+1
else:
moitie = int(n/2)
for x in range(3,moitie,2):
t = ((((float(n)-float(x))**2)*1/7)-((float(x)**2)*6/7))**0.5
print(" "+str(x)+" --> "+str(t))
-:--- ressellipse.py All L10 (Python)
Wrote /home/vella-chemla/Desktop/ressellipse.py

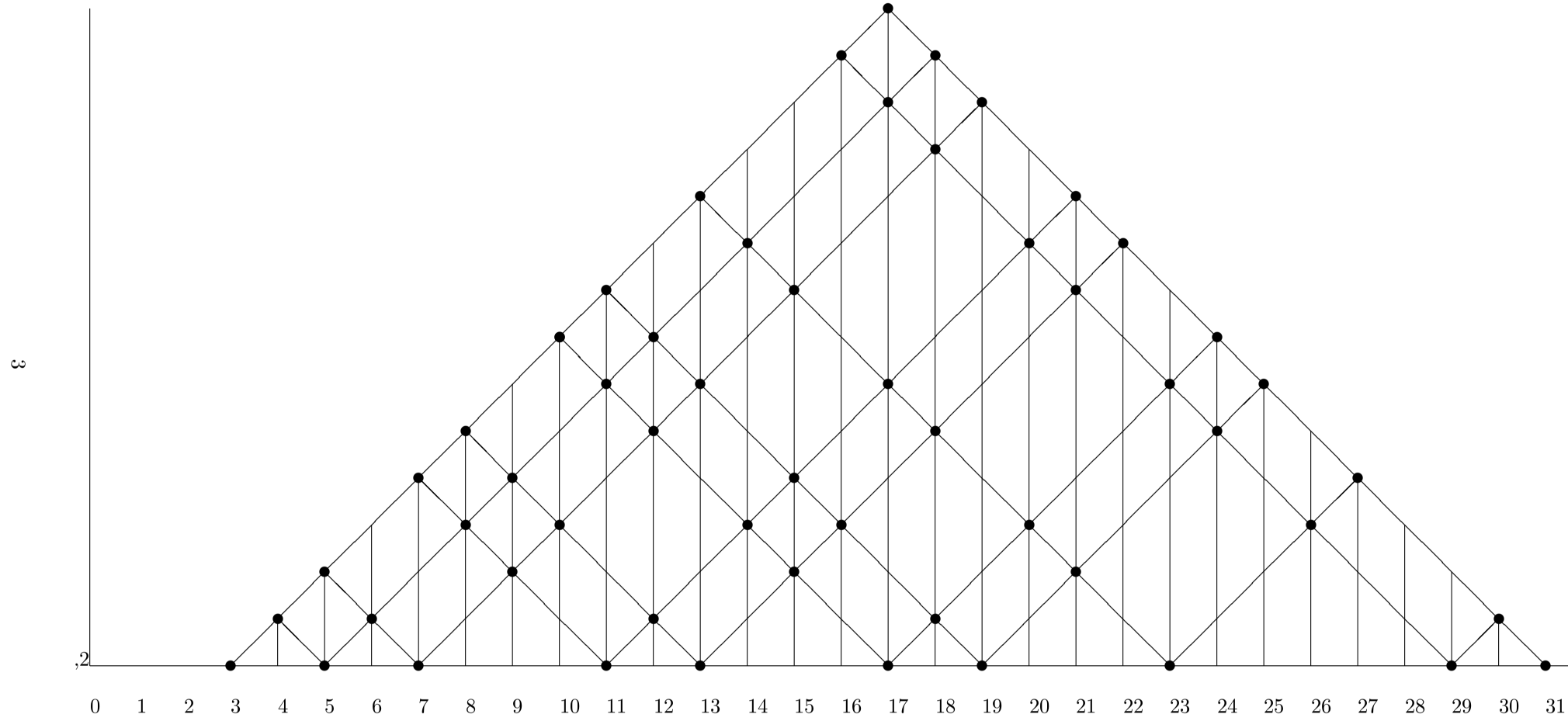


Figure 1: Le treillis Goldbach

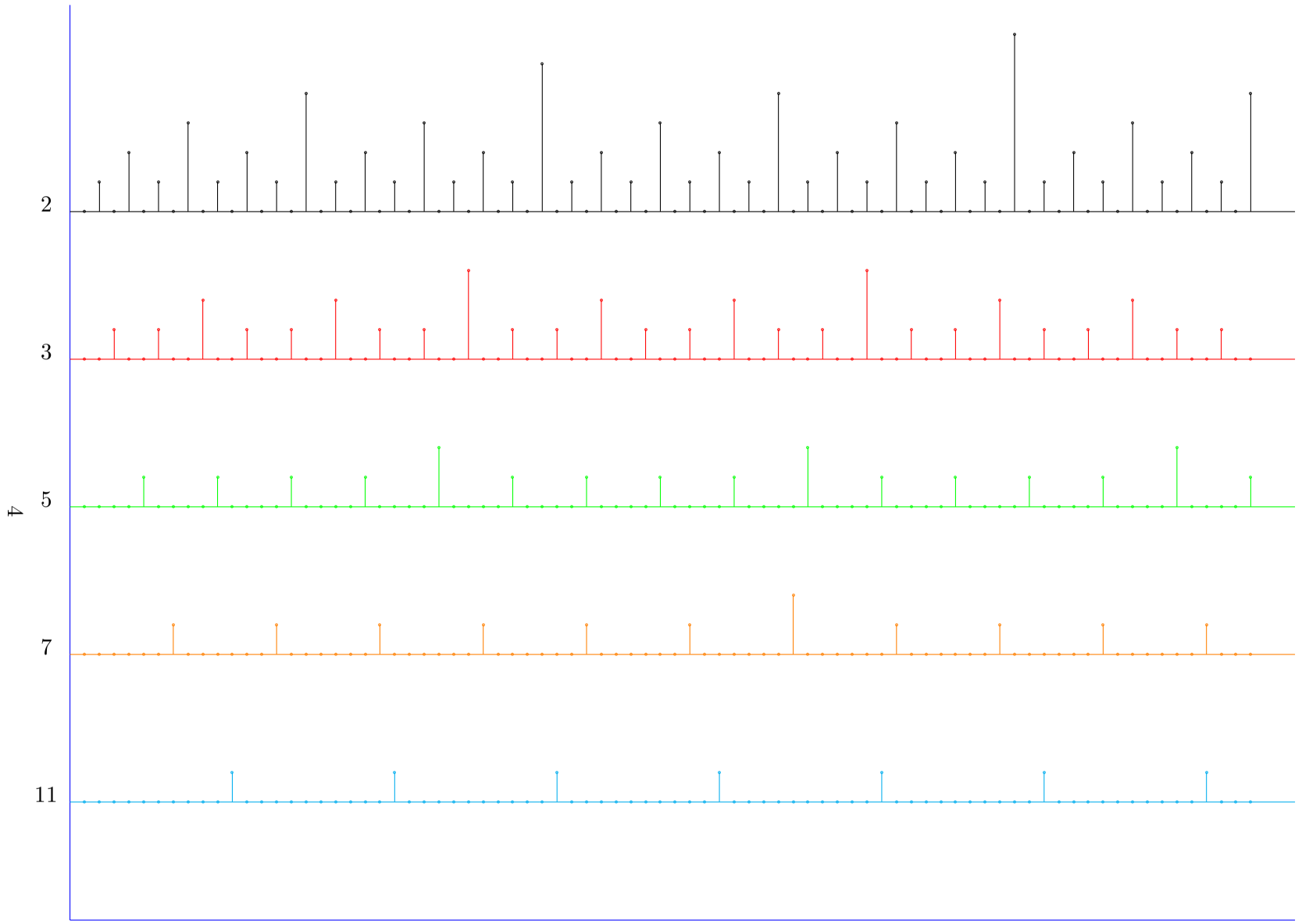
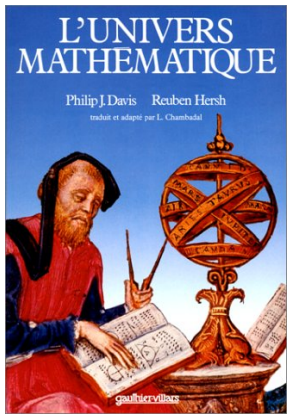


Figure 1 : Séquences fractales de valuations p-adiques

Le partage de *dg*

- $20902 = 3 + 20899$ $20962 = 3 + 20959$
  $20904 = 5 + 20899$ $20964 = 5 + 20959$
  $20906 = 3 + 20903$ $20966 = 3 + 20963$
  $20908 = 5 + 20903$ $20968 = 5 + 20963$
  $20910 = 7 + 20903$ $20970 = 7 + 20963$
  $20912 = 13 + 20899$ $20972 = 13 + 20959$
  $20914 = 11 + 20903$ $20974 = 11 + 20963$
  $20916 = 13 + 20903$ $20976 = 13 + 20963$
  $20918 = 19 + 20899$ $20978 = 19 + 20959$
  $20920 = 17 + 20903$ $20980 = 17 + 20963$
  $20922 = 19 + 20903$ $20982 = 19 + 20963$
  $20924 = 3 + 20921$ $20984 = 3 + 20981$



- Des causes différentes produisent les mêmes effets (écart de 60, congrus mod 3 et 5).

Permutations des racines

- $$\begin{array}{ccc} 2p_i & \xrightarrow{f} & 2c \\ \downarrow{\scriptstyle g_t} & & \downarrow{\scriptstyle g} \\ p_i & \xrightarrow{f} & p_j \end{array}$$

- $$\begin{array}{ccc} 94 = (1,4,3) & \xrightarrow{f} & 88 = (1,3,4) \\ \downarrow{\scriptstyle g_t} & & \downarrow{\scriptstyle g} \\ 47 = (2,2,5) & \xrightarrow{f} & 29 = (2,4,1) \end{array}$$

$$\begin{array}{ccc} \mathbb{Z}/3\mathbb{Z} \times \mathbb{Z}/5\mathbb{Z} \times \mathbb{Z}/7\mathbb{Z} & \xrightarrow{f} & \mathbb{Z}/3\mathbb{Z} \times \mathbb{Z}/5\mathbb{Z} \times \mathbb{Z}/7\mathbb{Z} \\ \downarrow{\scriptstyle g_t} & & \downarrow{\scriptstyle g} \\ \mathbb{Z}/3\mathbb{Z} \setminus \{0,1\} \times \mathbb{Z}/5\mathbb{Z} \setminus \{0,4\} \times \mathbb{Z}/7\mathbb{Z} \setminus \{0,3\} & \xrightarrow{f} & \mathbb{Z}/3\mathbb{Z} \setminus \{0,1\} \times \mathbb{Z}/5\mathbb{Z} \setminus \{0,3\} \times \mathbb{Z}/7\mathbb{Z} \setminus \{0,4\} \end{array}$$

Permutations des racines

- $$\begin{array}{ccc} 94 = (1,4,3) & \xrightarrow{f} & 88 = (1,3,4) \\ \downarrow g_t & & \downarrow g \\ 47 = (2,2,5) & \xrightarrow{f} & 29 = (2,4,1) \end{array}$$

- $\mathbb{Z}/3\mathbb{Z} \to Id$,
  $\mathbb{Z}/5\mathbb{Z} \to \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 0 & 1 & 4 & 2 & 3 \end{pmatrix}$,
  $\mathbb{Z}/7\mathbb{Z} \to \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ 0 & 5 & 2 & 4 & 3 & 1 & 6 \end{pmatrix}$

Conclusion

- On a utilisé un SNURPF : un Système de NUmération par les Restes dans les Parties Finies de $\mathbb{N}$.

- On se situe dans une théorie lexicale des nombres, selon laquelle les nombres sont des mots.

Continuer de suivre Galois (DV - 21/12/2013)

On cherche une équation polynomiale qui aurait ses racines qui se verraient permutées par une certaine fonction et dont les solutions seraient les décomposants de Goldbach de n, un nombre pair, i.e. les nombres premiers dont les complémentaires à n seraient premiers également.

On "sent bien" que le générateur doit sûrement être la fonction $f : x \mapsto n - x$ car cette fonction envoie chaque nombre entier sur son complémentaire à n, la somme de ces deux nombres permettant d'obtenir n.

On trouve donc l'inéquation polynomiale $x^2 - nx \neq 0$ qui est invariante par la fonction f. En effet, $(n-x)^2 - n(n-x) = x^2 + n^2 - 2nx - n^2 + nx = x^2 - nx$. On est conforté dans cette idée par le fait que le polynôme proposé est égal à $x(n-x)$:

- d'une part, ce polynôme s'annule lorsque x est nul et la congruence $x \not\equiv 0 \ (mod\ p_i)$ dans tous les corps premiers $\mathbb{Z}/p_i\mathbb{Z}$ pour p_i un nombre premier quelconque inférieur à $\sqrt{n}$ correspond au fait que x est un nombre premier supérieur à $\sqrt{n}$;

- d'autre part, ce polynôme s'annule lorsque $x = n$ et la congruence $x \not\equiv n \ (mod\ p_i)$ dans tous les corps premiers $\mathbb{Z}/p_i\mathbb{Z}$ pour p_i un nombre premier quelconque inférieur à $\sqrt{n}$ correspond au fait que le complémentaire de x à n est premier.

Il faudrait pour prouver la conjecture de Goldbach être assuré que cette inéquation polynomiale $x^2 - nx \neq 0$ a une solution commune inférieure à $n/2$ dans tous les corps premiers $\mathbb{Z}/p_i\mathbb{Z}$ avec p_i un nombre premier quelconque inférieur à $\sqrt{n}$.

Traitons l'exemple de la recherche des décompositions de Goldbach de 98.
Le polynôme $x^2 - 98x$ est égal à $x^2 - 2x$ dans $\mathbb{Z}/3\mathbb{Z}$ tandis qu'il est égal à $x^2 - 3x$ dans $\mathbb{Z}/5\mathbb{Z}$, ou encore égal à x^2 tout simplement dans $\mathbb{Z}/7\mathbb{Z}$ puisque 7 divise 98.

Notons dans un tableau pour les nombres premiers supérieurs à $\sqrt{98}$ et inférieurs à 49 la moitié de 98 les valeurs des polynômes en question et voyons ceux qui sont éliminés dans chacun des corps premiers.

	11	13	17	19	23	29	31	37	41	43	47
x^2 (dont on teste la nullité dans $\mathbb{Z}/7\mathbb{Z}$)	121	169	289	361	529	841	961	1369	1681	1849	2209
$x^2 - 2x$ (dont on teste la nullité dans $\mathbb{Z}/3\mathbb{Z}$)	~~99~~	143	~~255~~	323	~~483~~	~~783~~	899	1295	~~1599~~	1763	~~2115~~
$x^2 - 3x$ (dont on teste la nullité dans $\mathbb{Z}/5\mathbb{Z}$)	88	~~130~~	238	304	~~460~~	754	868	1258	1558	~~1720~~	2068

On voit que ne sont conservés que les nombres 19, 31 et 37 qui sont comme attendu les décomposants de Goldbach de 98.

Le problème de Goldbach est en quelque sorte un problème "relatif" (puisqu'à la recherche des décomposants de Goldbach de n le nombre n intervient dans l'inéquation dont il faut chercher une solution commune dans tous les corps finis $\mathbb{Z}/p_k\mathbb{Z}$ pour $p_k \leqslant \sqrt{n}$).

On peut considérer que le problème des jumeaux est quant à lui le problème "absolu" correspondant au problème "relatif" de Goldbach. En effet, si l'on appelle "père de jumeaux" le nombre pair entre deux nombres premiers jumeaux (par exemple 18 entre 17 et 19 ou encore 570 entre 569 et 571), ce nombre doit vérifier l'inéquation "absolue" $x^2 \not\equiv 1(mod\ p_k)$ pour tout $p_k \leqslant \sqrt{x+1}$ (il doit en effet vérifier simplement $(x-1)(x+1) \not\equiv 0(mod\ p_k)$ pour qu'$x-1$ et $x+1$ soient premiers tous les deux). Un père de jumeau est obligatoirement de la forme $6k$. Fournissons dans un tableau la classe de congruence de x^2 selon les modules premiers impairs inférieurs à $\sqrt{x+1}$ qui nous permettent d'aisément trouver les pères de jumeaux jusqu'à 300.

père	*mod* 3	*mod* 5	*mod* 7	*mod* 11	*mod* 13	*mod* 17	*jumeaux*
6							$(5,7)$
12	0						$(11,13)$
18	0						$(17,19)$
24	0	1					
30	0	0					$(29,31)$
36	0	1					
42	0	4					$(41,43)$
48	0	4	1				
54	0	1	4				
60	0	0	2				$(59,61)$
66	0	1	2				
72	0	4	4				$(71,73)$
78	0	4	1				
84	0	1	0				
90	0	0	1				
96	0	1	4				
102	0	4	2				$(101,103)$
108	0	4	2				$(107,109)$
114	0	1	4				
120	0	0	1	1			
126	0	1	0	3			
132	0	4	1	0			
138	0	4	4	3			$(137,139)$
144	0	1	2	1			
150	0	0	2	5			$(149,151)$
156	0	1	4	4			
162	0	4	1	9			
168	0	4	0	9	1		
174	0	1	1	4	12		
180	0	0	4	5	4		$(179,181)$
186	0	1	2	1	3		
192	0	4	2	3	9		$(191,193)$
198	0	4	4	0	9		$(197,199)$
204	0	1	1	3	3		
210	0	0	0	1	4		
216	0	1	1	5	12		
222	0	4	4	4	1		
228	0	4	2	9	10		$(227,229)$
234	0	1	2	9	0		
240	0	0	4	4	10		$(239,241)$
246	0	1	1	5	1		
252	0	4	0	1	12		
258	0	4	1	3	4		
264	0	1	4	0	3		
270	0	0	2	3	9		$(269,271)$
276	0	1	2	1	9		
282	0	4	4	5	3		$(281,283)$
288	0	4	1	4	4		
294	0	1	0	9	12	8	
300	0	0	1	9	1	2	

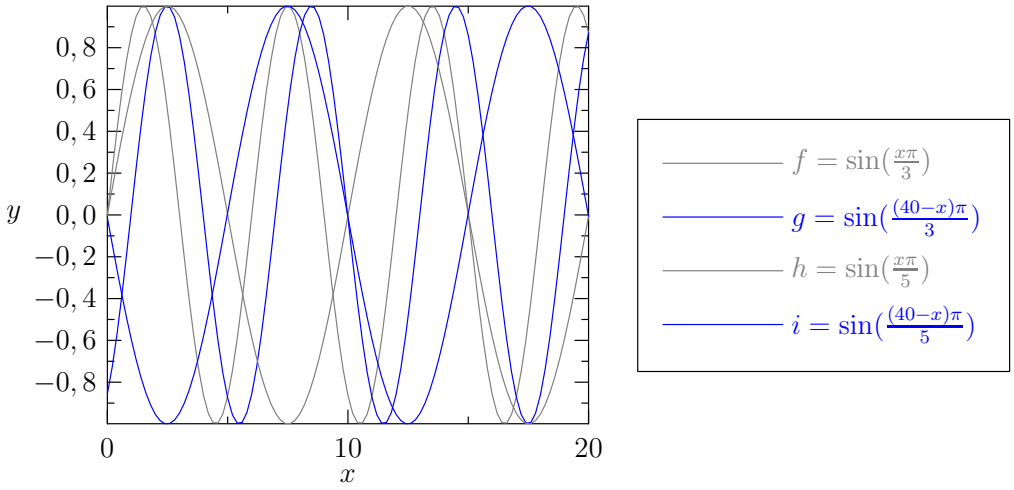
On ne réexplique pas ici la représentation par les grilles de divisibilité qui nous a permis de mieux comprendre la conjecture de Goldbach et dont l'exemple du nombre pair 40 est représenté ci-dessus. 11 et 17, qui ne sont divisibles ni par 3 ni par 5 et qui ne partagent avec 40 aucun de leur reste dans des divisions euclidiennes par 3 ou 5 (i.e. dont la colonne ne contient pas de case colorée) sont des décomposants de Goldbach de 40.

On a proposé à partir de ces grilles la possibilité de trouver les décomposants de Goldbach en calculant des produits de sinusoïdes.

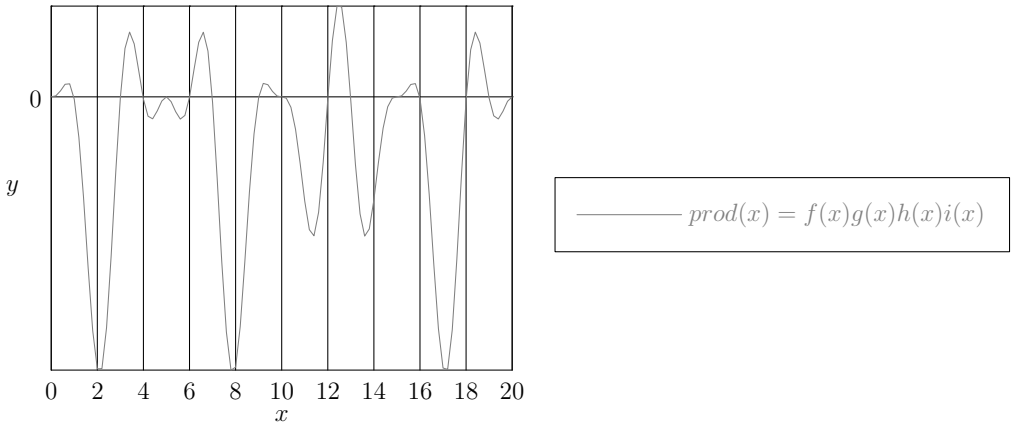
Les décomposants de Goldbach de n sont en effet les seuls nombres entiers impairs inférieurs à $n/2$ qui n'annulent pas le produit suivant :

$$\prod_{3 \leqslant p \ un \ nb \ 1^{er} \leqslant \sqrt{n}} sin\left(\frac{x\pi}{p}\right).sin\left(\frac{(n-x)\pi}{p}\right)$$

Les sinusoïdes correspondant au cas du nombre pair 40 (se reporter à la grille de divisibilité ci-dessus) sont :



Leur produit ne s'annule effectivement pas pour les nombres entiers impairs 11 et 17.

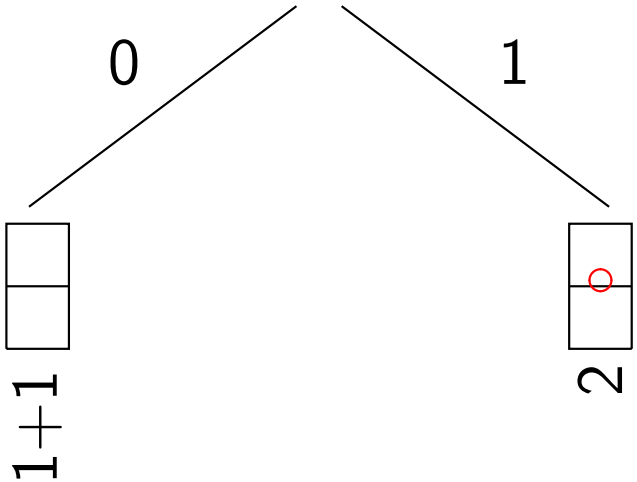


On peut établir une analogie entre ces sinusoïdes et les fonctions d'onde de la mécanique quantique. En poussant l'analogie, on peut imaginer qu'on puisse établir une probabilité qu'un nombre pair ait un décomposant de Goldbach sans pouvoir établir sa valeur, selon une sorte de principe d'incertitude.

Enfin, si on souhaite établir une analogie avec la propriété d'*intrication quantique* : on associe à chaque case de la grille de divisibilité ci-dessus un q-bit qui est simultanément dans les états 0 et 1. On peut imaginer cette grille comme de taille infinie si on considère tous les nombres pairs d'un même coup. Le fait de fixer

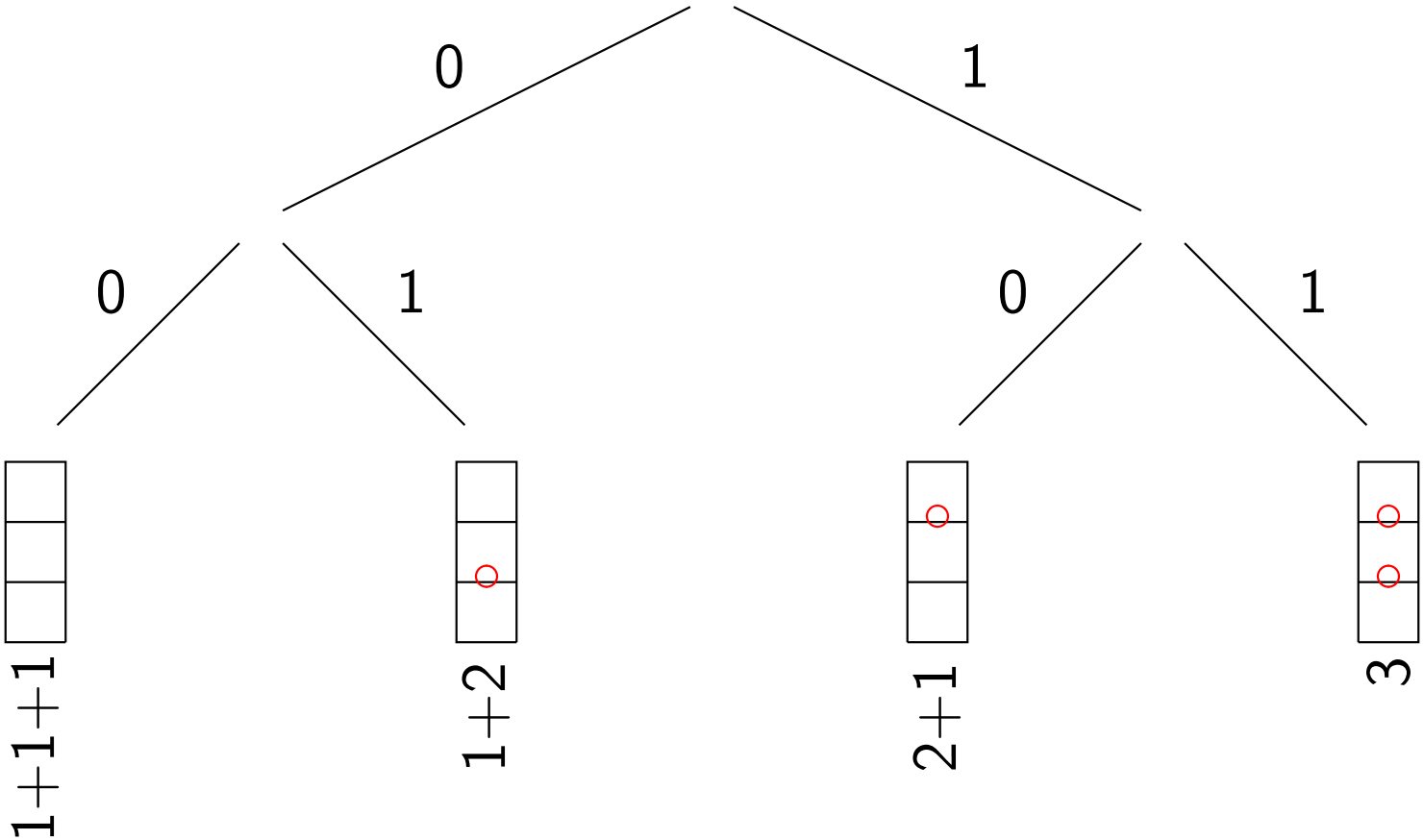
Compositions et mots booléens

- A chaque entier n, on associe ses 2^{n-1} compositions additives.
- *exemple* : arbre binaire des compositions de 2



Exemple

- *exemple* : arbre binaire des compositions de 3



- Noter que la composition $1 + 2$ est différente de la composition $2 + 1$.

Obtention des compositions de $n+1$ à partir de celles de n

- On concatène 0 ou 1 au début de chaque mot booléen de n.
- Cela correspond à deux actions syntaxiques : concaténer “1+” en début de composition ou bien remplacer le premier sommant par son successeur.

Nombre composé / nombre premier

- On appelle compositions triviales la composition correspondant au mot booléen ne contenant que des 0 (composition de la forme $1+1+\ldots+1$) ou bien la composition correspondant au mot booléen contenant $n-1$ lettres 1 (composition de la forme n).

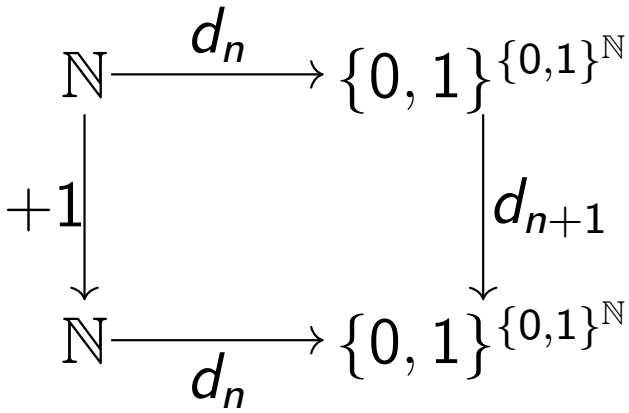
- Un nombre composé admet au moins une décomposition non triviale de la forme $x+x+x+\ldots+x$ contenant 2 occurrences de x au moins.

- A chaque entier est associé un ensemble de mots booléens, i.e. un ensemble de parties de $\mathbb{N}$.

$$\mathbb{N} \longrightarrow \{0,1\}^{\{0,1\}^{\mathbb{N}}}$$
$$n \longmapsto \{s \in \{0,1\}^{\mathbb{N}} / \forall i \geqslant n, s[i] = 0\} \subset \mathcal{P}(\mathbb{N})$$

Formalisation

- Le passage de n à $n+1$ est codé par le diagramme suivant :



avec $d_n : \mathbb{N} \longrightarrow \{0,1\}$

et

$$d_{n+1} : \begin{array}{ccc} k & \longmapsto & d_n(k-1), \forall k \geqslant 1 \\ 0 & \longmapsto & 0 \end{array}$$

- faire l'union de cet ensemble de fonctions avec l'ensemble des fonctions d_{n+1} qui associent l'image 1 (plutôt que 0) à 0.

Nombre composé / nombre premier

- Un nombre est composé n si l'un de ces mots non triviaux (dont on considère les $n-1$ premières lettres, i.e. la partie des mots avant l'infinité de zéros) admet une période (c'est un motif qui se répète, en théorie des langages).

- Un nombre est premier si tous ses mots sont apériodiques.

Pourquoi la moyenne des parties fractionnaires des parties réelles des zéros de zêta semble-t-elle être asymptotiquement égale à 1/2 ? (Denise Vella-Chemla, 7.3.2018)

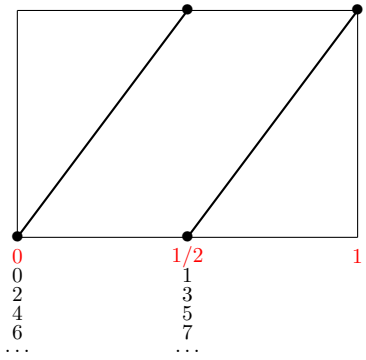
On voudrait fournir ici une explication informelle à une constatation effectuée récemment et qui est que la moyenne des parties fractionnaires des parties réelles des zéros de la fonction zêta semble tendre vers 1/2.

On utilise pour illustrer cette explication la notion de fonction ou signal en dents-de-scie dont on peut trouver une présentation dans les pages wikipedia en français (https://fr.wikipedia.org/wiki/Signal$_e n_d ents_d e_s cie$) ou en anglais (https://en.wikipedia.org/wiki/Sawtooth$_w ave$).

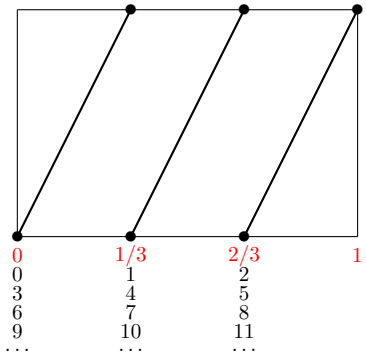
On pense au Snurpf, système de Numération par les Restes (modulaires de Gauss) dans les Parties Finies de $\mathbb{N}$. On conçoit aisément que les restes des divisions euclidiennes des nombres entiers successifs par 2 (en commençant par l'entier 1) sont 1, 0, 1, 0, 1, etc. De même les restes des divisions euclidiennes des entiers successifs par 3 sont 1, 2, 0, 1, 2, 0, 1, 2, 0, etc.

Pour représenter ces *fonctions croissantes "restes de divisions euclidiennes"* ainsi décrites (et qui retombent régulièrement à 0) , on se place sur l'intervalle $[0, 1]$ et on utilise des fonctions en dents-de-scie.

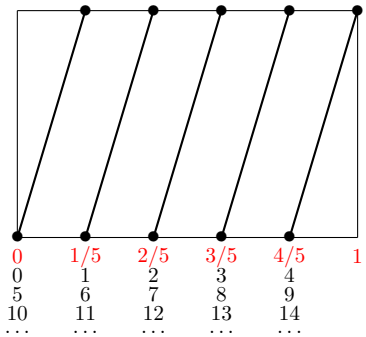
Voici la fonction en dents-de-scie qui permet de visualiser la notion de parité. Les entiers impairs ont pour image le point médian, les pairs ont pour image 0.



Voici la fonction en dents-de-scie qui permet de visualiser la notion de divisibilité par 3. Les entiers qui ont pour reste 1 dans la division euclidienne par 3 ont pour image le point 1/3, ceux qui ont pour reste 2 ont pour image le point 2/3, et les nombres divisibles par 3 ont pour image 0.



Voici la fonction en dents-de-scie qui permet de visualiser la notion de divisibilité par 5. Les nombres qui ont pour reste r dans la division euclidienne par 5 ont pour image $r/5$.



Comme les fonctions en dents-de-scie sont périodiques, il faut les imaginer sur un tore (ou du moins un cylindre). Sur les bords verticaux des représentations planes du cylindre ci-dessus vont se retrouver tous les nombres multiples d'un nombre donné, ainsi que le nombre en question. La périodicité qui fait que la fonction attribue la même image à deux nombres dont la différence vaut 1 s'exprime par la condition $x(t) = t \ (mod\ 1)$.

Il faut imaginer les nombres qui ne sont pas divisibles (par 3 par exemple) comme positionnés sur les lignes verticales de part et d'autre de la droite $x = 1/2$. Les nombres premiers doivent être équitablement répartis entre ces deux ordonnées de localisation (les densités associés à de tels sous-ensembles de nombres dans $\mathbb{N}$ doivent être égales). On imagine qu'on peut trouver une fonction qui envoie les entiers qui ont pour reste 1 dans une division euclidienne par 3 sur le "lieu" $x = 1/3$ et qui envoie les entiers qui ont pour reste 2 dans une division euclidienne par 3 sur le "lieu" $x = 2/3$.

On peut "mélanger" ou "agréger" toutes les fonctions en dents-de-scie en une seule fonction par la fonction inverse de la décomposition en série de Fourier. Du fait de l'équilibre des restes des divisions euclidiennes, les nombres doivent se répartir équitablement de part et d'autre de la droite $x = 1/2$. L'article wikipedia en anglais fournit comme renseignement supplémentaire : *"a sawtooth wave's sound is harsh and clear and its spectrum contains both even and odd harmonics of the fundamental frequency. Because it contains all the integer harmonics, it is one of the best waveforms to use for subtractive synthesis of musical sounds, particularly bowed string instruments like violins and cellos, since the slip-stick behavior of the bow drives the strings with a sawtooth-like motion."*.

Moyenne des parties fractionnaires des parties réelles des zéros de zêta (Denise Vella-Chemla, 4.3.2018)

On calcule par le programme python suivant les moyennes des parties fractionnaires des parties réelles des zéros de zêta qui sont fournies par Odlyzko ici http://www.dtc.umn.edu/ odlyzko/zeta$_t$*ables*/*index.html*.

```
import math
from math import *

zeros=[]
with open('leszerosdezeta', 'r') as f:
    for p in f.readlines():
        z = float(p.split()[1])
        zeros.append(z)
f.close()
print('')
somme=0.0
for i in range(0,100000):
    res = zeros[i]-floor(zeros[i])
    somme=somme+res
    print(res)
print('moyenne')
print(somme/i)
```

Pour le fichier contenant 100000 zéros, la moyenne obtenue est : 0.499356115471.

Pour le fichier des zéros autour de 10^{12}, la moyenne obtenue est : 0.499758301944.

Pour le fichier des zéros autour de 10^{21}, la moyenne obtenue est : 0.496515278444.

Pour le fichier des zéros autour de 10^{22}, la moyenne obtenue est : 0.4983445377.

Pour le très gros fichier fourni par Odlyzko et contenant 2001053 zéros de zêta, la moyenne obtenue est : 0.500277516477.

On refait quelques tests, en les complétant de tests sur un fichier de nombres aléatoires de l'intervalle $[0, 1]$.

On obtient pour le fichier contenant les 2001053 zéros fournis par Odlyzko :
- moyenne = 0.50027726647,
- médiane = 0.499993778489,
- écart-type = 0.288607100069.

On obtient pour le fichier contenant des nombres aléatoires :
- moyenne = 0.500009450015,
- médiane = 0.499661660179,
- écart-type = 0.28860413121.

```
import math
from math import *
import numpy
from numpy import *
import numpy.random

tabzeros=fromfile('leszerostresgrosfichier',dtype=float,count=-1,sep=' ')
tab2=numpy.random.random(tabzeros.size)
print('')
print(tabzeros.ndim)
print(tabzeros.size)
for i in range(tabzeros.size):
    tabzeros[i]=tabzeros[i]-floor(tabzeros[i])
print('tab1')
print(mean(tabzeros))
print(median(tabzeros))
print(std(tabzeros))
print('tab2')
print(mean(tab2))
print(median(tab2))
print(std(tab2))
```

$$exp(li(\kappa.t.i.o(n))).exp(e^{\partial i.t.i^{\nu e}})$$

Les zéros de zêta ne peuvent pas être ailleurs que sur la droite critique parce que sinon, admettons qu'il y en ait à peine 2 qui ne soient pas sur la droite des nombres de partie réelle $\frac{1}{2}$, que l'un soit de la forme $\rho = \alpha + \beta i$ et que son conjugué soit de la forme $\alpha - \beta i$, quand, dans la formule de Riemann

$$f(x) = Li(x) - \sum_{\rho}(Li(x^{\rho}) + Li(x^{\overline{\rho}})) + \int_{x}^{\infty} \frac{du}{u(u^2-1)ln\ u} - ln\ 2,$$

on ajouterait les logarithmes intégrals de ces 2 zéros conjugués, alors que lorsque les zéros appartiennent bien à la droite critique, les parties imaginaires s'annulent entraînant l'ajout seulement du double de la partie réelle commune aux deux zéros, là, pour ces deux zéros hors droite critique, on obtiendrait comme valeur de la somme des logarithmes intégrals de ces deux complexes un nombre complexe, et alors on serait obligé de conclure par une phrase du style "jusqu'à 3 000 456 278, il y a $5\ 678 + 8\ 528i$ nombres premiers, ce qui, on l'avouera, ne veut strictement rien dire". Nous ne voyons pas d'autre explication...

Primalité et zéros de sommes de cosinus

Denise Vella-Chemla

10/7/14

L'article d'Euler *Découverte d'une loi tout extraordinaire des nombres par rapport à la somme de leurs diviseurs* est magique. On reste subjugué par la manière dont le mathématicien a trouvé la formule récurrente de la somme des diviseurs.

On peut trouver sur un forum de mathématiques [1] une autre formule :

$$\sigma(n) = \sum_{k=1}^{n} \sum_{l=1}^{k} cos\left(\frac{2\pi nl}{k}\right)$$

Les cosinus se comportent ici comme des booléens qui comptent pour chaque diviseur sa valeur comme une somme de 1, de façon analogue à la fonction *Succ* de l'arithmétique de Peano.

Du coup, on en déduit une manière rigolote de trouver les nombres premiers : ce sont les zéros de la fonction *sumsumcos* ci-dessous :

$$sumsumcos(n) = \sum_{k=2}^{n-1} \sum_{l=1}^{k} cos\left(\frac{2\pi nl}{k}\right)$$

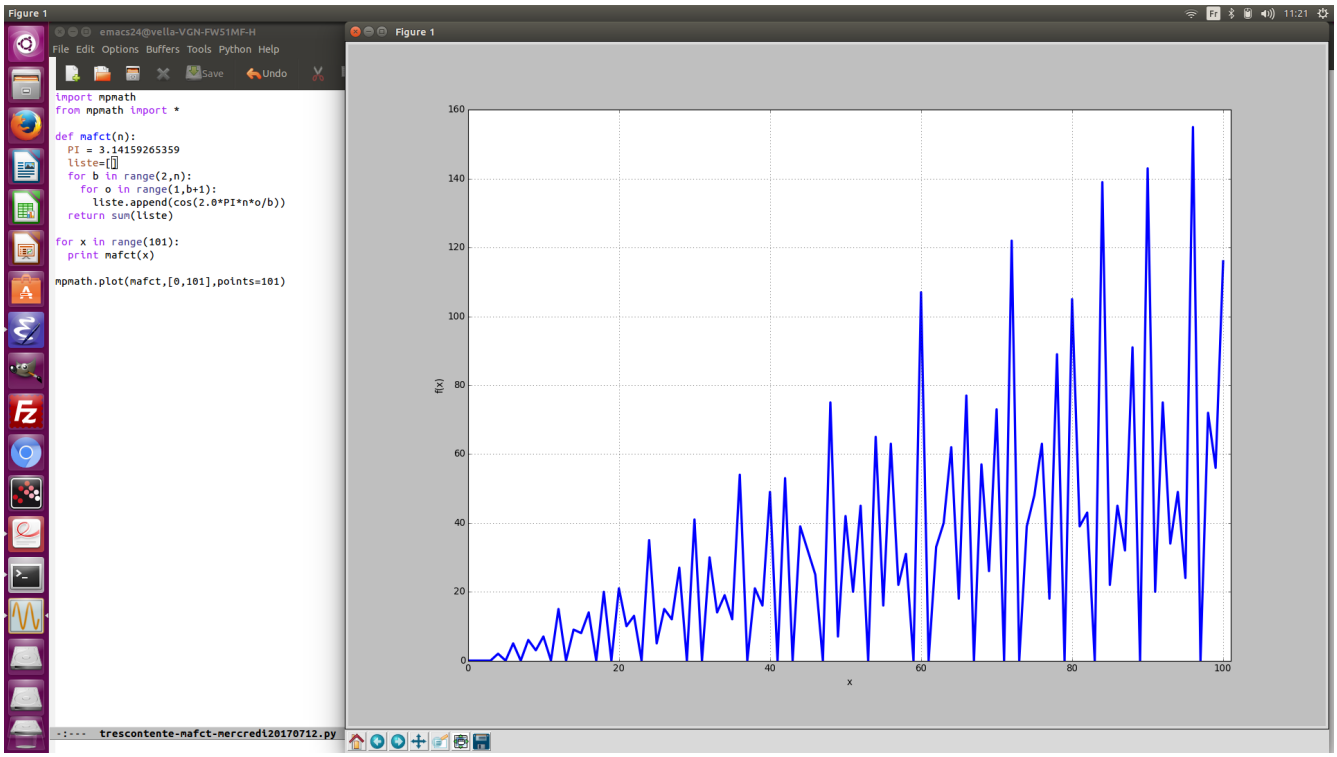
For me, that's F_{un} !

1. à l'adresse $http://www.les-mathematiques.net/phorum/read.php?5,892412,892412$.

Alterner les termes de la somme de cosinus qui s'annule pour les nombres premiers (Denise Vella-Chemla, 28.10.2018)

On a proposé en juillet 2014 * la caractérisation suivante des nombres premiers (motivée par le fait que si p est un nombre premier, $\sigma(p) = p + 1$ †) :

$$n \text{ est premier } \iff \sum_{b=2}^{n-1} \sum_{o=1}^{b} cos \frac{2\pi no}{b} = 0$$



On trouve ici une démonstration du fait que cette caractérisation des nombres premiers en est bien une : http://denise.vella.chemla.free.fr/VictorVarinKeldyshSumsumcos.pdf.

Il s'agit d'une caractérisation triviale des nombres premiers dans le sens où elle calcule la somme des diviseurs des entiers successifs par le biais de calcul d'angles et par le test du fait que ces angles sont ou non multiples de 2π. Cela revient au même pour tester si un nombre est premier qu'à étudier sa division par tous les entiers qui lui sont inférieurs ‡.

Voyons comment la somme de cosinus calcule la somme des diviseurs de 2, 3, 4 et 5. On note les angles en degrés pour que la divisibilité se voie mieux. θ dénote les angles dont sont calculés les cosinus.

∗. cf http://denise.vella.chemla.free.fr/primesommecos.pdf.
†. $\sigma(n)$ est la notation habituelle pour la somme des diviseurs de n.
‡. Cf page 9 de la première note qu'on avait écrite au sujet de la conjecture de Goldbach en septembre 2005.

n	$o,b \to \theta$	cos	$o,b \to \theta$	cos	$o,b \to \theta$	cos	$o,b \to \theta$	cos	$o,b \to \theta$	cos	$\sigma(n)$
2	$1,1 \to 720$	1									
	$1,2 \to 360$	1	$2,2 \to 720$	1							3
3	$1,1 \to 1080$	1									
	$1,2 \to 540$	-1	$2,2 \to 1080$	1							
	$1,3 \to 360$	1	$2,3 \to 720$	1	$3,3 \to 1080$	1					4
4	$1,1 \to 1440$	1									
	$1,2 \to 720$	1	$2,2 \to 1440$	1							
	$1,3 \to 480$	-0.5	$2,3 \to 960$	-0.5	$3,3 \to 1440$	1					
	$1,4 \to 360$	1	$2,4 \to 720$	1	$3,4 \to 1080$	1	$4,4 \to 1440$	1			7
5	$1,1 \to 1800$	1									
	$1,2 \to 900$	-1	$2,2 \to 1800$	1							
	$1,3 \to 600$	-0.5	$2,3 \to 1200$	-0.5	$3,3 \to 1800$	1					
	$1,4 \to 450$	0	$2,4 \to 900$	-1	$3,4 \to 1350$	0	$4,4 \to 1800$	1			
	$1,5 \to 360$	1	$2,5 \to 720$	1	$3,5 \to 1080$	1	$4,5 \to 1440$	1	$5,5 \to 1800$	1	6

On constate par programme qu'en alternant les signes $+$ et $-$ devant chaque terme de la somme et en soustrayant 1 au résultat global, on obtient que les nombres premiers de la forme $4k+1$ ont pour image 0 quand les nombres premiers de la forme $4k+3$ ont pour image 1 selon le programme suivant :

```
import mpmath
from mpmath import *

def mafct(n):
  oppose = 1
  liste=[]
  for b in range(2,n):
    for o in range(1,b+1):
      oppose = (-1) * oppose
      liste.append(oppose*cos(2*pi*n*o/b))
  res=sum(liste)-1
  return res

for x in range(101):
  print(str(x)+' a pour somme '+str(mafct(x)))

mpmath.plot(mafct,[0,101],points=101)
```

Résultat du programme ci-dessus : calcul d'une somme alternée de cosinus

```
1 a pour somme -1
2 a pour somme -1
3 a pour somme 1.0
4 a pour somme -2.0
5 a pour somme -2.66453525910038e-15
6 a pour somme -5.0
7 a pour somme 0.999999999999997
8 a pour somme -2.0
9 a pour somme 6.0
10 a pour somme -5.00000000000001
```

```
11 a pour somme 1.00000000000002
12 a pour somme -10.0
13 a pour somme 1.86517468137026e-14
14 a pour somme -4.99999999999998
15 a pour somme 16.9999999999999
16 a pour somme -2.00000000000003
17 a pour somme 1.06581410364015e-14
18 a pour somme -17.0
19 a pour somme 1.00000000000006
20 a pour somme -10.0000000000001
21 a pour somme 20.0
22 a pour somme -5.00000000000012
23 a pour somme 1.00000000000001
24 a pour somme -18.0000000000001
25 a pour somme 9.99999999999995
26 a pour somme -4.99999999999988
27 a pour somme 24.9999999999999
28 a pour somme -10.0000000000001
29 a pour somme -8.21565038222616e-14
30 a pour somme -36.9999999999999
31 a pour somme 1.00000000000026
32 a pour somme -2.00000000000002
33 a pour somme 27.9999999999999
34 a pour somme -4.99999999999991
35 a pour somme 25.0
36 a pour somme -33.9999999999997
37 a pour somme -3.19744231092045e-14
38 a pour somme -5.00000000000025
39 a pour somme 33.0000000000003
40 a pour somme -17.9999999999999
41 a pour somme 1.75859327100625e-13
42 a pour somme -45.0
43 a pour somme 1.0000000000002
44 a pour somme -9.9999999999999
45 a pour somme 64.0000000000001
46 a pour somme -5.00000000000072
47 a pour somme 1.00000000000015
48 a pour somme -34.0000000000001
49 a pour somme 14.0000000000003
50 a pour somme -24.9999999999995
51 a pour somme 40.9999999999997
52 a pour somme -9.99999999999957
53 a pour somme 1.93622895494627e-13
54 a pour somme -52.9999999999998
55 a pour somme 33.0000000000002
56 a pour somme -17.9999999999994
57 a pour somme 43.9999999999999
58 a pour somme -4.99999999999929
59 a pour somme 0.999999999999813
60 a pour somme -73.9999999999995
61 a pour somme -5.10924635932497e-13
62 a pour somme -4.99999999999921
63 a pour somme 81.0
64 a pour somme -2.00000000000007
65 a pour somme 35.999999999999
66 a pour somme -60.9999999999997
67 a pour somme 1.00000000000103
68 a pour somme -10.0000000000005
69 a pour somme 51.9999999999997
70 a pour somme -53.0000000000017
```

```
71 a pour somme 0.999999999999956
72 a pour somme -65.9999999999991
73 a pour somme 5.77315972805081e-13
74 a pour somme -4.99999999999963
75 a pour somme 97.0
76 a pour somme -10.0000000000007
77 a pour somme 36.0000000000004
78 a pour somme -68.9999999999998
79 a pour somme 1.00000000000032
80 a pour somme -33.9999999999996
81 a pour somme 78.0000000000006
82 a pour somme -5.00000000000226
83 a pour somme 0.999999999998838
84 a pour somme -89.9999999999989
85 a pour somme 43.9999999999991
86 a pour somme -4.99999999999942
87 a pour somme 64.9999999999984
88 a pour somme -18.0000000000019
89 a pour somme -5.15143483426073e-14
90 a pour somme -132.999999999998
91 a pour somme 41.0000000000008
92 a pour somme -9.99999999999778
93 a pour somme 67.9999999999987
94 a pour somme -5.00000000000054
95 a pour somme 48.9999999999985
96 a pour somme -65.9999999999998
97 a pour somme 1.24611432283928e-12
98 a pour somme -32.9999999999993
99 a pour somme 112.999999999998
100 a pour somme -50.0000000000001
```

Si l'on initialise le signe du premier terme de la somme à -1 plutôt qu'à $+1$ et si l'on ajoute 2 à la somme globale plutôt que de lui soustraire 1, alors les rôles des nombres premiers de la forme $4k+1$ et $4k+3$ sont échangés, les premiers ayant alors pour image 1 au lieu de 0 et les seconds ayant pour image 0 au lieu de 1 selon le programme, les graphiques et le tableau des images par les deux fonctions sommes alternées de cosinus suivants :

```
import mpmath
from mpmath import *

def mafct(n):
  oppose = -1
  liste=[]
  for b in range(2,n):
    for o in range(1,b+1):
      oppose = (-1) * oppose
      liste.append(oppose*cos(2*pi*n*o/b))
  res=sum(liste)-1
  return res

for x in range(101):
  print(str(x)+' a pour somme '+str(mafct(x)))

mpmath.plot(mafct,[0,101],points=101)
```

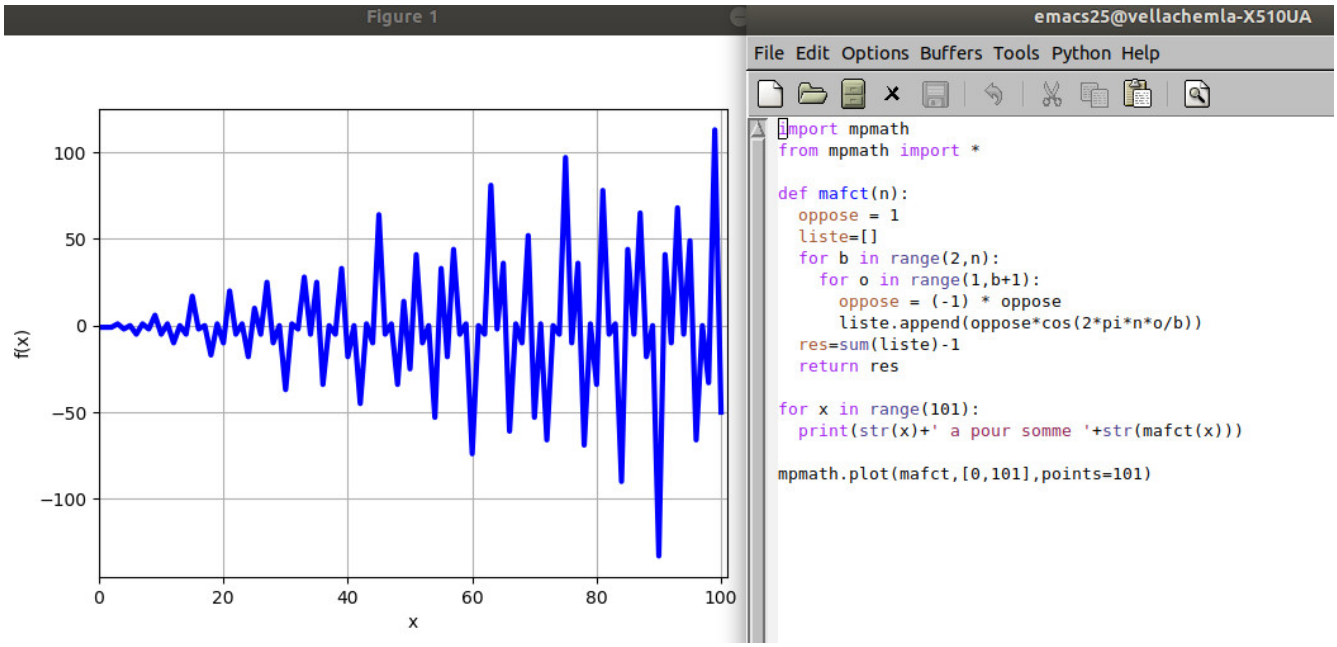


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import mpmath
from mpmath import *
def mafct(n):
oppose = 1
liste=[]
for b in range(2,n):
for o in range(1,b+1):
oppose = (-1) * oppose
liste.append(oppose*cos(2*pi*n*o/b))
res=sum(liste)-1
return res
for x in range(101):
print(str(x)+' a pour somme '+str(mafct(x)))
mpmath.plot(mafct,[0,101],points=101)
f(x)
x
100
50
0
−50
−100
0
20
40
60
80
100

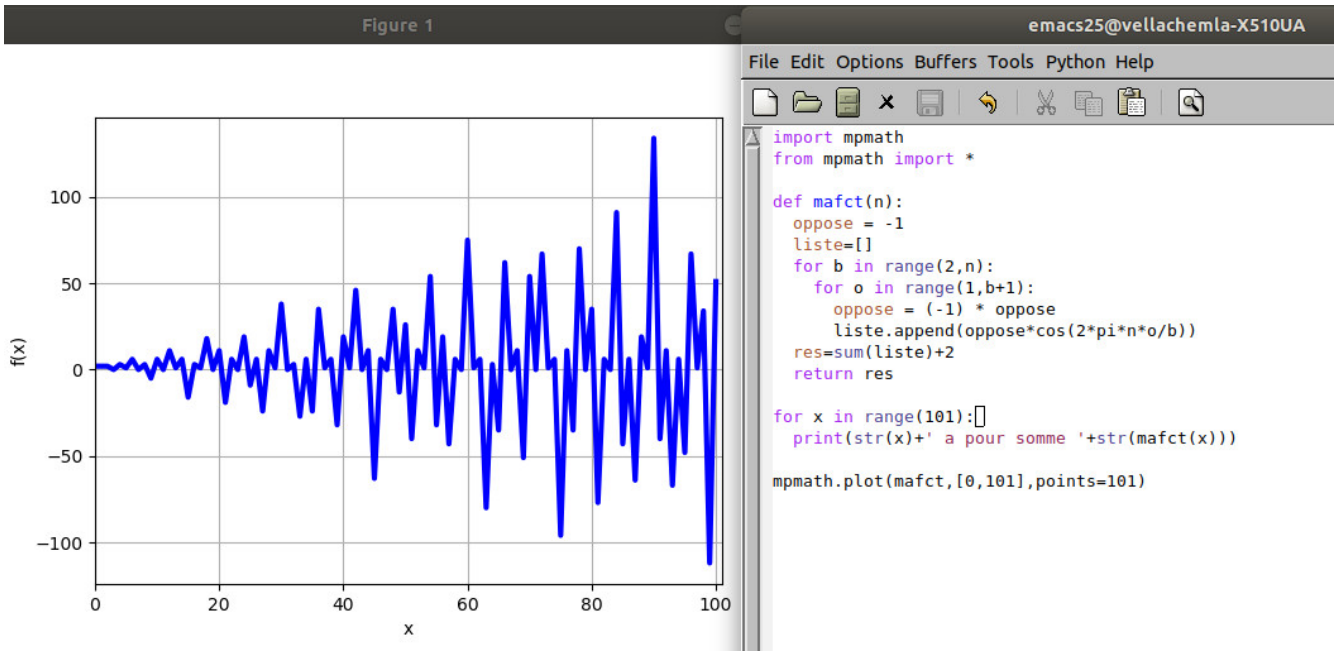


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import mpmath
from mpmath import *
def mafct(n):
oppose = -1
liste=[]
for b in range(2,n):
for o in range(1,b+1):
oppose = (-1) * oppose
liste.append(oppose*cos(2*pi*n*o/b))
res=sum(liste)+2
return res
for x in range(101):
print(str(x)+' a pour somme '+str(mafct(x)))
mpmath.plot(mafct,[0,101],points=101)
f(x)
x
100
50
0
−50
−100
0
20
40
60
80
100

p	*somme alternée 1*	*somme alternée 2*
3	1	0
5	0	1
7	1	0
11	1	0
13	0	1
17	0	1
19	1	0
23	1	0
29	0	1
31	1	0
37	0	1
41	0	1
43	1	0
47	1	0
53	0	1
59	1	0
61	0	1
67	1	0
71	1	0
73	0	1
79	1	0
83	1	0
89	0	1
97	0	1

Annexe : extrait de la première note de septembre 2005 (il faut plutôt prendre le complexe 1 comme sommet commun des polygones)

Au tout début, nous réfléchissions à une manière élégante d'implémenter les horloges modulaires Gaussiennes. On peut voir l'horloge modulaire de n comme un polygone régulier à n côtés sur le cercle unité. Prenons comme convention que tous les polygones ont en commun le sommet correspondant à midi. Deux nombres sont premiers entre eux si leurs polygones réguliers respectifs n'ont aucun sommet commun hormis le sommet midi. Cette idée des polygones réguliers nous a fait faire un détour par les fractions à coefficients entiers. 4 n'est pas premier car $2/4 = 1/2$. Cela nous a amenée naturellement a nous rendre compte qu'un nombre était premier si toutes les fractions de $1/n$ à $(n-1)/n$ étaient non réductibles.

La considération des fractions entières $1/5$, $2/5$, $3/5$, $4/5$, nous a fait dériver vers les sinusoïdes. En effet, les sinusoïdes sont des fonctions qui passent régulièrement par zéro. La sinusoïde $\sin(5\pi x)$ s'annule justement pour les 4 fractions qui nous intéressent sur l'intervalle $]0,1[$. Un nombre n est ainsi premier si sa sinusoïde s'annule exactement $n-1$ fois dans l'intervalle $]0,1[$ et ce, jamais sur un point pour lequel s'annule la sinusoïde d'un nombre premier inférieur à lui.

Nous avons vite abandonné cette voie de recherche : le fait d'assimiler un nombre premier p à sa sinusoïde $\sin(p\pi x)$ semblait ne pas présenter d'intérêt ; en effet, même si cela a l'avantage de restreindre l'étude à l'intervalle $]0,1[$, dans la mesure où il y a une infinité de sinusoïdes qui s'annulent dans cet intervalle, on ne fait que transformer un problème sur des données infiniment grandes en un problème sur des données infiniment petites.

Voir également http://denisevellachemla.eu/dents-de-scie.pdf pour une tentative d'explication d'une valeur moyenne de $\frac{1}{2}$ pour les restes modulaires vus comme des fractions rationnelles.

Interrupteurs (Denise Vella-Chemla, 31.10.2018)

A la suite d'une note publiée en juillet 2014 [1], on propose la fonction somme alternée suivante qui associe $\frac{1}{2}$ aux nombres premiers et des valeurs différentes de $\frac{1}{2}$ aux nombres composés :

$$f_D(n) = \sum_{k=2}^{n-1} \sum_{l=1}^{k} \left(cos\left(\frac{2\pi nl}{k}\right) \times (-1)^{\frac{k^2-k-2+2l}{2}} \right) - 1 + \left((-1)^{\frac{n}{2}} \times \frac{1}{2} \right)$$

Les cosinus d'angles opposés s'éliminent "presque tous" pour les nombres premiers, du fait de leur insécabilité.

Au contraire, pour les nombres composés, leur divisibilité par des nombres qui leur sont inférieurs entraîne par l'ajout des cosinus l'ajout d'un certain nombre d'unités.

Cette propriété rend la somme des cosinus :

- égale à 2 pour les nombres premiers de la forme $4k + 3$,
- et égale à 1 pour les nombres premiers de la forme $4k + 1$.

Le fait d'ajouter en dernier lieu au résultat $-1 + (-1)^{\frac{n}{2}}$ à la somme de cosinus permet de "ramener" les images des nombres premiers égales à 2 ou 1 sur l'image $\frac{1}{2}$.

1. http://denise.vella.chemla.free.fr/primesommecos.pdf

Programme en C++ pour les sommes alternées de cosinus (les $4k+3$ ont pour image 0 et les $4k+1$ ont pour image 1)

```
#include <iostream>
#include <complex>
#include <cmath>
#include <stdio.h>

using namespace std ;
typedef complex<double> dcomp ;

int prime(int atester)
{
  unsigned long diviseur=2;
  unsigned long k = 2;
  bool pastrouve = true ;

  if (atester == 1) return 0;
  if (atester == 2) return 1;
  if (atester == 3) return 1;
  if (atester == 5) return 1;
  if (atester == 7) return 1;
  while (pastrouve)
    {
    if ((k * k) > atester) return 1;
    else
      if ((atester % k) == 0) return 0 ;
      else k++;
    }
}

int main (int argc, char* argv[])
{
  int n, i, j ;
  double somme ;
  int oppose ;
  const double PI = 4.0 * atan(1.0);

  for (n = 2 ; n <= 50 ; n++)
    {
      oppose = 1 ;
      somme = 0.0 ;
      for (i = 2 ; i <= n-1 ; i++)
  {
    std::cout << "\n" ;
    for (j = 1 ; j <= i ; j++)
      {
        oppose = (-1) * oppose ;
        somme += oppose * cos(2.0 * PI * (double) n * (double) j / (double) i) ;
        std::cout << n << "," << i << "," << j << " -> " ;
        std::cout << 360 * (double) n * (double) j / (double) i << " " ;
        std::cout << oppose * cos(2.0 * PI * (double) n * (double) j / (double) i) << "\n" ;
      }
  }
      somme = somme-1-oppose*0.5 ;
      std::cout << n << " somme globale " << somme << "\n" ;
    }
}
```

```
3,2,1 -> 540  1
3,2,2 -> 1080  1
3 somme globale 0.5

4,2,1 -> 720  -1
4,2,2 -> 1440  1

4,3,1 -> 480  0.5
4,3,2 -> 960  -0.5
4,3,3 -> 1440  -1
4 somme globale -1.5

5,2,1 -> 900  1
5,2,2 -> 1800  1

5,3,1 -> 600  0.5
5,3,2 -> 1200  -0.5
5,3,3 -> 1800  -1

5,4,1 -> 450  3.06162e-16
5,4,2 -> 900  1
5,4,3 -> 1350  -2.69484e-15
5,4,4 -> 1800  -1
5 somme globale 0.5

6,2,1 -> 1080  -1
6,2,2 -> 2160  1

6,3,1 -> 720  -1
6,3,2 -> 1440  1
6,3,3 -> 2160  -1

6,4,1 -> 540  -1
6,4,2 -> 1080  -1
6,4,3 -> 1620  -1
6,4,4 -> 2160  -1

6,5,1 -> 432  0.309017
6,5,2 -> 864  0.809017
6,5,3 -> 1296  -0.809017
6,5,4 -> 1728  -0.309017
6,5,5 -> 2160  1
6 somme globale -5.5

7,2,1 -> 1260  1
7,2,2 -> 2520  1

7,3,1 -> 840  0.5
7,3,2 -> 1680  -0.5
7,3,3 -> 2520  -1

7,4,1 -> 630  -4.28626e-16
7,4,2 -> 1260  1
7,4,3 -> 1890  -4.90478e-16
7,4,4 -> 2520  -1

7,5,1 -> 504  -0.809017
7,5,2 -> 1008  -0.309017
7,5,3 -> 1512  0.309017
7,5,4 -> 2016  0.809017
7,5,5 -> 2520  1
```

```
7,6,1 -> 420  -0.5
7,6,2 -> 840  -0.5
7,6,3 -> 1260  1
7,6,4 -> 1680  -0.5
7,6,5 -> 2100  -0.5
7,6,6 -> 2520  1
7 somme globale 0.5

8,2,1 -> 1440  -1
8,2,2 -> 2880  1

8,3,1 -> 960  0.5
8,3,2 -> 1920  -0.5
8,3,3 -> 2880  -1

8,4,1 -> 720  1
8,4,2 -> 1440  -1
8,4,3 -> 2160  1
8,4,4 -> 2880  -1

8,5,1 -> 576  -0.809017
8,5,2 -> 1152  -0.309017
8,5,3 -> 1728  0.309017
8,5,4 -> 2304  0.809017
8,5,5 -> 2880  1

8,6,1 -> 480  0.5
8,6,2 -> 960  -0.5
8,6,3 -> 1440  -1
8,6,4 -> 1920  -0.5
8,6,5 -> 2400  0.5
8,6,6 -> 2880  1

8,7,1 -> 411.429  -0.62349
8,7,2 -> 822.857  -0.222521
8,7,3 -> 1234.29  0.900969
8,7,4 -> 1645.71  -0.900969
8,7,5 -> 2057.14  0.222521
8,7,6 -> 2468.57  0.62349
8,7,7 -> 2880  -1
8 somme globale -1.5

9,2,1 -> 1620  1
9,2,2 -> 3240  1

9,3,1 -> 1080  -1
9,3,2 -> 2160  1
9,3,3 -> 3240  -1

9,4,1 -> 810  5.51091e-16
9,4,2 -> 1620  1
9,4,3 -> 2430  -3.42963e-15
9,4,4 -> 3240  -1

9,5,1 -> 648  0.309017
9,5,2 -> 1296  0.809017
9,5,3 -> 1944  -0.809017
9,5,4 -> 2592  -0.309017
9,5,5 -> 3240  1
```

```
9,6,1 -> 540  1
9,6,2 -> 1080  1
9,6,3 -> 1620  1
9,6,4 -> 2160  1
9,6,5 -> 2700  1
9,6,6 -> 3240  1

9,7,1 -> 462.857  0.222521
9,7,2 -> 925.714  -0.900969
9,7,3 -> 1388.57  -0.62349
9,7,4 -> 1851.43  0.62349
9,7,5 -> 2314.29  0.900969
9,7,6 -> 2777.14  -0.222521
9,7,7 -> 3240  -1

9,8,1 -> 405  0.707107
9,8,2 -> 810  -5.51091e-16
9,8,3 -> 1215  -0.707107
9,8,4 -> 1620  1
9,8,5 -> 2025  -0.707107
9,8,6 -> 2430  3.42963e-15
9,8,7 -> 2835  0.707107
9,8,8 -> 3240  -1
9 somme globale 6.5

10,2,1 -> 1800  -1
10,2,2 -> 3600  1

10,3,1 -> 1200  0.5
10,3,2 -> 2400  -0.5
10,3,3 -> 3600  -1

10,4,1 -> 900  -1
10,4,2 -> 1800  -1
10,4,3 -> 2700  -1
10,4,4 -> 3600  -1

10,5,1 -> 720  1
10,5,2 -> 1440  -1
10,5,3 -> 2160  1
10,5,4 -> 2880  -1
10,5,5 -> 3600  1


10,6,1 -> 600  0.5
10,6,2 -> 1200  -0.5
10,6,3 -> 1800  -1
10,6,4 -> 2400  -0.5
10,6,5 -> 3000  0.5
10,6,6 -> 3600  1

10,7,1 -> 514.286  0.900969
10,7,2 -> 1028.57  0.62349
10,7,3 -> 1542.86  0.222521
10,7,4 -> 2057.14  -0.222521
10,7,5 -> 2571.43  -0.62349
10,7,6 -> 3085.71  -0.900969
10,7,7 -> 3600  -1
```

```
10,8,1 -> 450  3.06162e-16
10,8,2 -> 900  1
10,8,3 -> 1350  -2.69484e-15
10,8,4 -> 1800  -1
10,8,5 -> 2250  -2.45548e-16
10,8,6 -> 2700  1
10,8,7 -> 3150  -3.91949e-15
10,8,8 -> 3600  -1

10,9,1 -> 400  0.766044
10,9,2 -> 800  -0.173648
10,9,3 -> 1200  -0.5
10,9,4 -> 1600  0.939693
10,9,5 -> 2000  -0.939693
10,9,6 -> 2400  0.5
10,9,7 -> 2800  0.173648
10,9,8 -> 3200  -0.766044
10,9,9 -> 3600  1
10 somme globale -5.5
```

Interrupteurs ou bien une somme alternée de cosinus assez surprenante (Denise Vella-Chemla, 4.11.2018)

A la suite d'une note publiée en juillet 2014 [1], on propose la fonction somme alternée suivante qui associe $\frac{1}{2}$ aux nombres premiers et des valeurs différentes de $\frac{1}{2}$ aux nombres composés :

$$f_D(n) = \sum_{k=2}^{n-1} \sum_{l=1}^{k} \left(cos\left(\frac{2\pi nl}{k}\right) \times (-1)^{\frac{k^2-k-2+2l}{2}} \right) - 1 + \left((-1)^{\frac{n}{2}} \times \frac{1}{2} \right)$$

Les cosinus d'angles opposés s'éliminent "presque tous" pour les nombres premiers, du fait de leur insécabilité.

Au contraire, pour les nombres composés, leur divisibilité par des nombres qui leur sont inférieurs entraîne par l'ajout des cosinus l'ajout d'un certain nombre d'unités.

Cette propriété rend la somme des cosinus :

- égale à 2 pour les nombres premiers de la forme $4k+3$,
- et égale à 1 pour les nombres premiers de la forme $4k+1$.

Le fait d'ajouter en dernier lieu au résultat $-1+\left((-1)^{\frac{n}{2}} \times \frac{1}{2} \right)$ à la somme de cosinus permet de "ramener" les images des nombres premiers égales à 2 ou 1 sur l'image $\frac{1}{2}$.

On fait calculer par programme cette somme alternée de cosinus pour les entiers jusqu'à 10000 [2].

Cette somme alternée présente la propriété suivante : pour les puissances de nombres premiers, la fonction $f_D(n)$ coïncide avec des fonctions affines ; ainsi, $f_D(9) = 6.5, f_D(27) = 24.5, f_D(81) = 78.5$, i.e. $f_D(x = 3^k) = x - 2.5$; ou bien $f_D(25) = 10.5, f_D(125) = 60.5, f_D(625) = 310.5, f_D(3125) = 1560.5$, i.e. $f_D(x = 5^k) = \frac{1}{2}x - 2$; etc.

On trouve la formule générale pour les puissances de premiers $x = p^k$: si on note $f_D(x) = ax - b$, a est égal à $\frac{2}{p-1}$ et b est égal à $\frac{3p+1}{2p-2}$.

Pour les nombres dont la factorisation fait intervenir plusieurs nombres premiers différents, on n'arrive pas à dégager de formule générale qui coïnciderait avec la somme alternée de cosinus qu'on a proposée.

On est intrigué par ces résultats et on revient à l'idée initiale qui consistait à calculer la somme suivante :

$$g_D(n) = \sum_{k=2}^{n-1} \sum_{l=1}^{k} cos\left(\frac{2\pi nl}{k}\right)$$

On fait calculer à nouveau par programme cette somme de cosinus pour les entiers jusqu'à 1000 [3].

Les images des puissances de nombres premiers obéissent à la formule :

$$g_D(p^k) = \frac{p^k - p}{p-1}$$

Les images des produits simples de nombres premiers (à la puissance 1) obéissent à la formule :

$$g_D(pq) = p + q$$

1. http://denise.vella.chemla.free.fr/primesommecos.pdf
2. Le programme de calcul de $f_D(n)$ peut être trouvé ici http://denise.vella.chemla.free.fr/trouveformuleoppose.pdf. Le résultat du programme peut être trouvé ici http://denise.vella.chemla.free.fr/affin.pdf
3. Le programme de calcul de $g_D(n)$ peut être trouvé ici http://denise.vella.chemla.free.fr/sumsumcos.pdf. Le résultat du programme peut être trouvé ici http://denise.vella.chemla.free.fr/ressumsumcos.pdf

Par exemple, $g_D(21) = g_D(3 \times 7) = 3 + 7 = 10$ ou bien $g_D(10) = g_D(2 \times 5) = 2 + 5 = 7$.

Le calcul de $g_D(210) = g_D(2\times3\times5\times7) = 2+3+5+7+6+10+14+15+21+35+30+42+70+105 = 365$.

La fonction $g_D(n)$ est ainsi très simple à calculer pour les produits de nombres premiers simples.
Pour les produits faisant intervenir des puissances, on peut appliquer la formule :

$$g_D(p^k \times x) = (p+1)g_D(p^{k-1} \times x).$$

Conjecture de Goldbach, où l'on retrouve ζ autrement (Denise Vella-Chemla, 26.1.2019)

On s'intéresse à la conjecture de Goldbach qui stipule que tout nombre pair supérieur strictement à 2 est la somme de deux nombres premiers.

On rappelle qu'un nombre premier inférieur à $\frac{n}{2}$, qui ne partage aucun de ses restes avec n un nombre pair supérieur à 2, dans toute division par un nombre premier inférieur à $\sqrt{n}$, est un décomposant de Goldbach de n.

En effet, si x inférieur à $\frac{n}{2}$ ne partage aucun de ses restes avec n dans toute division par un nombre premier inférieur à $\sqrt{n}$, alors $n - x$ est lui aussi premier.

La probabilité qu'un nombre x inférieur à $\frac{n}{2}$ soit premier est fournie par le théorème des nombres premiers ; elle vaut :

$$\frac{\frac{n}{2}}{ln\left(\frac{n}{2}\right)}$$

Supposons maintenant que x est premier. Etudions le non-partage d'un reste au moins entre x et n dans les divisions par les nombres premiers inférieurs à $\sqrt{n}$.

Puisque x est premier, on sait au moins qu'il n'a aucun reste nul dans toute division par un nombre premier inférieur à $\sqrt{n}$.

Dans une division par 3, il lui reste 2 possibilités de reste (1 et 2), et il a une chance sur deux (i.e. 1/2) d'obtenir l'un ou l'autre.

Dans une division par 5, il lui reste 4 possibilités de reste (1, 2, 3 ou 4), et il a une chance sur 4 (i.e. 1/4) d'obtenir l'un ou l'autre.

Dans une division par 7, il lui reste 6 possibilités de reste (1, 2, 3, 4, 5 et 6), et il a une chance sur 6 (i.e. 1/6) d'obtenir l'un ou l'autre.

Plus généralement, dans une division par p, il lui reste $p-1$ possibilités de reste (1, 2, ..., $p-1$), et il a une chance sur $p-1$ (i.e. 1/(p-1)) d'obtenir l'un ou l'autre.

Tous ces événements ayant des probabilités indépendantes, la probabilité d'obtenir leur conjonction est le produit des probabilités de chaque événement séparé (les événements considérés étant "x et n ont même reste dans une division par 3", "x et n ont même reste dans une division par 5", etc.).

Ce produit s'écrit :

$$\prod_{p\ premier\ <\sqrt{n}} \frac{1}{p-1}$$

On peut le réécrire :

$$\prod_{p\ premier\ <\sqrt{n}} \frac{1}{p^{-(-1)}-1}$$

puis

$$-\prod_{p\ premier\ <\sqrt{n}} \frac{1}{1-p^{-(-1)}}$$

et l'on reconnaît alors $-\zeta(-1)$. Ramanujan a démontré que $\zeta(-1) = -\frac{1}{12}$. La note [1] fournit une démonstration simple de ce fait.

On obtient donc comme probabilité globale qu'un nombre x soit d'une part premier, et d'autre part ne partage aucun de ses restes avec n dans une division par un nombre premier inférieur à $\sqrt{n}$ [2] :

$$\frac{\frac{n}{2}}{ln\left(\frac{n}{2}\right)} \times (-\zeta(-1))$$

soit :

$$\frac{n}{2\ ln\ n - 2\ ln\ 2} \times \frac{1}{12}.$$

Ceci semble rendre la conjecture de Goldach vraie à partir de $n = 92$ [3].

1. Par définition $S = 1 + 2 + 3 + 4 + 5 + ...$ On remarque qu'en faisant la différence terme à terme :

$$\begin{array}{rccccl} S - B = & 1+2 & +3+4 & +5+6 & ... & \\ & -1+2 & -3+4 & -5+6 & ... & \\ = & 0+4 & +0+8 & +0+12 & ... = 4(1+2+3+...) = 4S & \end{array}$$

Donc $S - 4S = B$, i.e. $-3S = B$, d'où $S = -\frac{B}{3} = -\frac{\frac{1}{4}}{3}$ Ainsi, on retrouve le résultat attendu : $S = -\frac{1}{12}$.

2. Le fait pour x de ne partager aucun reste avec n dans les divisions par les nombres premiers inférieurs à $\sqrt{n}$ n'a rien à voir avec le fait d'être premier à n. Cette condition est nécessaire (i.e. *impliquée*) mais non suffisante (i.e. *impliquante*). Par exemple, 17 et 81, dont la somme vaut 98, sont tous les deux premiers à 98, mais ils n'en sont pas pour autant des décomposants de Goldbach (de 98) puisque 17 partage le reste de 2 avec 98 lorsqu'on les divise par 3 (Gauss écrit cela $17 \equiv 98\ (mod\ 3)$, c'est lui qui a attiré l'attention de tous sur l'importance de travailler dans les corps premiers).

3. $\frac{92}{2\ ln\ 92 - 2\ ln\ 2}.\frac{1}{12} = 1.0012254835$ alors que $\frac{90}{2\ ln\ 90 - 2\ ln\ 2)}.\frac{1}{12} = 0.9851149163$.

Conjecture de Goldbach, où l'on retrouve ζ autrement (Denise Vella-Chemla, 29.5.2019)

On s'intéresse à la conjecture de Goldbach qui stipule que tout nombre pair supérieur strictement à 2 est la somme de deux nombres premiers.

On rappelle qu'un nombre premier inférieur à $\frac{n}{2}$, qui ne partage aucun de ses restes avec n un nombre pair supérieur à 2, dans toute division par un nombre premier inférieur à $\sqrt{n}$, est un décomposant de Goldbach de n.

En effet, si x inférieur à $\frac{n}{2}$ ne partage aucun de ses restes avec n dans toute division par un nombre premier inférieur à $\sqrt{n}$, alors $n-x$ est lui aussi premier.

La probabilité asymptotique qu'un nombre x inférieur à $\frac{n}{2}$ soit premier est fournie par le théorème des nombres premiers ; elle vaut :

$$\frac{\frac{n}{2}}{ln\left(\frac{n}{2}\right)}$$

La minoration de $\pi(k)$ (le nombre de nombres premiers inférieurs à k) par $\frac{k}{ln\ k}$ est fournie dans [1], page 69, pour $x \geq 17$.

Supposons maintenant que x est premier. Etudions les probabilités d'égalité des restes de x et n quand on les divise par les nombres premiers inférieurs à $\sqrt{n}$.

Puisqu'on a supposé x premier, on sait au moins qu'il n'a aucun reste nul lorsqu'on le divise par un nombre premier inférieur à $\sqrt{n}$.

n a un certain reste, lorsqu'on le divise par un nombre premier inférieur à $\sqrt{n}$. x doit "éviter" le reste en question (ne doit pas avoir le même).

Si on considère une division de n par l'un de ses diviseurs premiers p de reste nul, x n'a que ce reste nul (0) à éviter. Or x a déjà évité 0 par le fait qu'il est premier. x a le choix entre $p-1$ restes possibles dans la division par p.

Si on considère une division de n par un nombre premier qui n'est pas l'un de ses diviseurs, n a selon ce nombre premier un reste non-nul que x doit éviter. x a alors le choix entre $p-2$ restes possibles dans sa division par p, qu'il peut avoir à égales probabilités l'un ou l'autre mais on va utiliser le fait que $\frac{1}{p-2} > \frac{1}{p-1}$ et minorer chaque probabilité selon un nombre premier p donné par $\frac{1}{p-1}$, pour homogénéiser les différents cas (considération des diviseurs ou des non-diviseurs de n).

Voyons des exemples, pour fixer les idées : dans une division par 3, on minore le nombre de possibilités par 2 possibilités de reste (1 et 2), et x a une chance sur deux (i.e. 1/2) d'obtenir l'un ou l'autre.

Dans une division par 5, il reste à x 4 possibilités de reste (1, 2, 3 ou 4), et x a une chance sur 4 (i.e. 1/4) d'obtenir l'un ou l'autre.

Dans une division par 7, il reste à x 6 possibilités de reste (1, 2, 3, 4, 5 et 6), et x a une chance sur 6 (i.e. 1/6) d'obtenir l'un ou l'autre.

Plus généralement, dans une division par p, on minore la probabilité que x et n aient le même reste ainsi : il y a $p-1$ possibilités de restes possibles au maximum pour x (qui sont 1, 2, ..., $p-1$), et x a une chance sur $p-1$ (i.e. 1/(p-1)) d'obtenir l'un ou l'autre de ces restes.

Tous ces événements ayant des probabilités indépendantes, la probabilité d'obtenir leur conjonction est le produit des probabilités de chaque événement séparé (les événements considérés étant "x et n ont même

reste dans une division par 3", "x et n ont même reste dans une division par 5", etc.).

Ce produit s'écrit :

$$\prod_{p\ premier\ <\sqrt{n}} \frac{1}{p-1}$$

On peut le réécrire :

$$\prod_{p\ premier\ <\sqrt{n}} \frac{1}{p^{-(-1)}-1}$$

puis

$$-\prod_{p\ premier\ <\sqrt{n}} \frac{1}{1-p^{-(-1)}}$$

On peut étendre ce produit à l'infinité des nombres premiers car en fait, c'est selon tout nombre premier que n et x ne peuvent être congrus, pour que le complémentaire à n de x qu'est $n-x$ soit premier. On reconnaît alors $-\zeta(-1)$ dans le produit proposé pour que x et n aient des restes différents dans un division par un nombre premier quelconque. Ramanujan a démontré que $\zeta(-1) = -\frac{1}{12}$. La note [1] fournit une démonstration simple de ce fait.

On obtient donc comme probabilité globale qu'un nombre x soit d'une part premier, et d'autre part ne partage aucun de ses restes avec n dans une division par un nombre premier inférieur à $\sqrt{n}$ (en fait quelconque) [2] :

$$\frac{\frac{n}{2}}{ln\left(\frac{n}{2}\right)} \times (-\zeta(-1))$$

soit :

$$\frac{n}{2\ ln\ n - 2\ ln\ 2} \times \frac{1}{12}.$$

Ceci semble rendre la conjecture de Goldach vraie à partir de $n = 92$ [3].

Tentative de réécriture mathématique :

On cherche à démontrer que $\forall n$ pair, $\exists x, 3 \leq x \leq n/2$ premier impair tel que $n-x$ est premier aussi.

$$\begin{array}{lll} (1)\quad x \text{ premier} & \iff \forall p \text{ premier} \leq \sqrt{x}, & x \not\equiv 0\ (mod\ p). \\ (2)\quad n-x \text{ premier} & \iff \forall p \text{ premier} \leq \sqrt{n-x}, & n-x \not\equiv 0\ (mod\ p) \\ & \iff \forall p \text{ premier} \leq \sqrt{n-x}, & x \not\equiv n\ (mod\ p). \end{array}$$

On peut remplacer dans (1) la condition $\forall p$ premier $\leq \sqrt{x}$ par la condition plus forte $\forall p$ premier $\leq \sqrt{n/2}$ puisqu'on a posé $x \leq n/2$.

---

1. Par définition $S = 1+2+3+4+5+....$ On remarque qu'en faisant la différence terme à terme :

$$\begin{array}{rccccl} S-B= & 1+2 & +3+4 & +5+6 & ... & \\ & -1+2 & -3+4 & -5+6 & ... & \\ = & 0+4 & +0+8 & +0+12 & ... = 4(1+2+3+...) = 4S & \end{array}$$

Donc $S-4S = B$, i.e. $-3S = B$, d'où $S = -\frac{B}{3} = -\frac{\frac{1}{4}}{3}$ Ainsi, on retrouve le résultat attendu : $S = -\frac{1}{12}$.

2. Le fait pour x de ne partager aucun reste avec n dans les divisions par les nombres premiers inférieurs à $\sqrt{n}$ n'a rien à voir avec le fait d'être premier à n. Cette condition est nécessaire (i.e. *impliquée*) mais non suffisante (i.e. *impliquante*). Par exemple, 17 et 81, dont la somme vaut 98, sont tous les deux premiers à 98, mais ils n'en sont pas pour autant des décomposants de Goldbach (de 98) puisque 17 partage le reste de 2 avec 98 lorsqu'on les divise par 3 (Gauss écrit cela $17 \equiv 98\ (mod\ 3)$, c'est lui qui a attiré l'attention de tous sur l'importance de travailler dans les corps premiers).

3. $\frac{92}{2\ ln\ 92 - 2\ ln\ 2}.\frac{1}{12} = 1.0012254835$ alors que $\frac{90}{2\ ln\ 90 - 2\ ln\ 2)}.\frac{1}{12} = 0.9851149163$.

On minore le nombre de nombres premiers inférieurs à $\frac{n}{2}$ par $\frac{\frac{n}{2}}{log\left(\frac{n}{2}\right)}$.

Il s'agit alors de trouver combien de nombres dans cet ensemble de nombres premiers inférieurs à $\frac{n}{2}$, dont on a le cardinal, partagent leur reste avec n ; le partage d'un reste avec n le nombre pair considéré consiste à "fixer" le reste possible et donc à diminuer le nombre de restes possibles de 1 selon chaque module ; on doit multiplier le cardinal $\pi\left(\frac{n}{2}\right)$ minoré par $\frac{\frac{n}{2}}{log\left(\frac{n}{2}\right)}$ (qui correspond à la condition (1) ci-dessus) par la probabilité qu'il y ait un partage de reste selon chaque nombre premier indépendamment (qui correspond à la condition (2) ci-dessus) et cette probabilité a comme valeur $-\zeta(-1) = \frac{1}{12}$. C'est un cardinal d'ensemble qu'on obtient par ce procédé de multiplication d'un cardinal par une probabilité. Un tel calcul semble faire sens et assure un cardinal d'au moins 1 à partir de 92.

Bibliographie

[1] J. B. Rosser et L. Schoenfeld, *Approximate formulas for some functions of prime numbers*, dedicated to Hans Rademacher for his seventieth birthday, Illinois J. Math., Volume 6, Issue 1 (1962), 64-94.

Annexe à la proposition denitac.pdf (Denise Vella-Chemla, 23.7.2019)

On cherche à décomposer un nombre pair n en somme de 2 nombres premiers $p_1 + p_2$.

On ne peut pas faire référence à $\zeta(-1)$ comme on l'a fait dans [1]. On peut cependant, pour obtenir une minoration du nombre de décomposants de Goldbach de n, utiliser le cardinal $\left|\mathcal{P}_{\frac{n}{2}}\right|$ de l'ensemble des nombres premiers inférieurs ou égaux à $\frac{n}{2}$ et le multiplier par le produit $\prod_{p\leq\sqrt{n}}\left(1-\frac{1}{p}\right)$ qui compte combien de chances a le nombre premier p_1 de ne pas partager son reste avec n selon chaque module p inférieur à $\sqrt{n}$ (le fait de ne pas partager son reste avec n permet à p_1 d'avoir un complémentaire à n (appelé p_2) qui est premier également).

La minoration [1] de $\pi(x)$ (le nombre de nombres premiers inférieurs à x) par $\frac{x}{\log x}$ est fournie dans [2], page 69, pour $x \geq 17$ (Corollaire 1, (3.5), du Théorème 2, dont la démonstration est fournie au paragraphe 7 de [2]).

On a en conséquence $|\mathcal{P}_{\frac{n}{2}}| > \frac{\frac{n}{2}}{\log\left(\frac{n}{2}\right)}$.

La minoration de $\prod_{p\leq\sqrt{n}}\left(1-\frac{1}{p}\right)$ est également fournie dans [2], page 70 (c'est le corollaire (3.27) du Théorème 7 dont la démonstration est fournie au paragraphe 8 de [2], avec γ la constante d'Euler-Mascheroni).

$$(3.27) \qquad \frac{e^{-\gamma}}{\log x}\left(1-\frac{1}{\log^2 x}\right) < \prod_{p\leq x}\left(1-\frac{1}{p}\right) \quad \text{pour } 1 < x.$$

En multipliant ces expressions ensemble, on obtient que le nombre de décomposants de Goldbach de n doit être supérieur à :

$$\frac{n/2}{\log(n/2)}\frac{e^{-\gamma}}{\log\sqrt{n}}\left(1-\frac{1}{\log^2\sqrt{n}}\right)$$

qui est strictement supérieur à 1 à partir de 24.

Bibliographie

[1] http://denisevellachemla.eu/denitac.pdf.

[2] J. B. Rosser et L. Schoenfeld, *Approximate formulas for some functions of prime numbers*, dedicated to Hans Rademacher for his seventieth birthday, Illinois J. Math., Volume 6, Issue 1 (1962), 64-94.

1. Cette minoration est à distinguer du Théorème des nombres premiers, prouvé indépendamment par Hadamard et La Vallée-Poussin, et qui fournit une tendance asymptotique pour $\pi(x)$.

Réécrire

Denise Vella-Chemla (7.12.2019) aidée par Leila Schneps pour la section 1

1. Caractérisation des décomposants de Goldbach d'un nombre pair

Soit n un nombre pair supérieur à 4 et p_k un nombre premier compris entre 3 et $\sqrt{n}$.

Notons $F(p_k, n) = \{m \in \mathbb{N} : m \text{ impair}, \sqrt{n} \leq m \leq n/2, m \neq 0\ [p_k], m \neq n\ [p_k]\}$

Appelons $D(n) = \cap F(p_k, n)$ l'intersection des ensembles $F(p_k, n)$ pour tous les premiers p_k compris entre 3 et $\sqrt{n}$.

Démontrons que $D(n)$ ne contient que des nombres premiers.

Lemme 1 : Soit m un entier positif impair. Si m n'est divisible par aucun nombre premier compris entre 3 et $\sqrt{m}$, alors m est premier.

Démonstration : Supposons que m ne soit pas premier. Alors il existe un nombre premier $p < m$ qui divise m. Mais on sait que p n'est pas compris entre 3 et $\sqrt{m}$, donc $p > \sqrt{m}$. On pose $k = m/p$. On a donc $kp = m$. Si $k \geq \sqrt{m}$, alors puisqu'on a aussi $p > \sqrt{m}$, on obtient $kp > m$, ce qui est impossible. On doit donc avoir $k < \sqrt{m}$. Mais comme tout entier, l'entier k est divisible par un nombre premier $q \leq k$. Comme q divise k et k divise m, on a que q divise aussi m, et comme $k \leq \sqrt{m}$, on a que $q \leq \sqrt{m}$, ce qui contredit notre hypothèse de départ que m n'est divisible par aucun premier $\leq \sqrt{m}$. QED.

Appliquons ce résultat à $D(n)$ pour obtenir que $D(n)$ ne contient que des nombres premiers.

Lemme 2 : $D(n)$ ne contient que des nombres premiers [*].

Démonstration : Soit $m \in D(n)$. Alors m est impair et $m \leq n/2$. On sait par le lemme 1 que si m n'est divisible par aucun premier compris entre 3 et $\sqrt{m}$, alors m est premier. Mais par la définition de $D(n)$, on sait déjà que m n'est divisible par aucun premier compris entre 3 et $\sqrt{n}$, et puisque $m < n$, on a $\sqrt{m} < \sqrt{n}$ et donc a fortiori m n'est divisible par aucun premier compris entre 3 et $\sqrt{m}$, donc par le lemme 1, m est bien premier. QED.

Lemme 3 : Si m appartient à $D(n)$, alors $n - m$ est premier.

Démonstration : On commence par montrer qu'aucun nombre premier p compris entre 3 et $\sqrt{n}$ ne divise $n - m$. En effet, si $n - m$ est divisible par p, alors m est congru à n modulo p, ce qui contredit le fait que m appartient à $D(n)$. Ensuite, on note que puisque $n - m < n$, on a $\sqrt{n - m} < \sqrt{n}$ et donc a fortiori, $n - m$ n'est divisible par aucun premier $\leq \sqrt{n - m}$, donc par le lemme 1, $n - m$ est bien un nombre premier.

Si $D(n)$ est non vide, alors n vérifie la conjecture de Goldbach.

2. Existence d'un décomposant de Goldbach pour tout nombre pair

On a vu que $D(n)$ ne contient que des nombres premiers qui sont décomposants de Goldbach de n. Il faut maintenant démontrer que $D(n)$ est non vide pour que n vérifie la conjecture de Goldbach.

Essayons de comprendre pourquoi $D(n) = \cap F(p_k, n)$ ne peut être vide. On reprend l'écriture initiale qu'on avait choisie, sous forme logique : dire que l'intersection des ensembles de la forme $\{\neg 0_{p_k} \wedge \neg n_{p_k}\}$

∗. si $D(n)$ est vide, le lemme est vrai par vacuité.

Preuve de Victor Varin que les nombres premiers annulent ma somme de somme de cosinus et qu'ils sont les seuls nombres à le faire (Denise Vella-Chemla, 13.7.2017)

$$dn1 = \sigma(n) = \sum_{k=1}^{n}\sum_{l=1}^{k} \cos\frac{2\pi nl}{k}.$$

$\sigma(n)$ est la somme des diviseurs de n.

$\sigma(1) = 1$
$\sigma(2) = 3$
$\sigma(3) = 4$

Maintenant la belle formule ($ssc = sumsumcos$).

$$dn2 = ssc(n) = \sum_{k=2}^{n-1}\sum_{l=1}^{k} \cos\frac{2\pi nl}{k}.$$

$ssc(2) = 0$
$ssc(3) = 0$
$ssc(4) = 2$
$ssc(5) = 0$

On a :

$$\sum_{k=1}^{n}\sum_{l=1}^{k} \cos\frac{2\pi nl}{k} = \sum_{l=1}^{1} \cos 2\pi nl + \sum_{k=2}^{n-1}\sum_{l=1}^{k} \cos\frac{2\pi nl}{k} + \sum_{l=1}^{n} \cos 2\pi l$$

$$\begin{aligned} dn3 \quad &= \sigma(n) = 1 + n + \sum_{k=2}^{n-1}\sum_{l=1}^{k} \cos\frac{2\pi nl}{k} \\ &= 1 + n + ssc(n) \end{aligned}$$

Ce qui est évident pour les nombres premiers.

Maintenant prouvons formellement que les nombres premiers annulent la fonction $ssc(n)$.

On définit $sss(n)$ par :

$$dn4 = \sum_{k=1}^{n}\sum_{l=1}^{k} \sin\frac{2\pi nl}{k}$$

$sss(2) = 0$
$sss(3) = 0$
$sss(4) = 0$
$sss(5) = 0$

Mais nous n'utiliserons pas cette propriété, elle découlera automatiquement.

Maintenant prenons

$v(n) = ssc(n) + i.sss(n)$ avec $i^2 = -1$

$$dn5 = v(n) = \sum_{k=2}^{n-1}\sum_{l=1}^{k} \cos\frac{2\pi nl}{k} + i\left(\sum_{k=2}^{n-1}\sum_{l=1}^{k} \sin\frac{2\pi nl}{k}\right)$$

De façon triviale, $Re(v(n)) = ssc(n)$ quel que soit $sss(n)$! Mais :

$$dn6 = v(n) = \sum_{k=2}^{n-1}\sum_{l=1}^{k} e^{\frac{2i\pi nl}{k}}$$

$v(4) = 2$

Maintenant, considérons la somme intérieure $dn6$:

$$dn7 = sm(n) = \sum_{l=1}^{k} e^{\frac{2i\pi nl}{k}}$$

Maintenant, une ruse importante : oublions temporairement que n est un entier dans $dn7$! !

$$dn7 = sm(n) = \sum_{l=1}^{k} \left(e^{\frac{2i\pi n}{k}}\right)^{l}$$

Et posons :

$e^{\frac{2i\pi n}{k}} = X$

alors

$$dn7 = sm(n) = \sum_{l=1}^{k} X^l = \frac{X(X^k - 1)}{X - 1}$$

Finalement

$$dn7 = sm(n) = \frac{e^{\frac{2i\pi n}{k}}\left(\left(e^{\frac{2i\pi n}{k}}\right)^k - 1\right)}{e^{\frac{2i\pi n}{k}} - 1}$$
$$= \frac{e^{\frac{2i\pi n}{k}}\left(e^{2i\pi n} - 1\right)}{e^{\frac{2i\pi n}{k}} - 1}$$

Ces calculs sont valides si le dénominateur est non nul !

Maintenant, si n est premier, alors $\frac{n}{k}$ est fractionnaire et le dénominateur n'est pas nul, dans la mesure où $exp(2i\pi n) = 1$ pour n entier !

Ainsi la preuve est faite que les nombres premiers n annulent $ssc(n)$. Mais si n est un nombre composé, il faut utiliser une petite ruse : considérons seulement $k|n$ (k divise n) dans $dn6$.

subs(n=d*k, dn7) mais d est légèrement différente d'une valeur entière et alors la formule ci-dessous a du sens :

$$sm(dk) = \frac{e^{2i\pi d}(e^{2i\pi dk} - 1)}{e^{2i\pi d} - 1}$$
$$e^{2i\pi d} = Y$$
$$e^{2i\pi d} = 1$$
$$\frac{Y^k - 1}{Y - 1} = \sum_{j=0}^{k-1} Y^j$$
$$sm(dk) = e^{2i\pi d} \sum_{j=0}^{k-1} \left(e^{2i\pi d}\right)^j$$

On rappelle alors que d est entier et que donc,

$$sm(dk) = k$$

ce qui termine la preuve.

Orthogonalité

- On note $\mathcal{F}(x) = \{y \in \mathbb{N}^\times \mid 3 \leq y \leq x\}$
- On note $98^\perp$ l'ensemble des décomposants de Goldbach de 98.
- $98 \in (2\mathbb{N}) \cap (3\mathbb{N}+2) \cap (5\mathbb{N}+3) \cap (7\mathbb{N})$.
- $98^\perp \in \mathcal{F}(49) \cap [(2\mathbb{N}+1) \cap (3\mathbb{N}+1) \cap [(5\mathbb{N}+1) \cup (5\mathbb{N}+2) \cup (5\mathbb{N}+4)] \cap [(7\mathbb{N}+1) \cup \ldots \cup (7\mathbb{N}+6)]]$.

  Attention : Bien avoir à l'esprit que les règles de développement de la multiplication $\cap$ sur l'addition $\cup$ se font comme habituellement en algèbre, i.e. l'union $(7\mathbb{N}+1) \cup \ldots \cup (7\mathbb{N}+6)$ ne représente pas tous les entiers sauf les multiples de 7 par exemple. On développe par distributivité d'une opération sur l'autre.

- Plus généralement, si on note $\mathcal{P}$ l'ensemble des nombres premiers.
- $n \in \bigcap_{a \in \mathcal{F}(\sqrt{n}) \cap \mathcal{P}^\times, b \in \{0,\ldots,a-1\}} \{ax+b\}$ et
- $n^\perp \in \mathcal{F}(n/2) \cap \bigcap_{a \in \mathcal{F}(\sqrt{n}) \cap \mathcal{P}^\times, b' \in \{1,\ldots,a-1\}} \{ax+b', \text{ avec } b' \neq b, \forall a\}$.

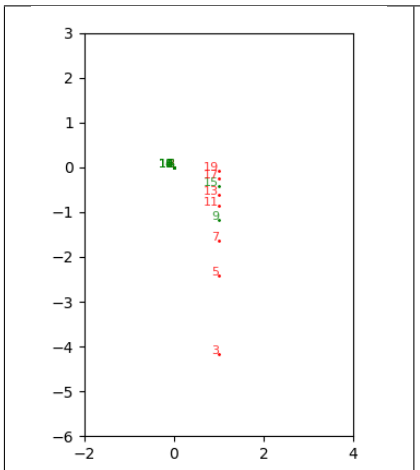


Fig. 1 : $x \leq 20$

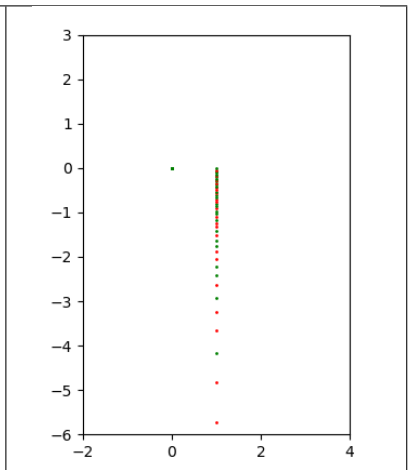


Fig. 2 : $x \leq 100$

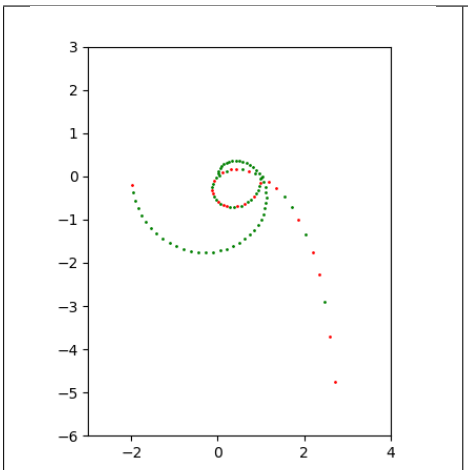


Fig. 3 : 100 nbs, exposants $\leq$ 98

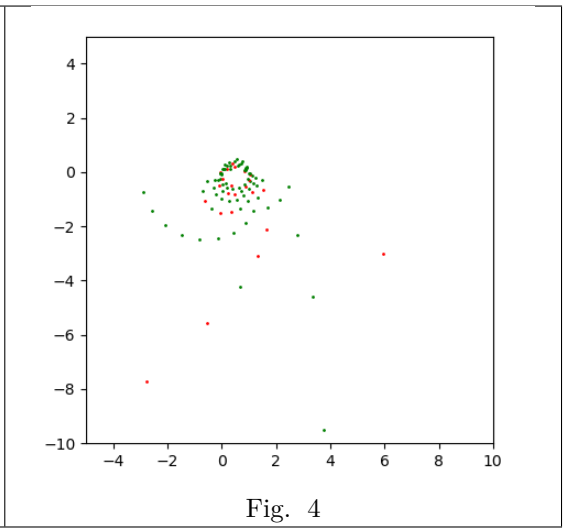


Fig. 4

L'explication de l'alignement de tous les nombres (si ce n'est un point qu'on semble distinguer comme non aligné avec les autres) sur la droite de partie réelle 1 est la suivante :

Puisque $t_k = e^{\frac{-i\pi k}{n}}$ et $z_k = \dfrac{1-t_k^m}{1-t_k}$, avec $k \in [1, n-1]$, si $m = n$, alors on a $t_k^m = e^{-i\pi k}$ et de ce fait,

$$\begin{aligned} z_k &= \frac{1-e^{-i\pi k}}{1-e^{\frac{-i\pi k}{n}}} \\ &= \frac{1-(-1)^k}{1-e^{\frac{-i\pi k}{n}}} \end{aligned}$$

Le dernier numérateur ci-dessus est nul lorsque k est pair.

Si k est impair, on a :

$$\begin{aligned} z_k &= \frac{2}{1-e^{\frac{-i\pi k}{n}}} \\ &= \frac{2\left(1-e^{\frac{i\pi k}{n}}\right)}{\left(1-\cos\ \frac{k\pi}{n}\right)^2+\left(\sin\ \frac{k\pi}{n}\right)^2} \end{aligned}$$

$$
\begin{aligned}
z_k &= \frac{2\left(1-e^{\frac{i\pi k}{n}}\right)}{2-2\cos\ \frac{k\pi}{n}} \\
&= \frac{1-e^{\frac{i\pi k}{n}}}{1-\cos\ \frac{k\pi}{n}} \\
&= \frac{1-\left(\cos\ \frac{k\pi}{n}+i\sin\ \frac{k\pi}{n}\right)}{1-\cos\ \frac{k\pi}{n}} \\
&= \frac{1-\cos\ \frac{k\pi}{n}}{1-\cos\ \frac{k\pi}{n}}+i\frac{\sin\ \frac{k\pi}{n}}{1-\cos\ \frac{k\pi}{n}}
\end{aligned}
$$

qui est de partie réelle constante égale à 1, d'où l'alignement des complexes obtenus comme images des nombres entiers.

La dernière figure ci-dessous montre les nombres pairs mal lisibles et tous collés sur 0. Pour ne pas écraser davantage les nombres les uns sur les autres, on a laissé de côté le nombre 1 qui est bien aligné aussi mais tout en bas à $-32i$.

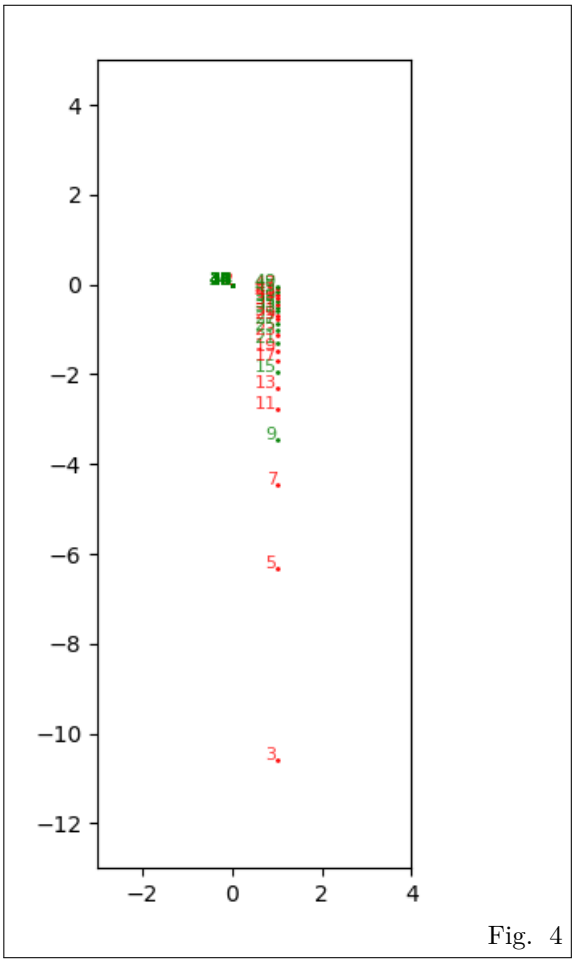


Fig. 4

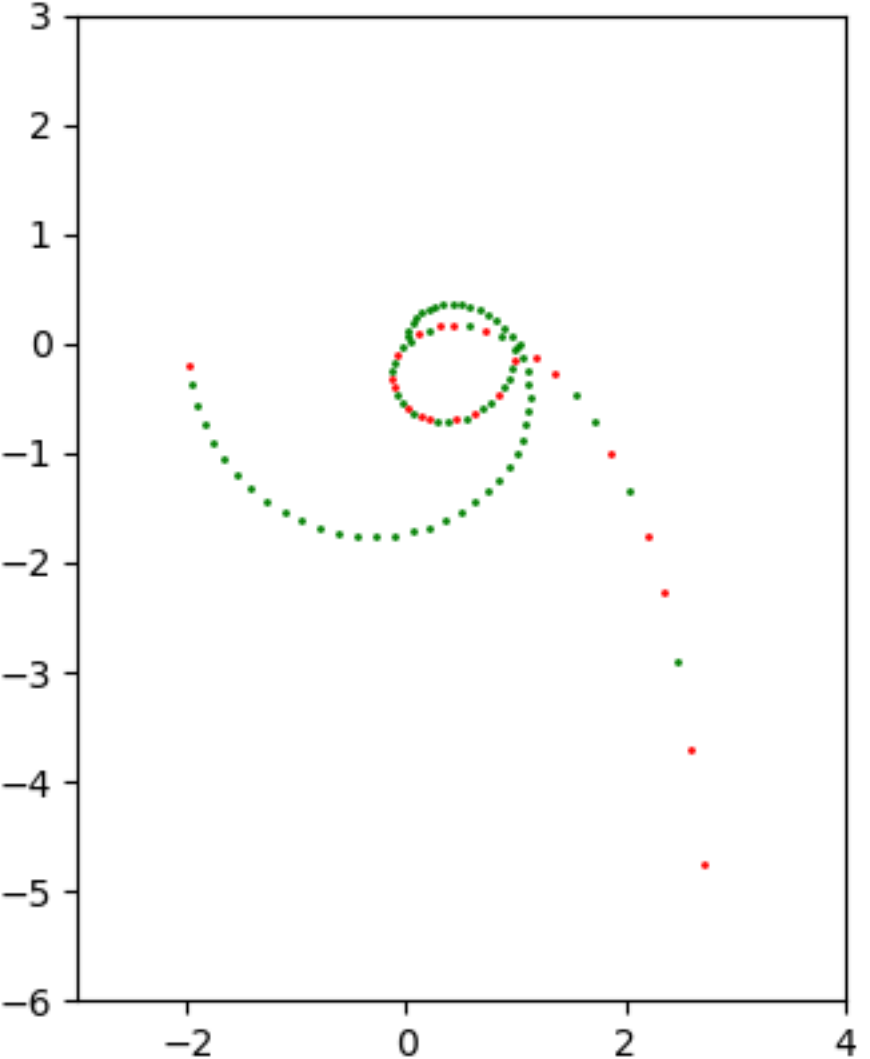
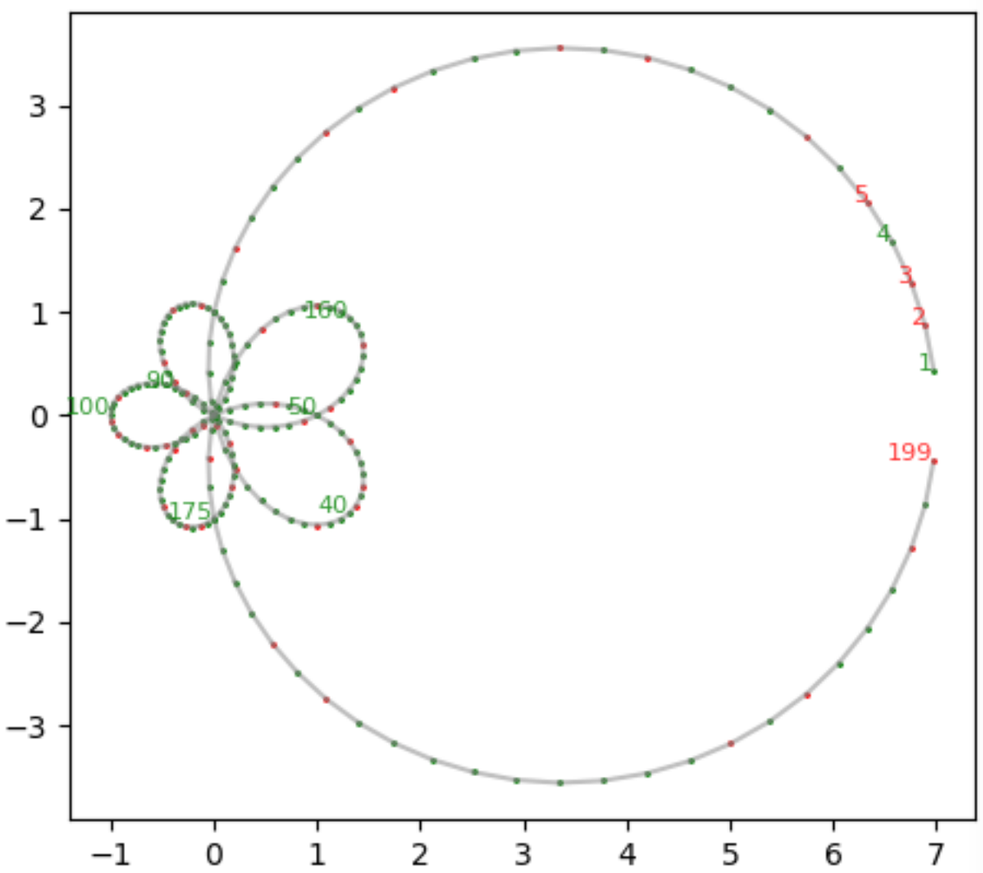

3
2
1
0
−1
−2
−3
−4
−5
−6
−2
0
2
4

Figure 1



emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *

def prime(atester):
  pastrouve = True ; k = 2 ;
  if (atester in [0,1]): return False ;
  if (atester in [2,3,5,7]): return True ;
  while (pastrouve):
    if ((k * k) > atester): return True
    else:
      if ((atester % k) == 0): return False
      else: k=k+1

fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')

n = 100
m = 200
for k in range(1,m):
  t = exp(j*math.pi*k/n)
  z = 1/t+1+t+t*t+t*t*t+t*t*t*t+t*t*t*t*t
  if True:
    c = 'r' if prime(k) else 'g'
    plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
    if (k <= 5) or (k == 90) or (k == 160) or (k == 40) or (k == 100) or (k == 175) or (k == 50) or (k == 199):
      ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
  if (k != 1):
    ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
```

-:--- **fleur5petales.py** Top L28 (Python)

Wrote /home/vella-chemla/Desktop/fleur5petales.py

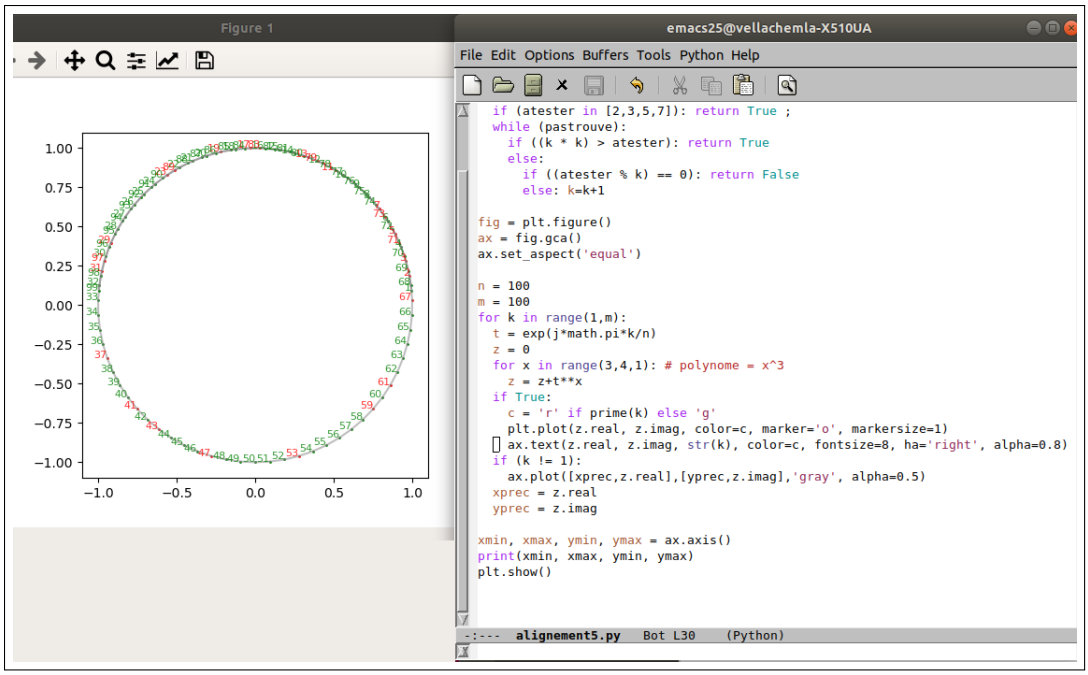


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(3,4,1): # polynome = x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
plt.show()
-:--- alignement5.py Bot L30 (Python)

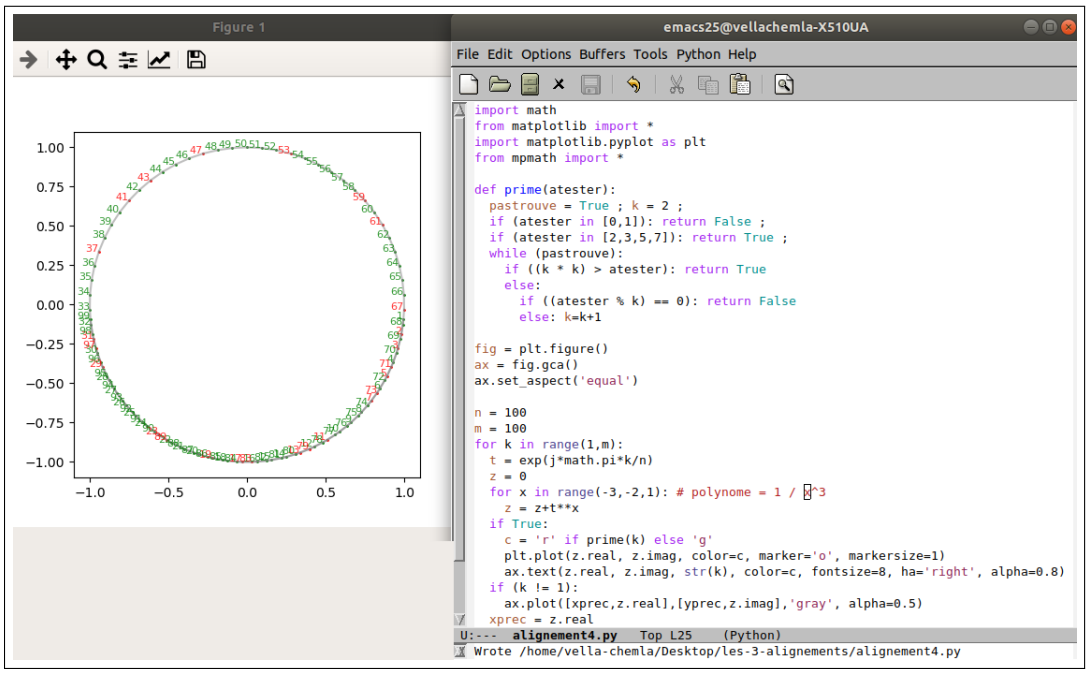


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-3,-2,1): # polynome = 1 / x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
U:--- alignement4.py Top L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement4.py

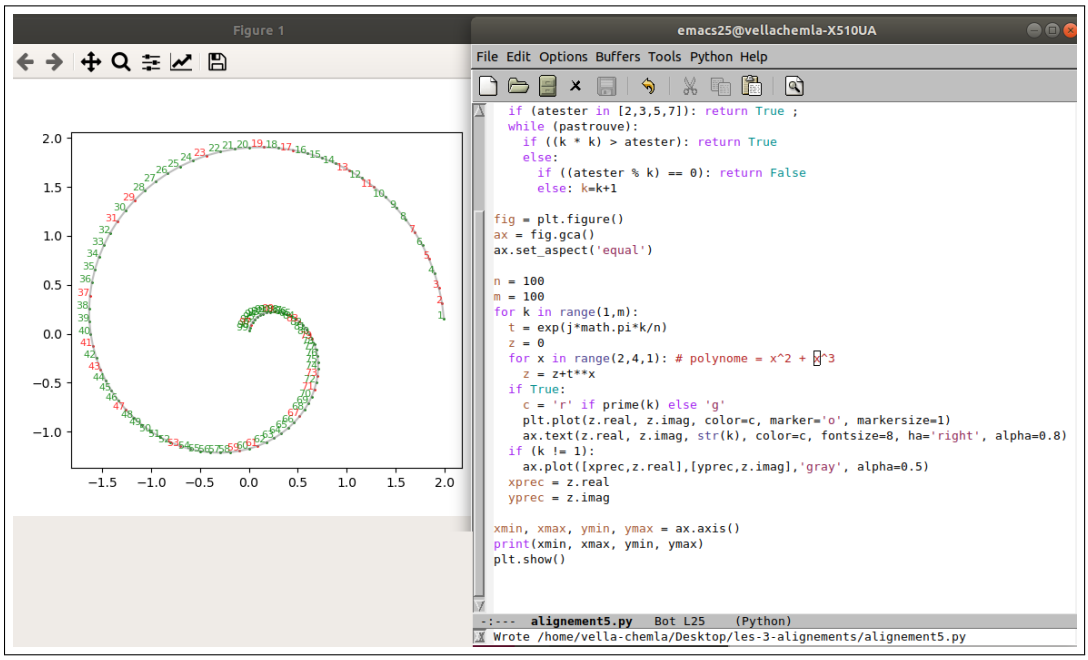


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(2,4,1): # polynome = x^2 + x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
plt.show()
-:--- alignement5.py Bot L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement5.py

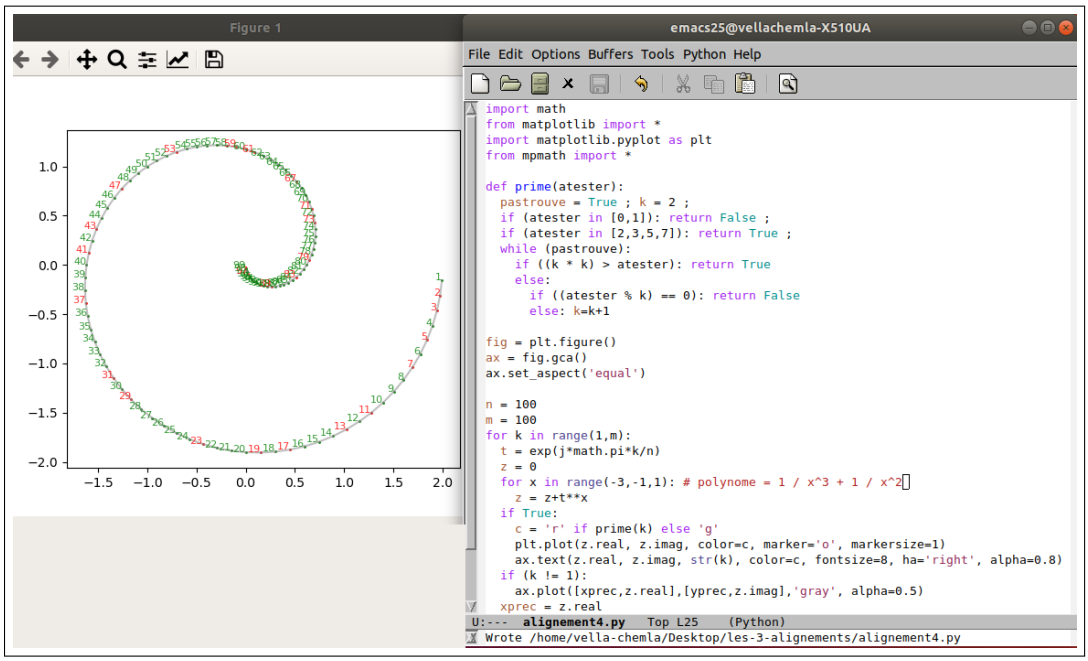


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-3,-1,1): # polynome = 1 / x^3 + 1 / x^2
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
U:--- alignement4.py Top L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement4.py

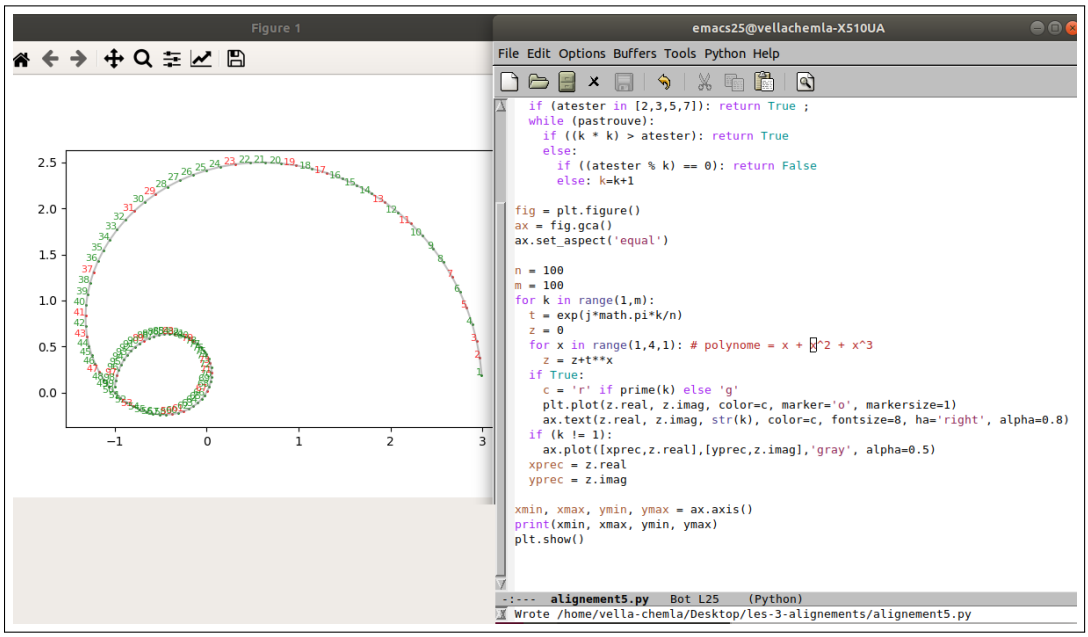


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(1,4,1): # polynome = x + x^2 + x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
plt.show()
-:--- alignement5.py Bot L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement5.py

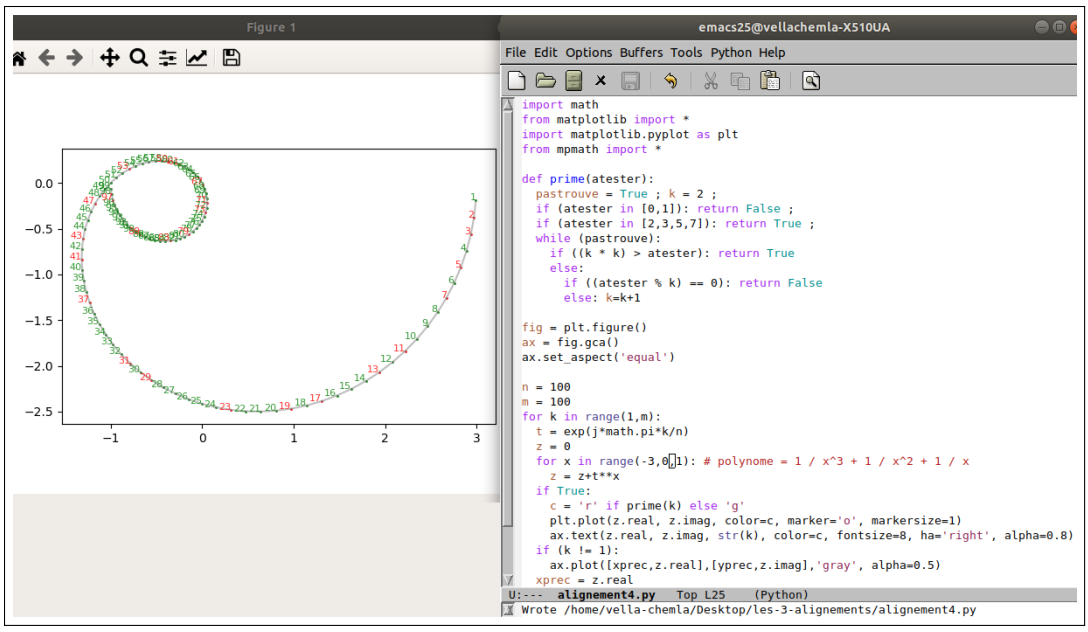


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-3,0,1): # polynome = 1 / x^3 + 1 / x^2 + 1 / x
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
U:--- alignement4.py Top L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement4.py

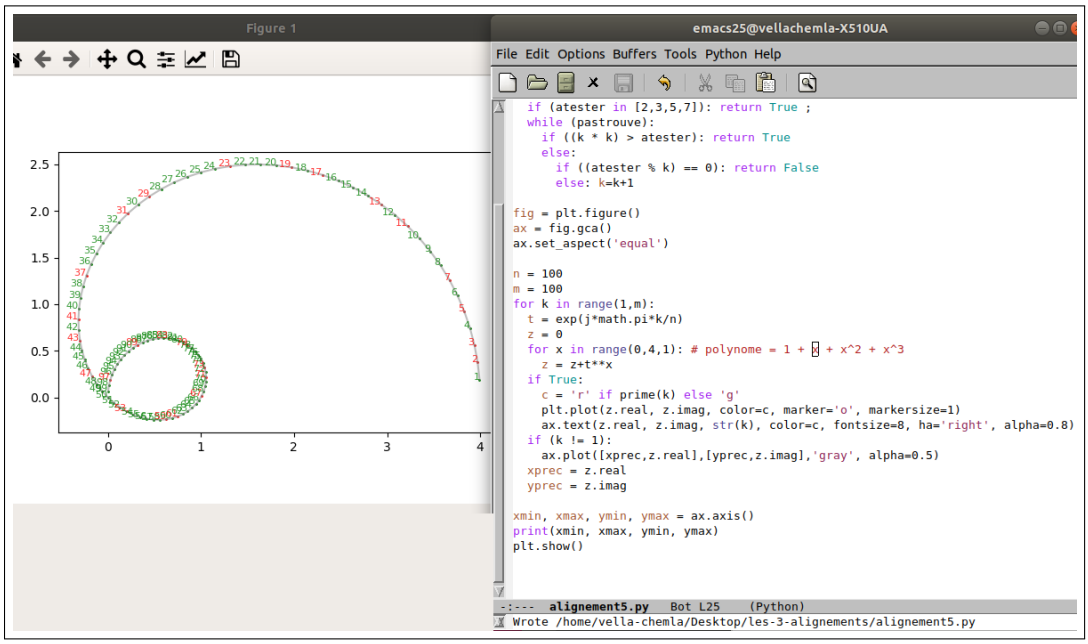


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(0,4,1): # polynome = 1 + x + x^2 + x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
plt.show()
-:--- alignement5.py Bot L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement5.py

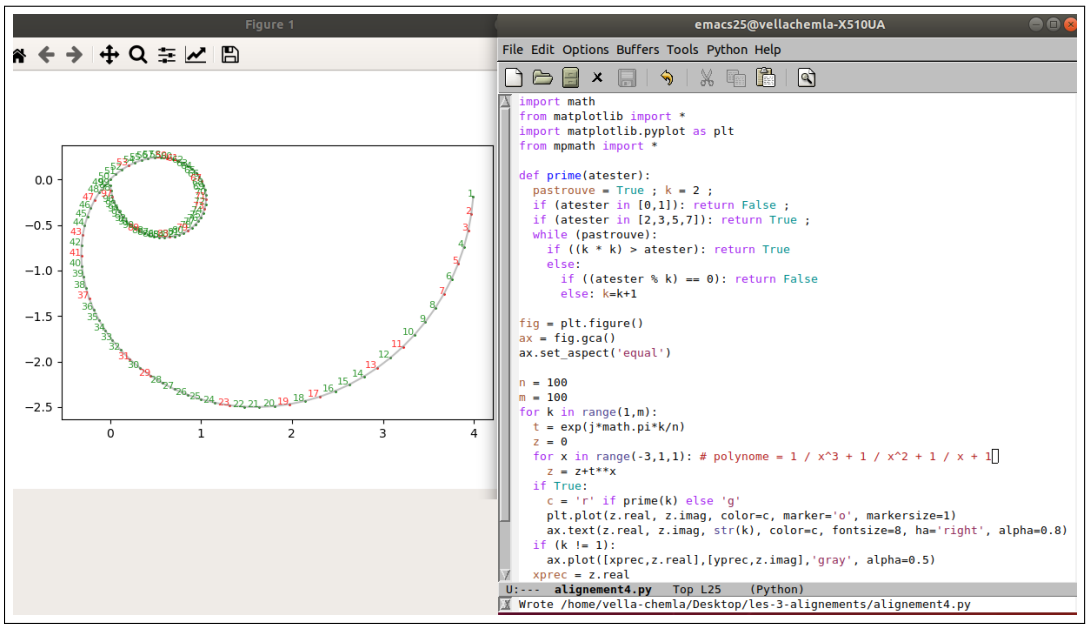


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-3,1,1): # polynome = 1 / x^3 + 1 / x^2 + 1 / x + 1
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
U:--- alignement4.py Top L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement4.py

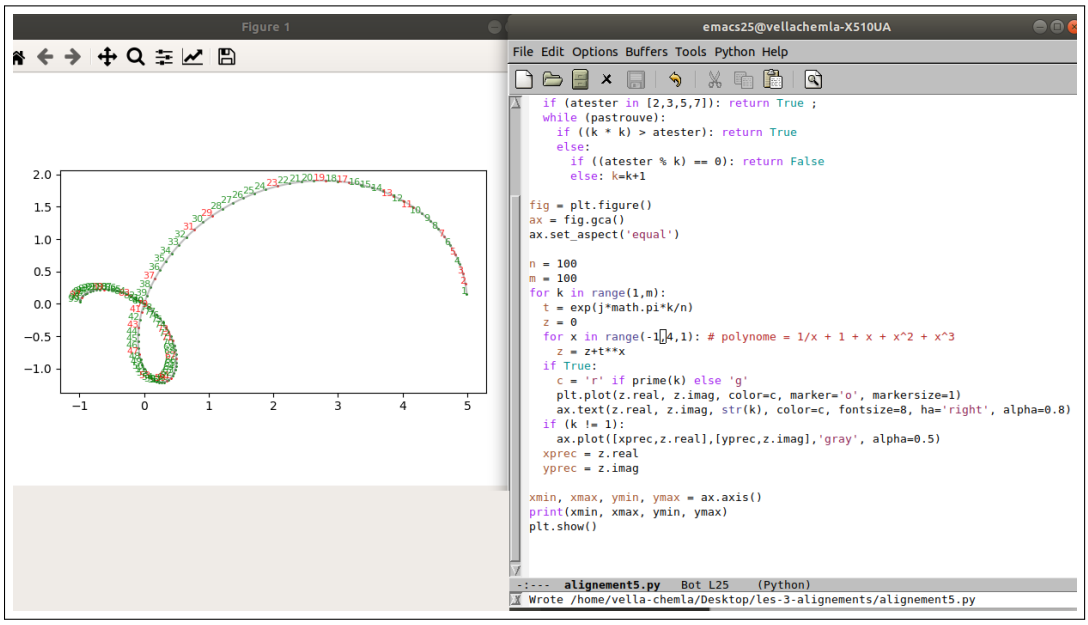


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-1[illegible]4,1): # polynome = 1/x + 1 + x + x^2 + x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
plt.show()
-:--- alignement5.py Bot L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement5.py

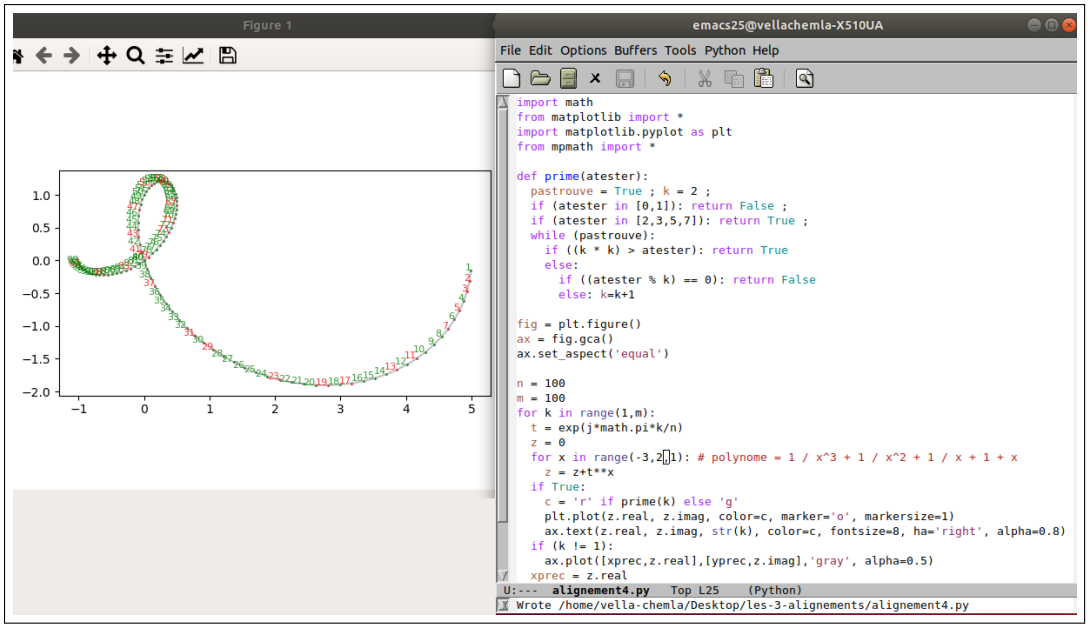


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-3,2[illegible]1): # polynome = 1 / x^3 + 1 / x^2 + 1 / x + 1 + x
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
U:--- alignement4.py Top L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement4.py

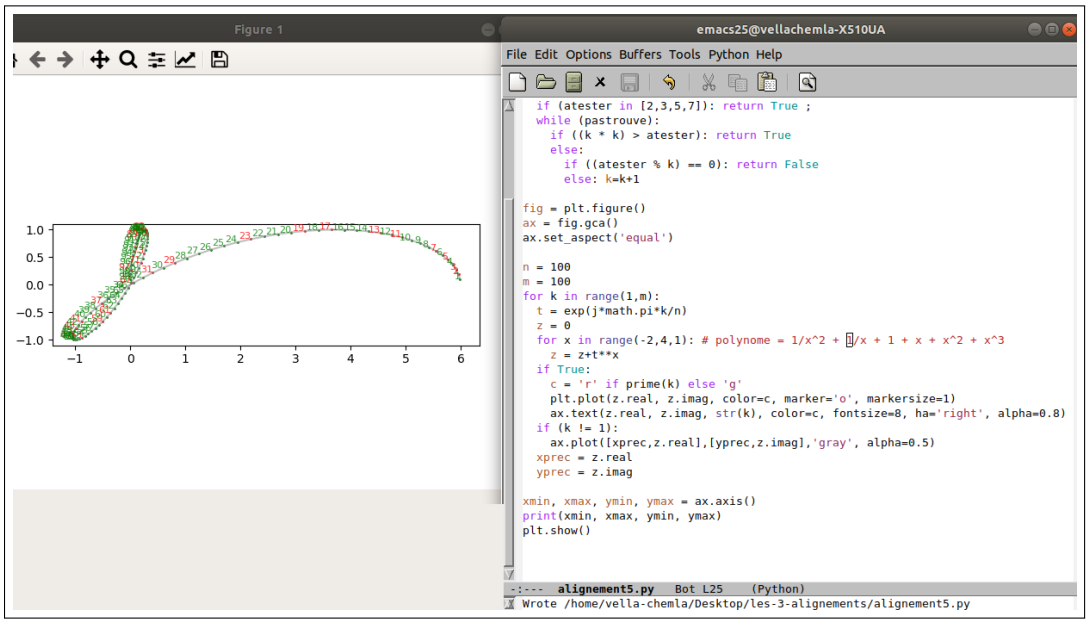


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-2,4,1): # polynome = 1/x^2 + 1/x + 1 + x + x^2 + x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
yprec = z.imag
xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
plt.show()
-:--- alignement5.py Bot L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement5.py

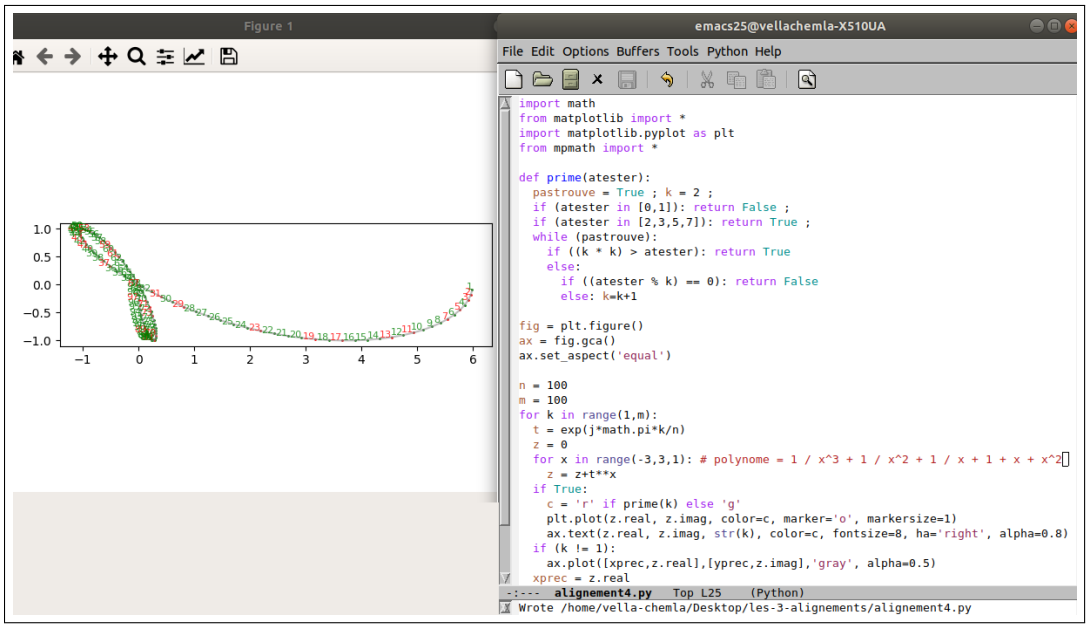


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-3,3,1): # polynome = 1 / x^3 + 1 / x^2 + 1 / x + 1 + x + x^2
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
xprec = z.real
-:--- alignement4.py Top L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement4.py

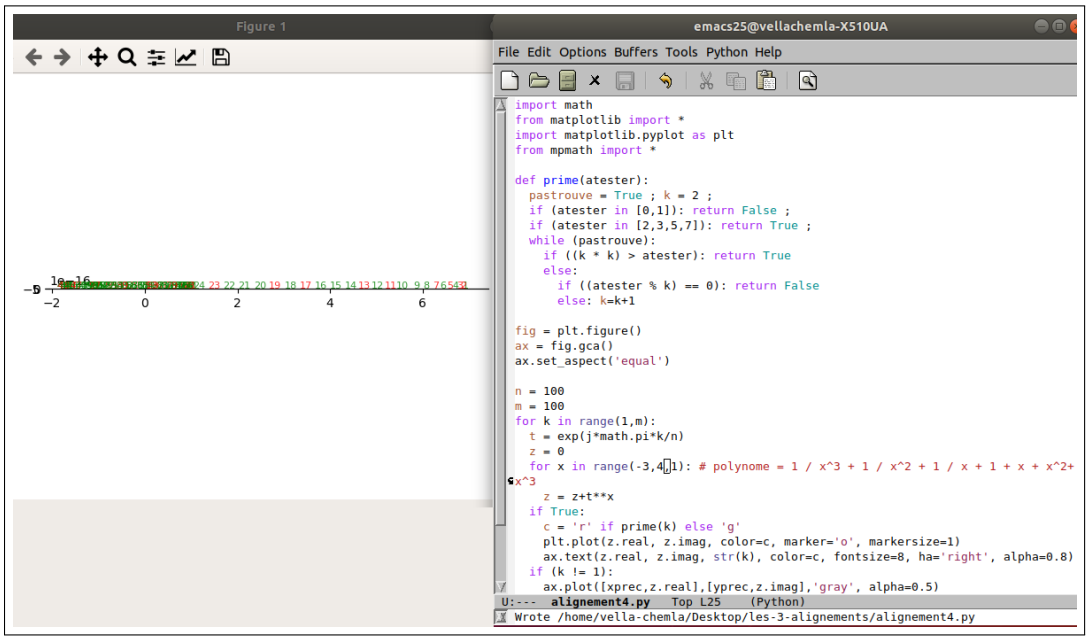
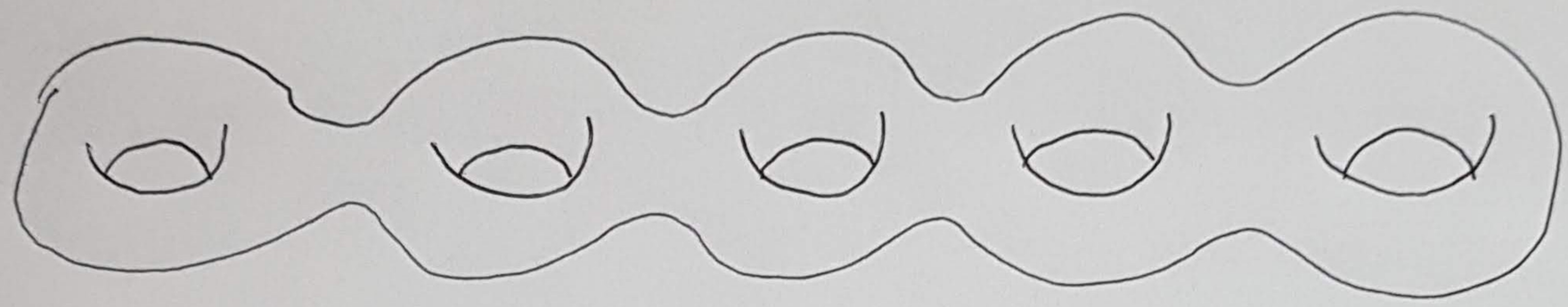
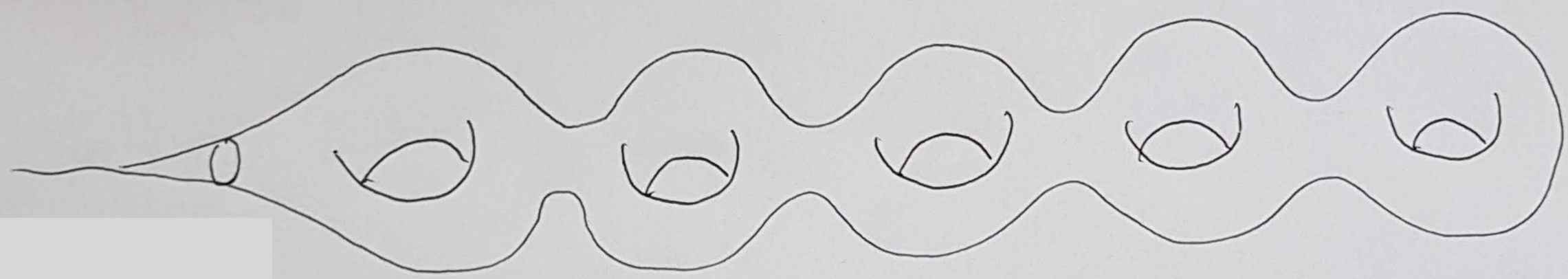
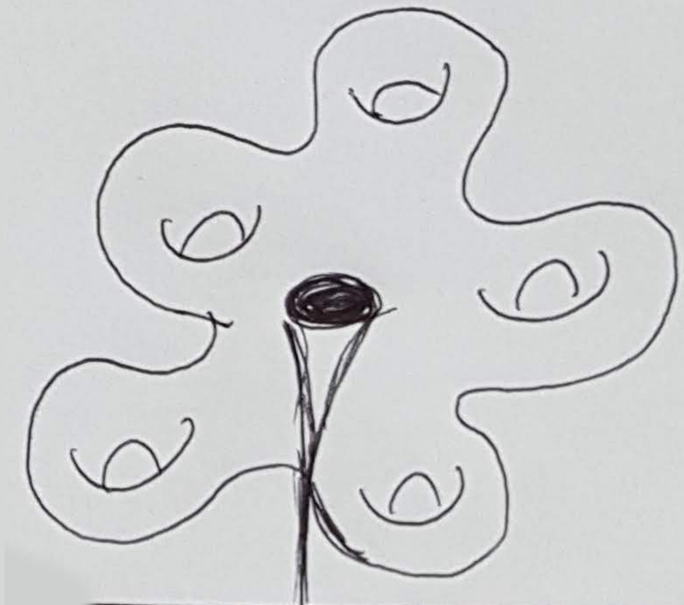


Figure 1
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
import math
from matplotlib import *
import matplotlib.pyplot as plt
from mpmath import *
def prime(atester):
pastrouve = True ; k = 2 ;
if (atester in [0,1]): return False ;
if (atester in [2,3,5,7]): return True ;
while (pastrouve):
if ((k * k) > atester): return True
else:
if ((atester % k) == 0): return False
else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
n = 100
m = 100
for k in range(1,m):
t = exp(j*math.pi*k/n)
z = 0
for x in range(-3,4,1): # polynome = 1 / x^3 + 1 / x^2 + 1 / x + 1 + x + x^2+x^3
z = z+t**x
if True:
c = 'r' if prime(k) else 'g'
plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=0.8)
if (k != 1):
ax.plot([xprec,z.real],[yprec,z.imag],'gray', alpha=0.5)
U:--- alignement4.py Top L25 (Python)
Wrote /home/vella-chemla/Desktop/les-3-alignements/alignement4.py







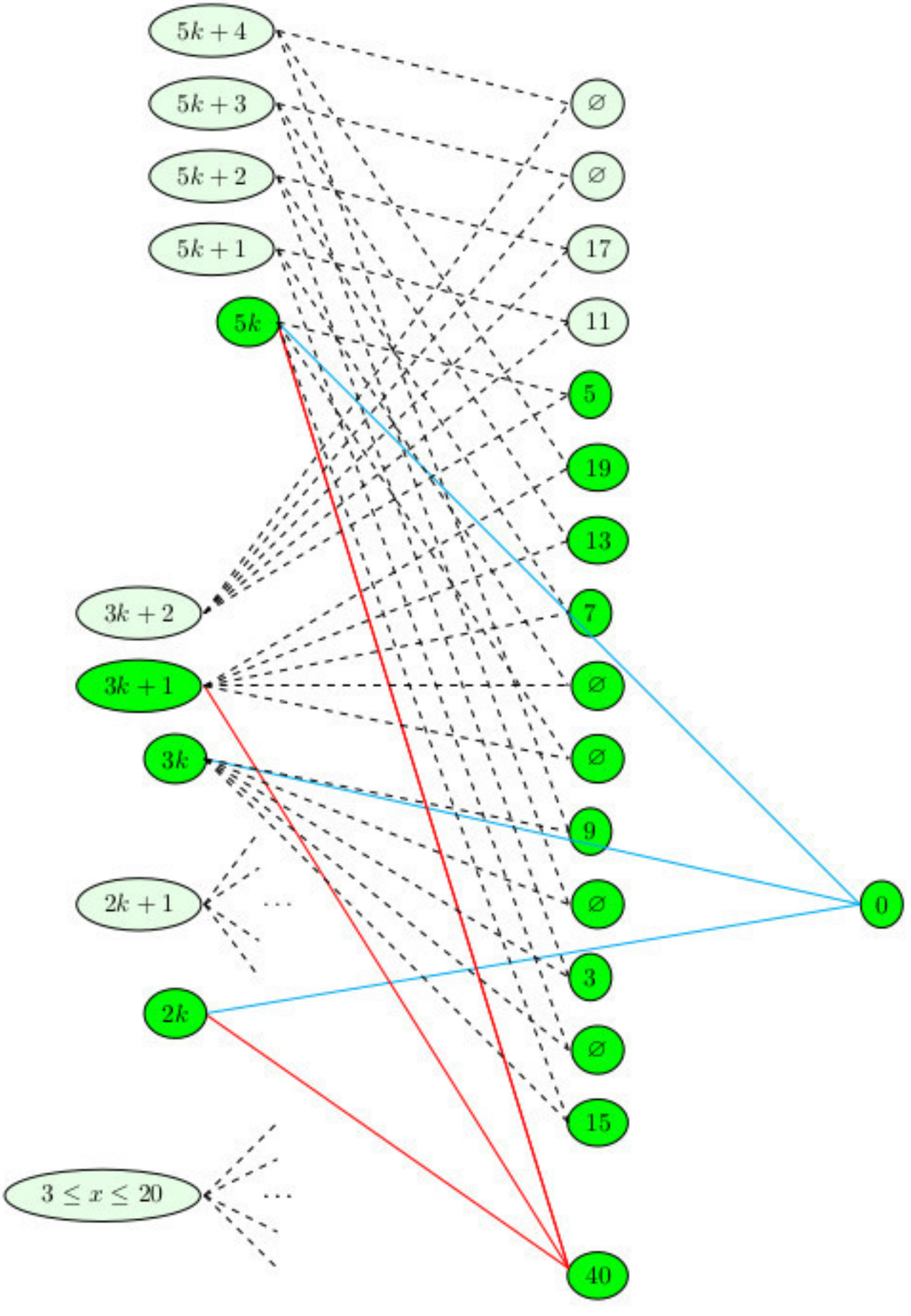
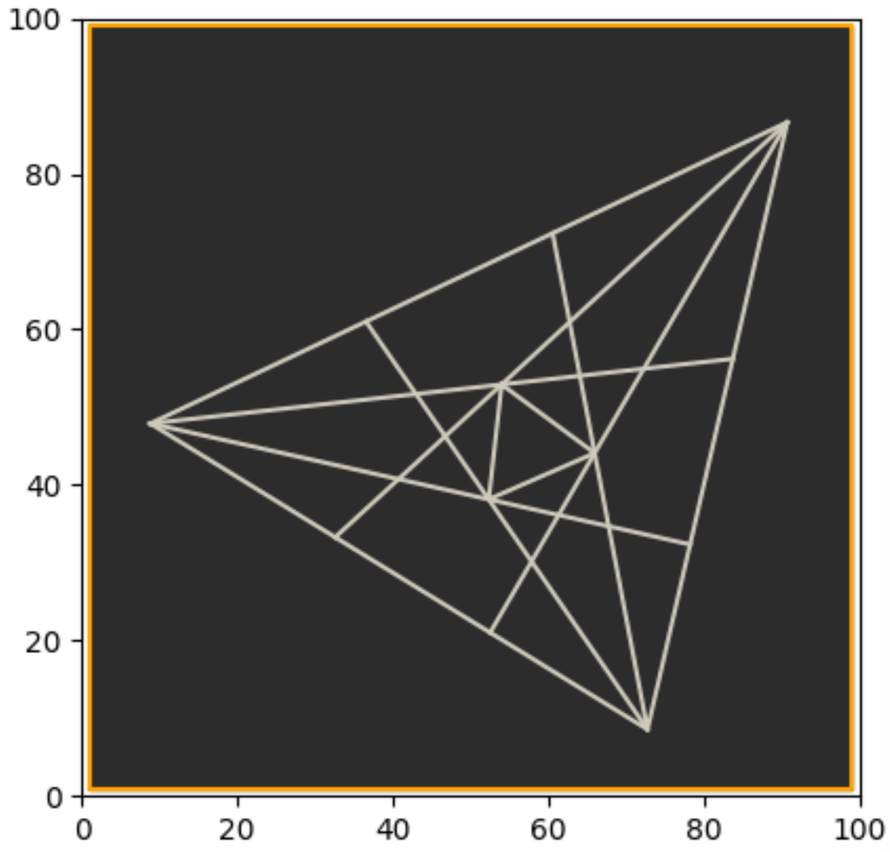

$5k+4$
$5k+3$
$5k+2$
$5k+1$
$5k$
$3k+2$
$3k+1$
$3k$
$2k+1$
$2k$
$3 \le x \le 20$
$\cdots$
$\cdots$
$\varnothing$
$\varnothing$
17
11
5
19
13
7
$\varnothing$
$\varnothing$
9
$\varnothing$
3
$\varnothing$
15
40
0

Figure 1



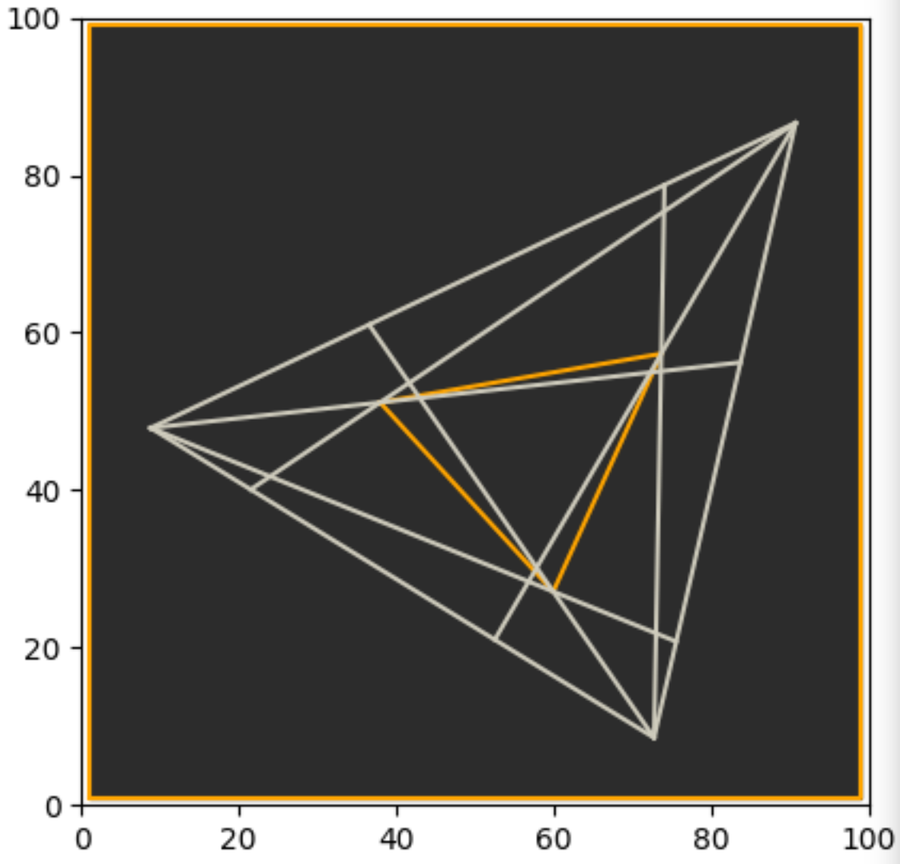
emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
print "x_L = "+str(xl) ;
print "y_L = "+str(yl) ;
print "x_X = "+str(xx) ;
print "y_X = "+str(yx) ;
print "x_D = "+str(xd) ;
print "y_D = "+str(yd) ;
fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
matplotlib.pyplot.fill([1,99,99,1,1],[1,1,99,99,1],color='#2c2c2c') ;
matplotlib.pyplot.plot([1,99,99,1,1],[1,1,99,99,1],color='#ffa500') ;
matplotlib.pyplot.plot([xa,xb,xc,xa],[ya,yb,yc,ya], color='#cecbbc') ;
matplotlib.pyplot.plot([xn,xo,xx,xn],[yn,yo,yx,yn],color='#cecbbc') ;
matplotlib.pyplot.plot([xa,xp],[ya,yp],color='#cecbbc') ;
matplotlib.pyplot.plot([xa,xq],[ya,yq],color='#cecbbc') ;
matplotlib.pyplot.plot([xb,xt],[yb,yt],color='#cecbbc') ;
matplotlib.pyplot.plot([xb,xu],[yb,yu],color='#cecbbc') ;
matplotlib.pyplot.plot([xc,xr],[yc,yr],color='#cecbbc') ;
matplotlib.pyplot.plot([xc,xs],[yc,ys],color='#cecbbc') ;

matplotlib.pyplot.xlim(0,100)
matplotlib.pyplot.ylim(0,100)

ax.set_aspect('equal')

matplotlib.pyplot.show() ;
```

-:--- **morley-python-2-traits-seuls.py** Bot L168 (Python)

Figure 1



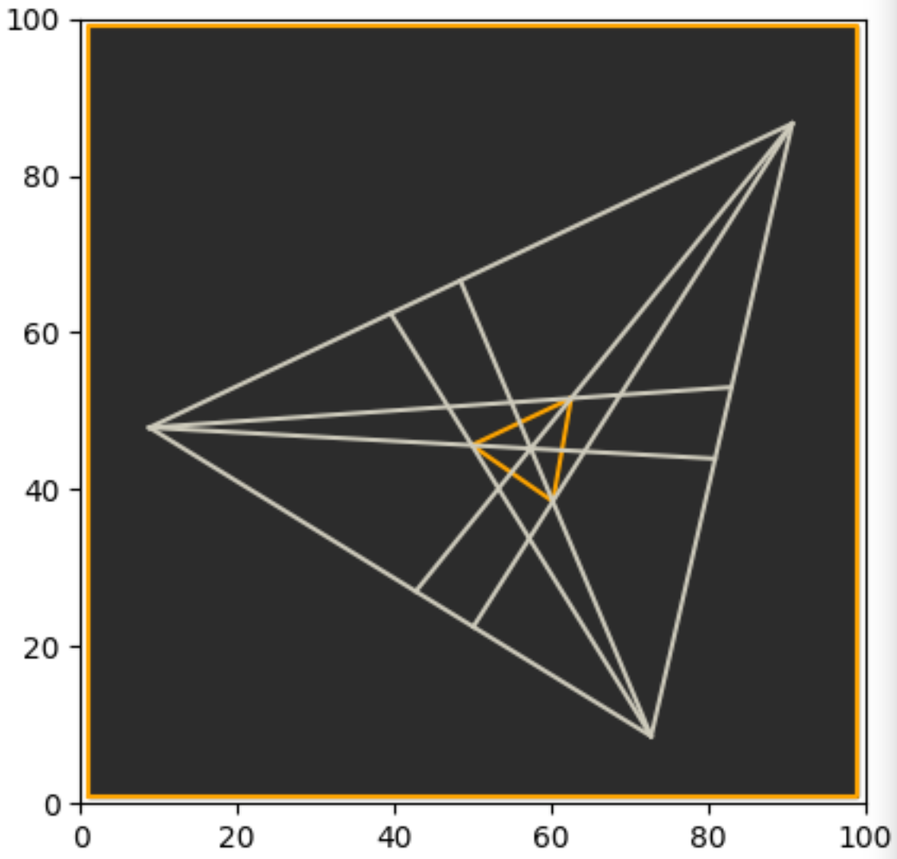
emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
yba=ya-yb ;
xbc=xc-xb ;
ybc=yc-yb ;
xca=xa-xc ;
yca=ya-yc ;
xcb=xb-xc ;
ycb=yb-yc ;
anglea=angle(xab,yab,xac,yac) ;
angleb=angle(xbc,ybc,xba,yba) ;
anglec=angle(xca,yca,xcb,ycb) ;
print "angle en a "+str(anglea) ;
print "angle en b "+str(angleb) ;
print "angle en c "+str(anglec) ;
xaprime = xa+rotabs(xab,yab,anglea/6.0) ;
yaprime = ya+rotord(xab,yab,anglea/6.0) ;
xaseconde = xa+rotabs(xab,yab,anglea*2.0/3.0) ;
yaseconde = ya+rotord(xab,yab,anglea*2.0/3.0) ;
xbprime = xb+rotabs(xbc,ybc,angleb/6.0) ;
ybprime = yb+rotord(xbc,ybc,angleb/6.0) ;
xbseconde = xb+rotabs(xbc,ybc,angleb*2.0/3.0) ;
ybseconde = yb+rotord(xbc,ybc,angleb*2.0/3.0) ;
xcprime = xc+rotabs(xca,yca,anglec/6.0) ;
ycprime = yc+rotord(xca,yca,anglec/6.0) ;
xcseconde = xc+rotabs(xca,yca,anglec*2.0/3.0) ;
ycseconde = yc+rotord(xca,yca,anglec*2.0/3.0) ;

xp=intersecteabs(xa,ya,xaseconde,yaseconde,xb,yb,xc,yc) ;
yp=intersecteord(xa,ya,xaseconde,yaseconde,xb,yb,xc,yc) ;
xq=intersecteabs(xa,ya,xaprime,yaprime,xb,yb,xc,yc) ;
yq=intersecteord(xa,ya,xaprime,yaprime,xb,yb,xc,yc) ;
xr=intersecteabs(xc,yc,xcseconde,ycseconde,xa,ya,xb,yb) ;
yr=intersecteord(xc,yc,xcseconde,ycseconde,xa,ya,xb,yb) ;
xs=intersecteabs(xc,yc,xcprime,ycprime,xa,ya,xb,yb) ;
```

-:--- **morley-python-2-traits-seuls.py** 19% L76 (Python)

Figure 1



emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

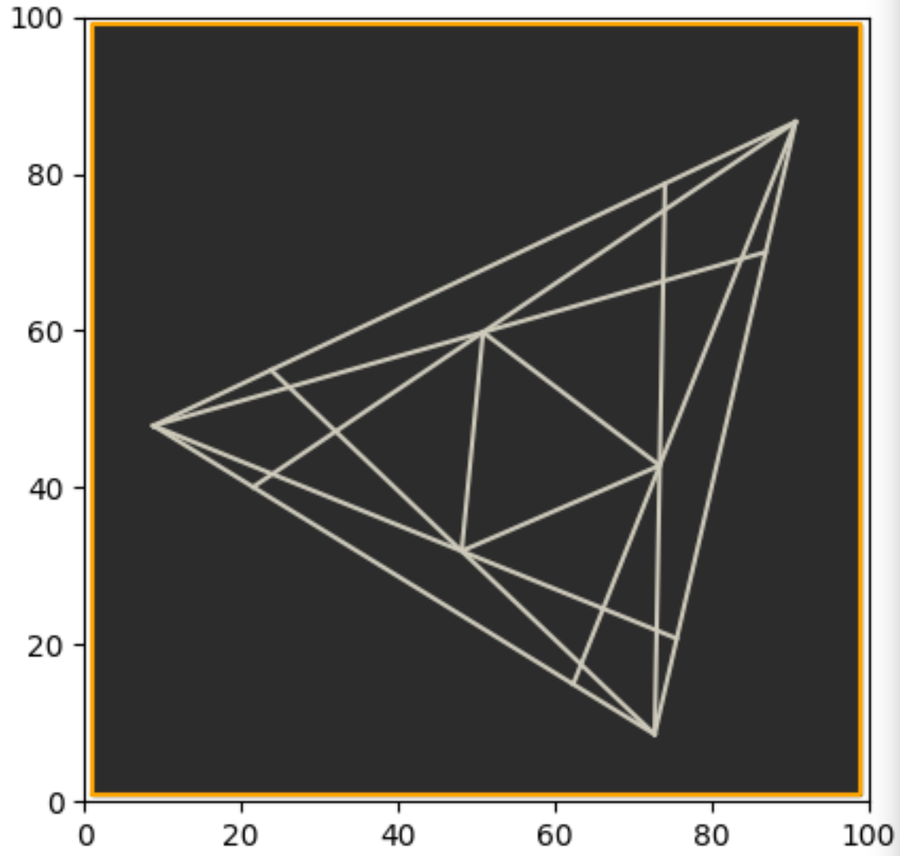
```
yba=ya-yb ;
xbc=xc-xb ;
ybc=yc-yb ;
xca=xa-xc ;
yca=ya-yc ;
xcb=xb-xc ;
ycb=yb-yc ;
anglea=angle(xab,yab,xac,yac) ;
angleb=angle(xbc,ybc,xba,yba) ;
anglec=angle(xca,yca,xcb,ycb) ;
print "angle en a "+str(anglea) ;
print "angle en b "+str(angleb) ;
print "angle en c "+str(anglec) ;
xaprime = xa+rotabs(xab,yab,anglea/2.0) ;
yaprime = ya+rotord(xab,yab,anglea/2.0) ;
xaseconde = xa+rotabs(xab,yab,anglea*5.0/8.0) ;
yaseconde = ya+rotord(xab,yab,anglea*5.0/8.0) ;
xbprime = xb+rotabs(xbc,ybc,angleb/2.0) ;
ybprime = yb+rotord(xbc,ybc,angleb/2.0) ;
xbseconde = xb+rotabs(xbc,ybc,angleb*5.0/8.0) ;
ybseconde = yb+rotord(xbc,ybc,angleb*5.0/8.0) ;
xcprime = xc+rotabs(xca,yca,anglec/2.0) ;
ycprime = yc+rotord(xca,yca,anglec/2.0) ;
xcseconde = xc+rotabs(xca,yca,anglec*5.0/8.0) ;
ycseconde = yc+rotord(xca,yca,anglec*5.0/8.0) ;

xp=intersecteabs(xa,ya,xaseconde,yaseconde,xb,yb,xc,yc) ;
yp=intersecteord(xa,ya,xaseconde,yaseconde,xb,yb,xc,yc) ;
xq=intersecteabs(xa,ya,xaprime,yaprime,xb,yb,xc,yc) ;
yq=intersecteord(xa,ya,xaprime,yaprime,xb,yb,xc,yc) ;
xr=intersecteabs(xc,yc,xcseconde,ycseconde,xa,ya,xb,yb) ;
yr=intersecteord(xc,yc,xcseconde,ycseconde,xa,ya,xb,yb) ;
```

-:--- **morley-python-2-traits-seuls.py** 19% L77 (Python)

Wrote /home/vella-chemla/Desktop/mon-Morley-a-moi/morley-python-2-traits-seuls.p

y

Figure 1



emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

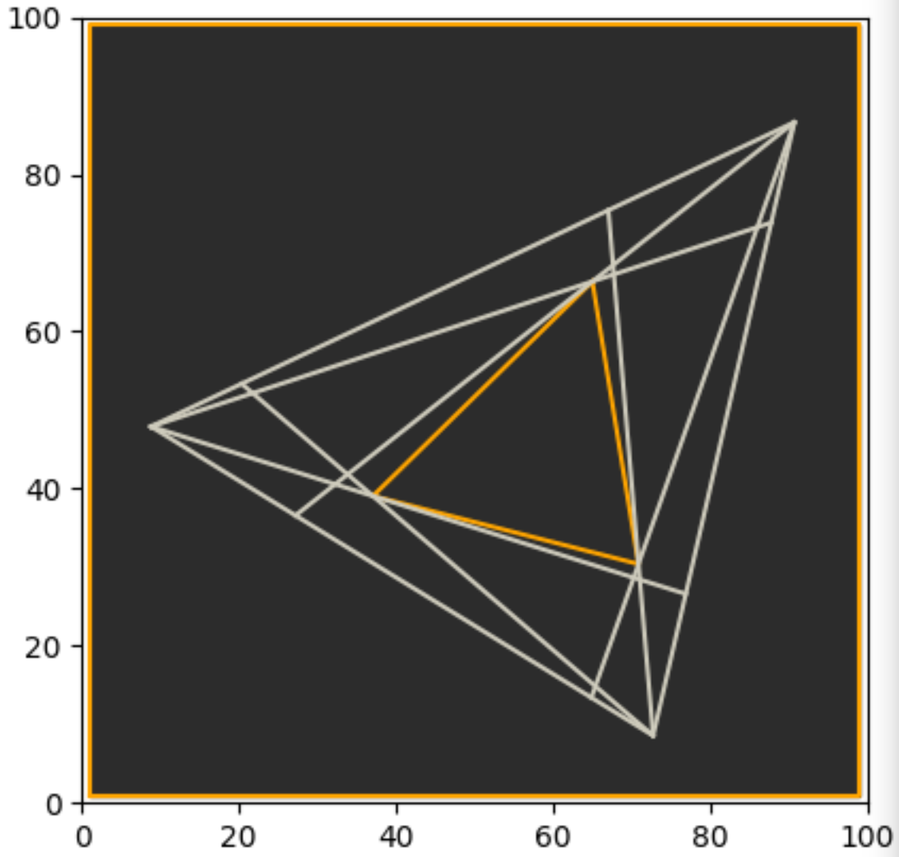
```
ycb=yb-yc ;
anglea=angle(xab,yab,xac,yac) ;
angleb=angle(xbc,ybc,xba,yba) ;
anglec=angle(xca,yca,xcb,ycb) ;
print "angle en a "+str(anglea) ;
print "angle en b "+str(angleb) ;
print "angle en c "+str(anglec) ;
xaprime = xa+rotabs(xab,yab,anglea/6.0) ;
yaprime = ya+rotord(xab,yab,anglea/6.0) ;
xaseconde = xa+rotabs(xab,yab,anglea*5.0/6.0) ;
yaseconde = ya+rotord(xab,yab,anglea*5.0/6.0) ;
xbprime = xb+rotabs(xbc,ybc,angleb/6.0) ;
ybprime = yb+rotord(xbc,ybc,angleb/6.0) ;
xbseconde = xb+rotabs(xbc,ybc,angleb*5.0/6.0) ;
ybseconde = yb+rotord(xbc,ybc,angleb*5.0/6.0) ;
xcprime = xc+rotabs(xca,yca,anglec/6.0) ;
ycprime = yc+rotord(xca,yca,anglec/6.0) ;
xcseconde = xc+rotabs(xca,yca,anglec*5.0/6.0) ;
ycseconde = yc+rotord(xca,yca,anglec*5.0/6.0) ;

xp=intersecteabs(xa,ya,xaseconde,yaseconde,xb,yb,xc,yc) ;
yp=intersecteord(xa,ya,xaseconde,yaseconde,xb,yb,xc,yc) ;
xq=intersecteabs(xa,ya,xaprime,yaprime,xb,yb,xc,yc) ;
yq=intersecteord(xa,ya,xaprime,yaprime,xb,yb,xc,yc) ;
xr=intersecteabs(xc,yc,xcseconde,ycseconde,xa,ya,xb,yb) ;
yr=intersecteord(xc,yc,xcseconde,ycseconde,xa,ya,xb,yb) ;
xs=intersecteabs(xc,yc,xcprime,ycprime,xa,ya,xb,yb) ;
ys=intersecteord(xc,yc,xcprime,ycprime,xa,ya,xb,yb) ;
xt=intersecteabs(xb,yb,xbseconde,ybseconde,xc,yc,xa,ya) ;
yt=intersecteord(xb,yb,xbseconde,ybseconde,xc,yc,xa,ya) ;
xu=intersecteabs(xb,yb,xbprime,ybprime,xc,yc,xa,ya) ;
yu=intersecteord(xb,yb,xbprime,ybprime,xc,yc,xa,ya) ;
```

-:--- **morley-python-2-traits-seuls.py** 21% L81 (Python)

Wrote /home/vella-chemla/Desktop/mon-Morley-a-moi/morley-python-2-traits-seuls.p

y

Figure 1



emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
yba=ya-yb ;
xbc=xc-xb ;
ybc=yc-yb ;
xca=xa-xc ;
yca=ya-yc ;
xcb=xb-xc ;
ycb=yb-yc ;
anglea=angle(xab,yab,xac,yac) ;
angleb=angle(xbc,ybc,xba,yba) ;
anglec=angle(xca,yca,xcb,ycb) ;
print "angle en a "+str(anglea) ;
print "angle en b "+str(angleb) ;
print "angle en c "+str(anglec) ;
xaprime = xa+rotabs(xab,yab,anglea/4.0) ;
yaprime = ya+rotord(xab,yab,anglea/4.0) ;
xaseconde = xa+rotabs(xab,yab,anglea*7.0/8.0) ;
yaseconde = ya+rotord(xab,yab,anglea*7.0/8.0) ;
xbprime = xb+rotabs(xbc,ybc,angleb/4.0) ;
ybprime = yb+rotord(xbc,ybc,angleb/4.0) ;
xbseconde = xb+rotabs(xbc,ybc,angleb*7.0/8.0) ;
ybseconde = yb+rotord(xbc,ybc,angleb*7.0/8.0) ;
xcprime = xc+rotabs(xca,yca,anglec/4.0) ;
ycprime = yc+rotord(xca,yca,anglec/4.0) ;
xcseconde = xc+rotabs(xca,yca,anglec*7.0/8.0) ;
ycseconde = yc+rotord(xca,yca,anglec*7.0/8.0) ;

xp=intersecteabs(xa,ya,xaseconde,yaseconde,xb,yb,xc,yc) ;
yp=intersecteord(xa,ya,xaseconde,yaseconde,xb,yb,xc,yc) ;
xq=intersecteabs(xa,ya,xaprime,yaprime,xb,yb,xc,yc) ;
yq=intersecteord(xa,ya,xaprime,yaprime,xb,yb,xc,yc) ;
xr=intersecteabs(xc,yc,xcseconde,ycseconde,xa,ya,xb,yb) ;
yr=intersecteord(xc,yc,xcseconde,ycseconde,xa,ya,xb,yb) ;
```

-:--- **morley-python-2-traits-seuls.py** 19% L81 (Python)

Wrote /home/vella-chemla/Desktop/mon-Morley-a-moi/morley-python-2-traits-seuls.p

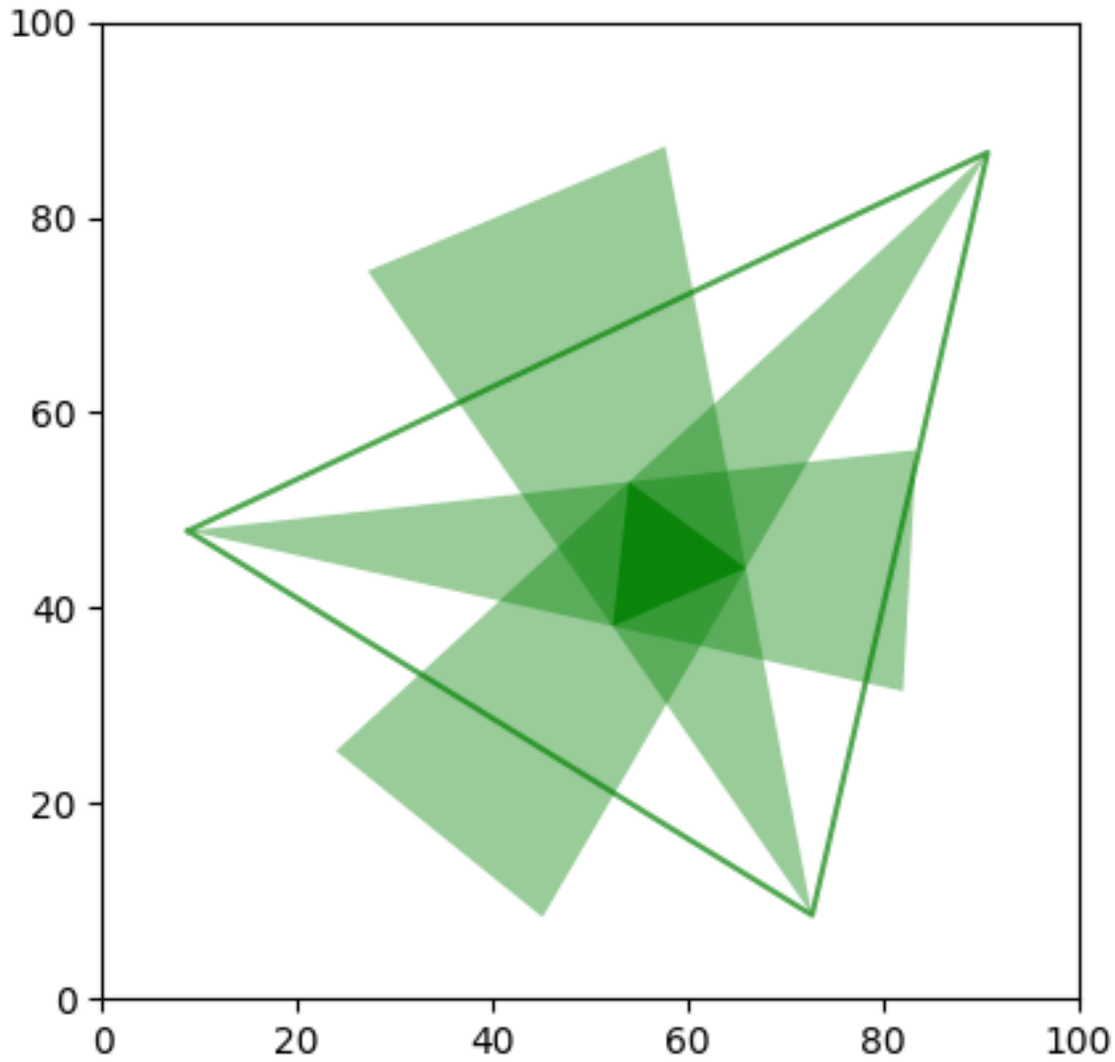
Sy

Terminal

Fichier Édition Affichage Rechercher Terminal Aide

```
vella-chemla@vellachemla-X510UA:~$ cd Desktop/Morley-python/
vella-chemla@vellachemla-X510UA:~/Desktop/Morley-python$ python modif-oubli-p-q-
etc.py
norme(vecteur-NO) = 14.8637468062
norme(vecteur-OX) = 14.8637468062
norme(vecteur-XN) = 14.8637468062
```

Figure 1

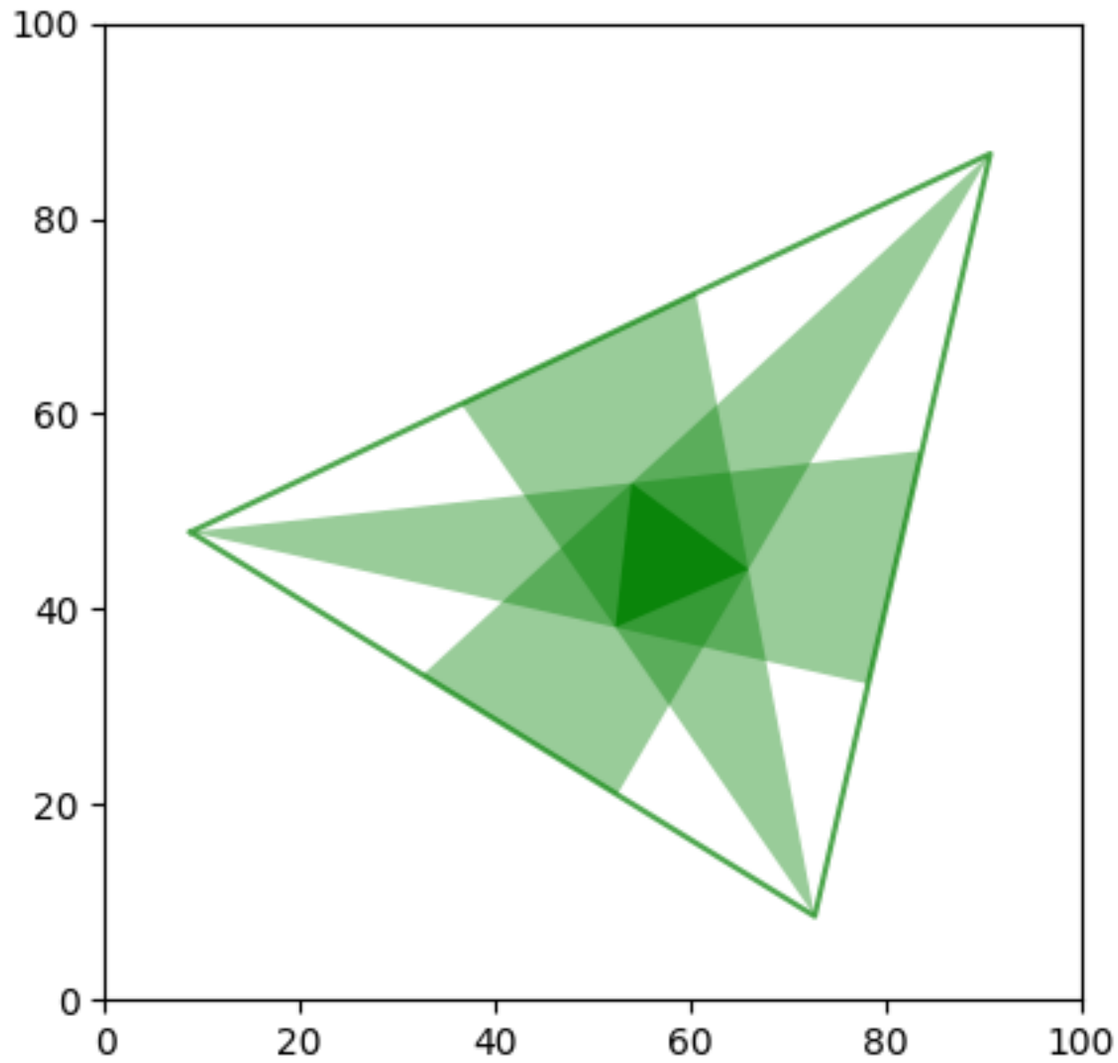


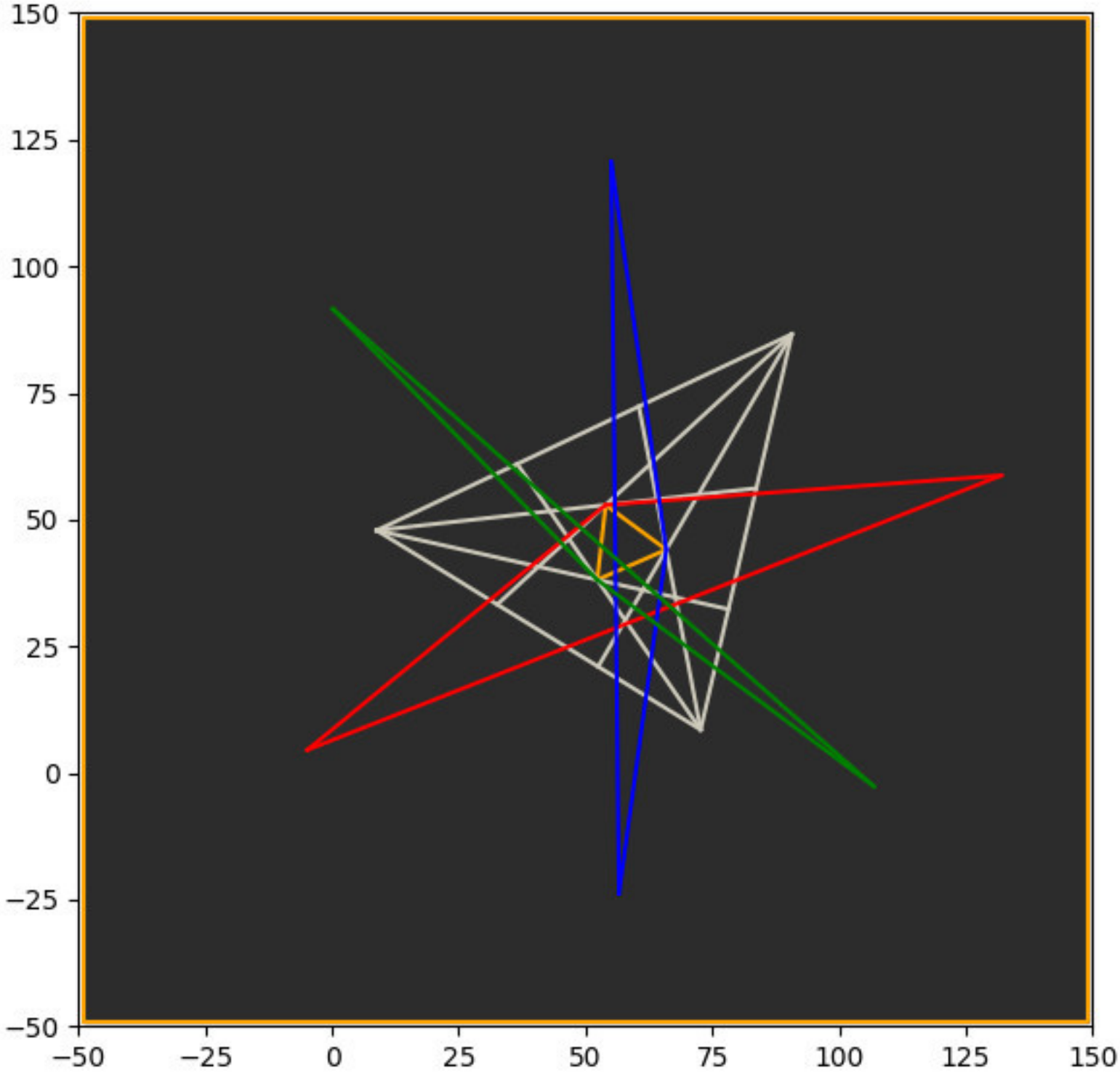
Terminal

Fichier Édition Affichage Rechercher Terminal Aide

```
vella-chemla@vellachemla-X510UA:~$ cd Desktop/Morley-python/
vella-chemla@vellachemla-X510UA:~/Desktop/Morley-python$ python essai-morley-pyt
hon.py
norme(vecteur-NO) = 14.8637468062
norme(vecteur-OX) = 14.8637468062
norme(vecteur-XN) = 14.8637468062
```

Figure 1




150
125
100
75
50
25
0
−25
−50
−50
−25
0
25
50
75
100
125
150

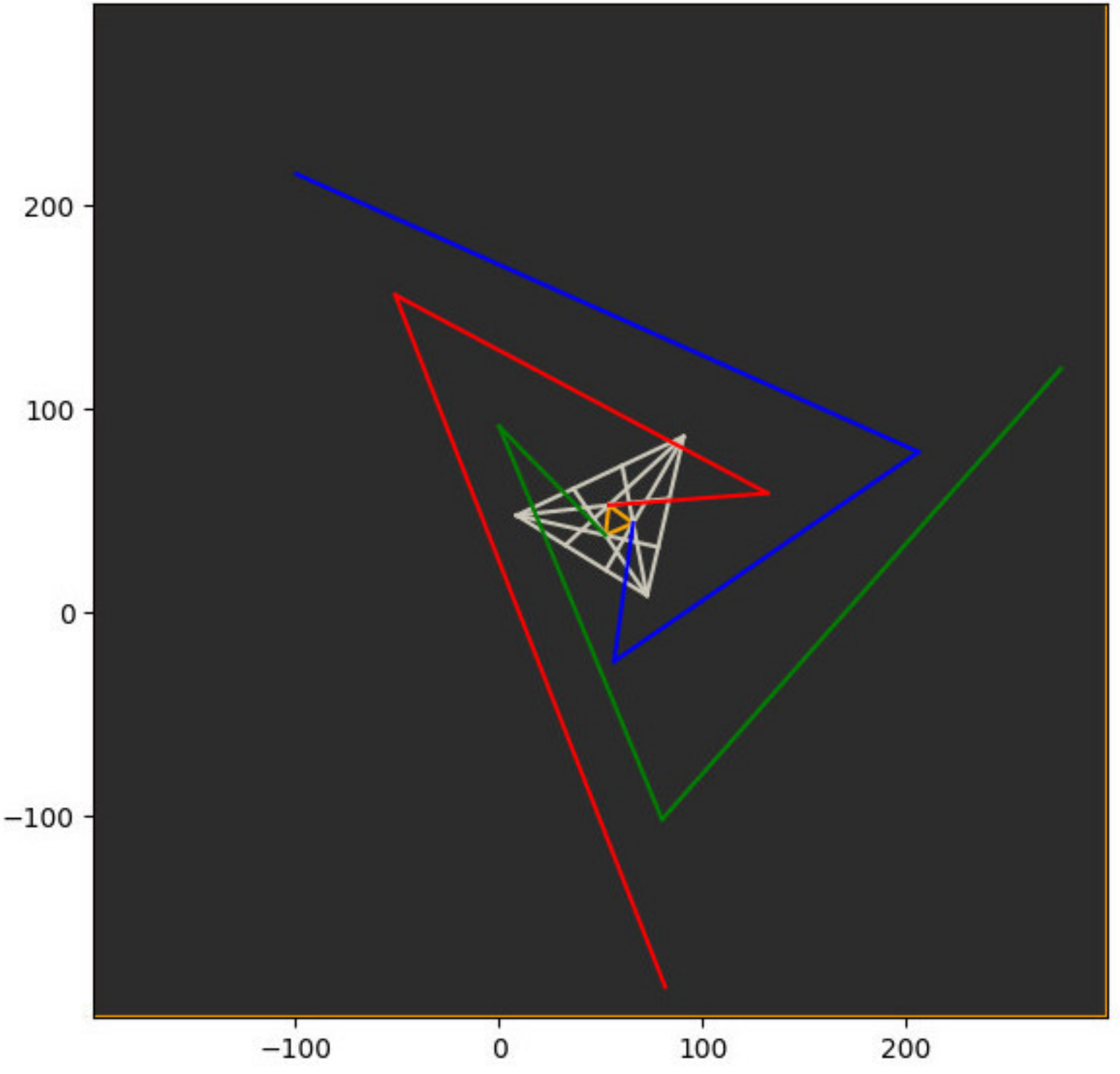
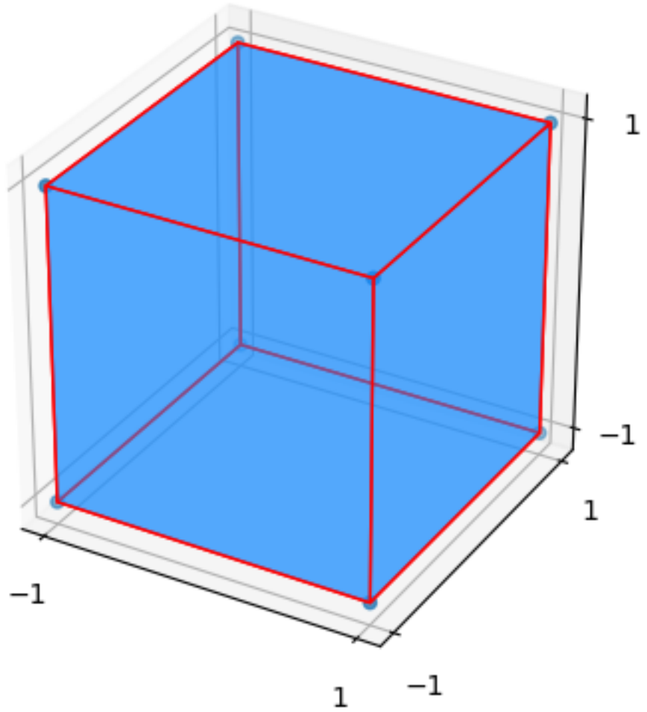

200
100
0
−100
−100
0
100
200

Figure 1



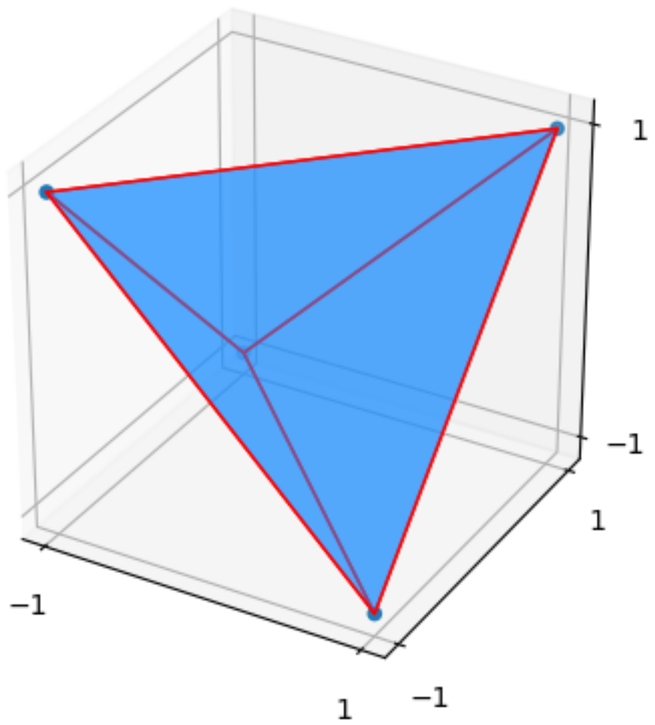
emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
import numpy as np
from matplotlib import colors
from mpl_toolkits.mplot3d import Axes3D
from mpl_toolkits.mplot3d.art3d import Poly3DCollection, Line3DCollection
import matplotlib.pyplot as plt

points = np.array([[-1, -1, -1],
                   [1, -1, -1 ],
                   [1, 1, -1],
                   [-1, 1, -1],
                   [-1, -1, 1],
                   [1, -1, 1 ],
                   [1, 1, 1],
                   [-1, 1, 1]])
alpha = 0.5
fig = plt.figure()
axes = fig.gca(projection='3d')
axes.set_xticks([-1,1])
axes.set_yticks([-1,1])
axes.set_zticks([-1,1])
color = 'dodgerblue'
rgb = colors.colorConverter.to_rgb(color)
rgba = rgb + (alpha,)
r = [-1,1]
X, Y = np.meshgrid(r, r)
axes.scatter3D(points[:, 0], points[:, 1], points[:, 2])
verts = [[points[0],points[1],points[2],points[3]],
         [points[4],points[5],points[6],points[7]],
         [points[0],points[1],points[5],points[4]],
         [points[2],points[3],points[7],points[6]],
         [points[1],points[2],points[6],points[5]],
         [points[4],points[7],points[3],points[0]]]
axes.add_collection3d(Poly3DCollection(verts, facecolors=rgba, edgecolors='r'))
axes.set_aspect('equal')
plt.show()
```

-:**- **cube.py** All L14 (Python)

Figure 1



emacs25@vellachemla-X510UA

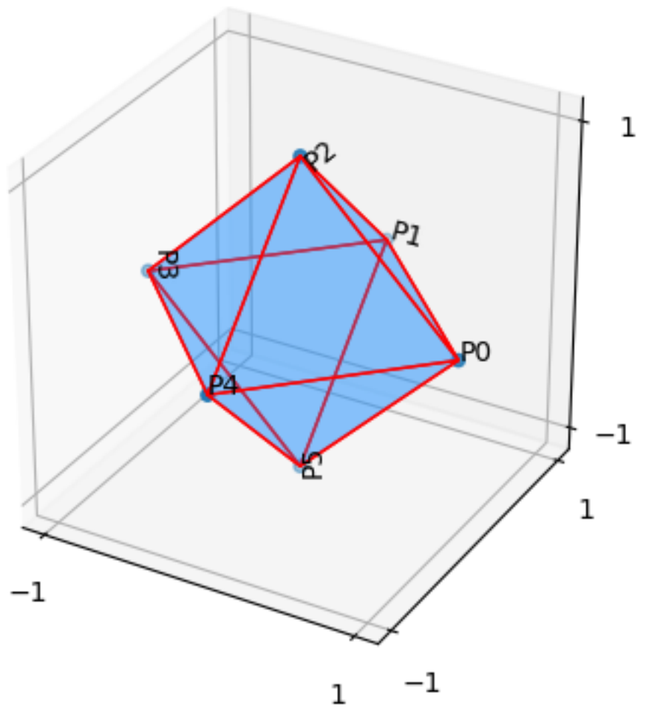
File Edit Options Buffers Tools Python Help

```
import numpy as np
from matplotlib import colors
from mpl_toolkits.mplot3d import Axes3D
from mpl_toolkits.mplot3d.art3d import Poly3DCollection, Line3DCollection
import matplotlib.pyplot as plt

points = np.array([[1, 1, 1],
                   [1, -1, -1],
                   [-1, 1, -1],
                   [-1, -1, 1]])
alpha = 0.5
fig = plt.figure()
axes = fig.gca(projection='3d')
axes.set_xticks([-1,1])
axes.set_yticks([-1,1])
axes.set_zticks([-1,1])
color = 'dodgerblue'
rgb = colors.colorConverter.to_rgb(color)
rgba = rgb + (alpha,)
r = [-1,1]
X, Y = np.meshgrid(r, r)
axes.scatter3D(points[:, 0], points[:, 1], points[:, 2])
verts = [[points[0],points[1],points[2]],
         [points[0],points[3],points[1]],
         [points[0],points[2],points[3]],
         [points[1],points[3],points[2]]]
axes.add_collection3d(Poly3DCollection(verts, facecolors=rgba, edgecolors='r'))
axes.set_aspect('equal')
plt.show()
```

-:--- **tetraedre.py** All L1 (Python)

Can't guess python-indent-offset, using defaults: 4

Figure 1



emacs25@vellachemla-X510UA

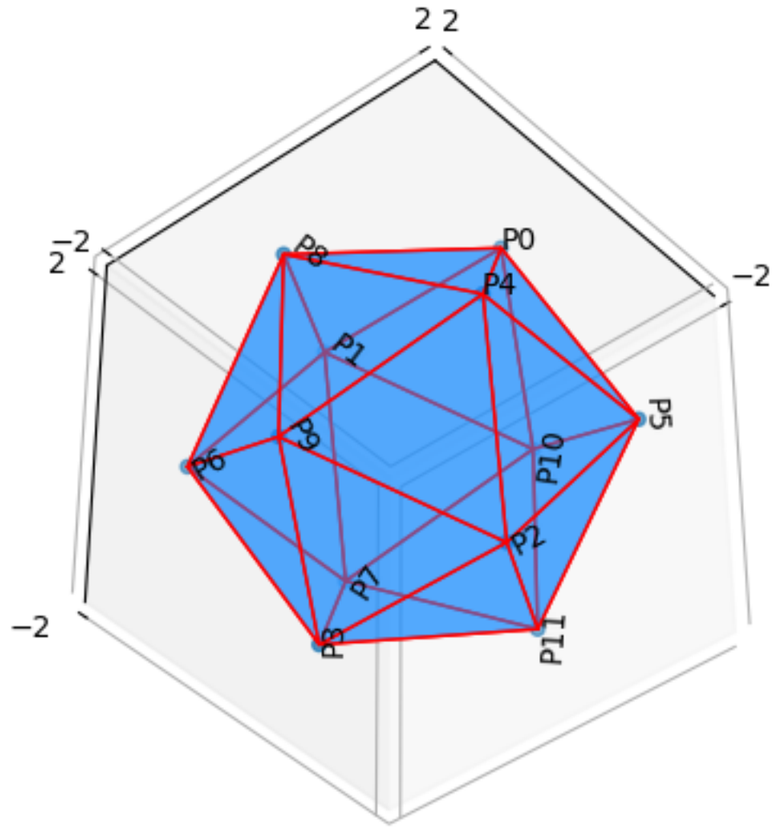
File Edit Options Buffers Tools Python Help

```
zs = (0,0,1,0,0,-1)

alpha = 0.3
fig = plt.figure()
axes = fig.gca(projection='3d')
for zdir, x, y, z, i in zip(zdirs, xs, ys, zs, range(12)):
    label = 'P%d' % (i)
    axes.text(x, y, z, label, zdir)
axes.set_xticks([-1,1])
axes.set_yticks([-1,1])
axes.set_zticks([-1,1])
color = 'dodgerblue'
rgb = colors.colorConverter.to_rgb(color)
rgba = rgb + (alpha,)
r = [-1,1]
X, Y = np.meshgrid(r, r)
axes.scatter3D(points[:, 0], points[:, 1], points[:, 2])
verts = [[points[0],points[1],points[2]],
         [points[1],points[2],points[3]],
         [points[2],points[3],points[4]],
         [points[2],points[4],points[0]],
         [points[0],points[1],points[5]],
         [points[1],points[3],points[5]],
         [points[3],points[4],points[5]],
         [points[4],points[0],points[5]]]
axes.add_collection3d(Poly3DCollection(verts, facecolors=rgba, edgecolors='r'))
axes.set_aspect('equal')
plt.show()
```

-:--- **octaedre.py** Bot L40 (Python)

Wrote /home/vella-chemla/Desktop/solides-de-platon/octaedre.py

Figure 1



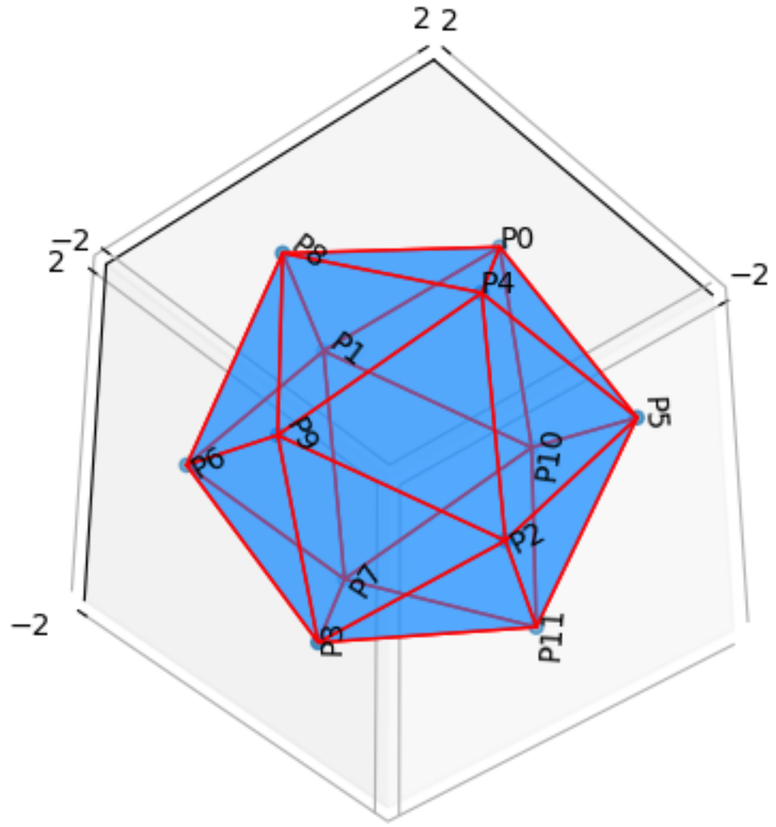
emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
import numpy as np
from matplotlib import colors
from mpl_toolkits.mplot3d import Axes3D
from mpl_toolkits.mplot3d.art3d import Poly3DCollection, Line3DCollection
import matplotlib.pyplot as plt

points = np.array([[0, 1, 1.618034],
                   [0, -1, 1.618034],
                   [0, 1, -1.618034],
                   [0, -1, -1.618034],
                   [1, 1.618034, 0],
                   [-1, 1.618034, 0],
                   [1, -1.618034, 0],
                   [-1, -1.618034, 0],
                   [1.618034, 0, 1],
                   [1.618034, 0, -1],
                   [-1.618034, 0, 1],
                   [-1.618034, 0, -1]])
zdirs = (None, 'x', 'y', 'z', (0,0,0),(0,0,1),(0,1,0),(0,1,1),(1,0,0),(1,0,1),(1
,1,0),(1,1,1))
xs = (0,0,0,0,1,-1,1,-1,1.61,1.61,-1.61,-1.61)
ys = (1,-1,1,-1,1.61,1.61,-1.61,-1.61,0,0,0,0)
zs = (1.61,1.61,-1.61,-1.61,0,0,0,0,1,-1,1,-1)

alpha = 0.5
fig = plt.figure()
axes = fig.gca(projection='3d')
for zdir, x, y, z, i in zip(zdirs, xs, ys, zs, range(12)):
    label = 'P%d' % (i)
    axes.text(x, y, z, label, zdir)
axes.set_xticks([-2,2])
axes.set_yticks([-2,2])
axes.set_zticks([-2,2])
```

-:--- **icosaedre.py** Top L31 (Python)

Figure 1



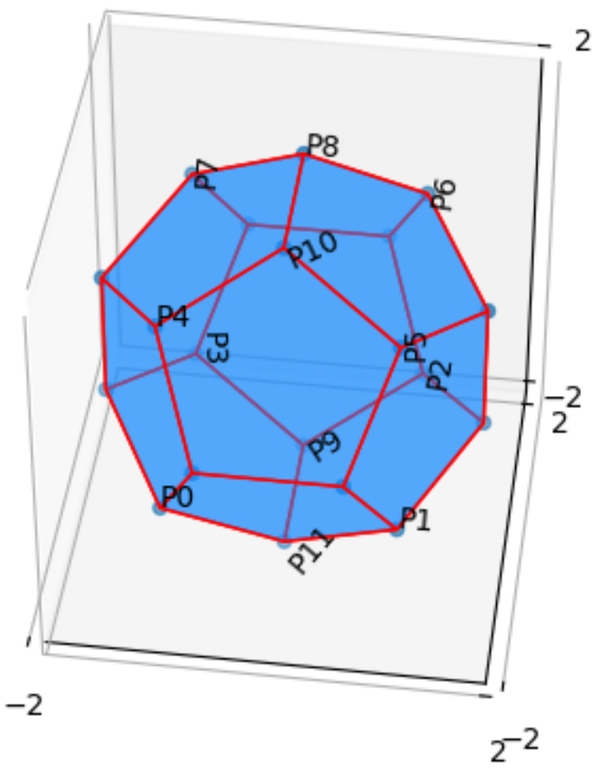
emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
color = 'dodgerblue'
rgb = colors.colorConverter.to_rgb(color)
rgba = rgb + (alpha,)
r = [-1,1]
X, Y = np.meshgrid(r, r)
axes.scatter3D(points[:, 0], points[:, 1], points[:, 2])
verts = [[points[7],points[11],points[10]],
         [points[10],points[5],points[11]],
         [points[8],points[6],points[9]],
         [points[8],points[4],points[9]],
         [points[2],points[4],points[5]],
         [points[2],points[4],points[9]],
         [points[5],points[2],points[11]],
         [points[8],points[6],points[1]],
         [points[1],points[6],points[7]],
         [points[7],points[1],points[10]],

         [points[0],points[5],points[10]],
         [points[0],points[4],points[5]],
         [points[0],points[8],points[4]],
         [points[0],points[1],points[8]],
         [points[0],points[10],points[1]],
         [points[11],points[7],points[3]],
         [points[2],points[11],points[3]],
         [points[9],points[2],points[3]],
         [points[6],points[9],points[3]],
         [points[7],points[6],points[3]]]
axes.add_collection3d(Poly3DCollection(verts, facecolors=rgba, edgecolors='r'))
axes.set_aspect('equal')
plt.show()
```

-:--- **icosaedre.py** Bot L48 (Python)

Figure 1



emacs25@vellachemla-X510UA

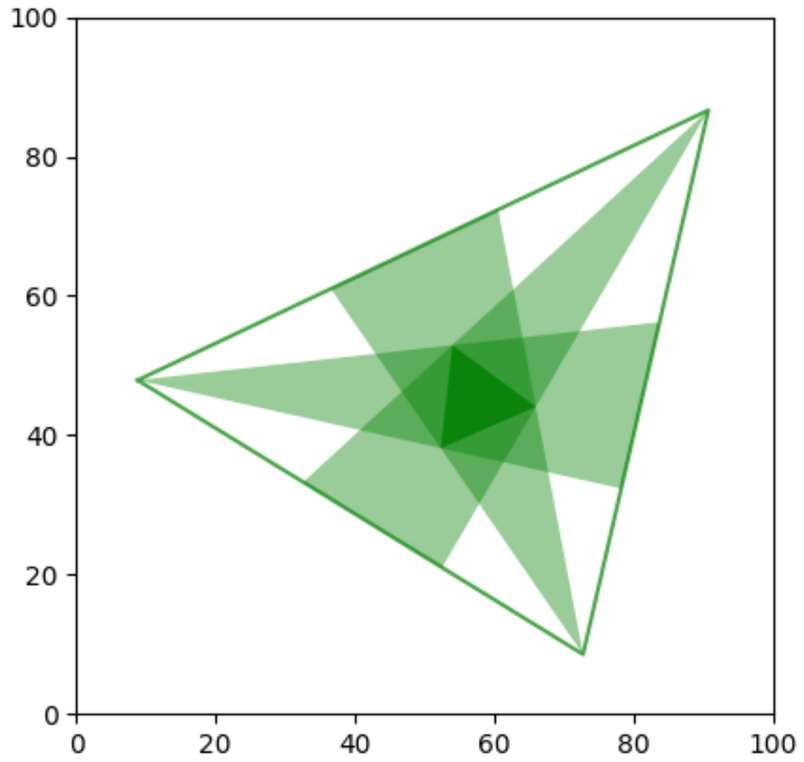
File Edit Options Buffers Tools Python Help

```
color = 'dodgerblue'
rgb = colors.colorConverter.to_rgb(color)
rgba = rgb + (alpha,)
r = [-1,1]
X, Y = np.meshgrid(r, r)
axes.scatter3D(points[:, 0], points[:, 1], points[:, 2])
verts = [[points[5],points[10],points[8],points[6],points[16]],
         [points[0],points[19],points[3],points[9],points[11]],
         [points[5],points[13],points[1],points[18],points[16]],
         [points[4],points[17],points[7],points[8],points[10]],
         [points[4],points[10],points[5],points[13],points[15]],
         [points[6],points[8],points[7],points[14],points[12]],
         [points[14],points[12],points[2],points[9],points[3]],
         [points[1],points[18],points[2],points[9],points[11]],
         [points[6],points[12],points[2],points[18],points[16]],
         [points[7],points[14],points[3],points[19],points[17]],
         [points[1],points[11],points[0],points[15],points[13]],
         [points[4],points[15],points[0],points[19],points[17]]]

zdirs = (None, 'x', 'y', 'z', (0,0,0),(0,0,1),(0,1,0),(0,1,1),(1,0,0),(1,0,1),(1,1,0),(1,1,1))
xs = (-1,1,1,-1,-1,1,1,-1,0,0,0,0,0.61,0.61,-0.61,-0.61,1.61,-1.61,1.61,-1.61)
ys = (-1,-1,1,1,-1,-1,1,1,0.61,0.61,-0.61,-0.61,1.61,-1.61,1.61,-1.61,0,0,0,0)
zs = (-1,-1,-1,-1,1,1,1,1,1.61,-1.61,1.61,-1.61,0,0,0,0,0.61,0.61,-0.61,-0.61)
for zdir, x, y, z, i in zip(zdirs, xs, ys, zs, range(20)):
    label = 'P%d' % (i)
    axes.text(x, y, z, label, zdir)

axes.add_collection3d(Poly3DCollection(verts, facecolors=rgba, edgecolors='r'))
axes.set_aspect('equal')
plt.show()
```

-:**- **dodecaedre.py** Bot L47 (Python)

Activités GNU Emacs 25 (GUI) mar. 09:51

Figure 1




```
vella-chemla@vellachemla-X510UA:~/Desktop$
vella-chemla@vellachemla-X510UA:~/Desktop$
vella-chemla@vellachemla-X510UA:~/Desktop$
vella-chemla@vellachemla-X510UA:~/Desktop$
vella-chemla@vellachemla-X510UA:~/Desktop$
vella-chemla@vellachemla-X510UA:~/Desktop$
vella-chemla@vellachemla-X510UA:~/Desktop$ python Morley-dernier.py
Normes des cotes 14.863746806168091
Normes des cotes 14.863746806168082
Normes des cotes 14.863746806168091
```

emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

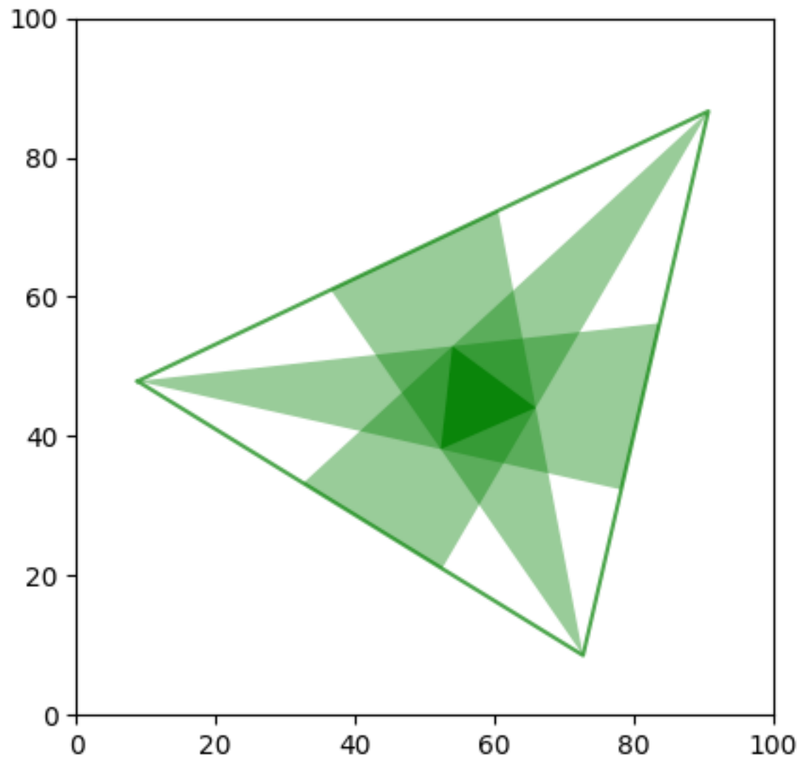
```
from math import *
from matplotlib import *
from matplotlib.pyplot import *

def vecteur(point1, point2):
  return [y - x for x, y in zip(point1, point2)]

def add(vecteur1, vecteur2):
  return [x + y for x, y in zip(vecteur1, vecteur2)]

def norme(vecteur):
  return sqrt(prodscal(vecteur, vecteur))

def prodscal(vecteur1, vecteur2):
  return sum([x*y for x, y in zip(vecteur1, vecteur2)])

def determ(vecteur1, vecteur2):
 return vecteur1[0]*vecteur2[1]-vecteur1[1]*vecteur2[0]

def angle(vecteur1, vecteur2):
  cosinus=prodscal(vecteur1, vecteur2)/(norme(vecteur1)*norme(vecteur2))
  sinus=determ(vecteur1, vecteur2)/(norme(vecteur1)*norme(vecteur2))
  return atan2(sinus,cosinus)

def rotation(u, theta):
  return [u[0]*cos(theta)-u[1]*sin(theta),u[0]*sin(theta)+u[1]*cos(theta)]

def intersecte(x, y, z, t):
  a1 = (y[1]-x[1])/(y[0]-x[0])
  b1 = x[1]-a1*x[0]
  a2 = (t[1]-z[1])/(t[0]-z[0])
  b2 = z[1]-a2*z[0]
  return [(b2-b1)/(a1-a2),((b2-b1)/(a1-a2))*a1+b1]

a=[8.83, 47.89]
```

-:--- **Morley-dernier.py** Top L1 (Python)

5 1

Activités GNU Emacs 25 (GUI) mar. 09:51

Figure 1




```
vella-chemla@vellachemla-X510UA:~/Desktop$
vella-chemla@vellachemla-X510UA:~/Desktop$
vella-chemla@vellachemla-X510UA:~/Desktop$
vella-chemla@vellachemla-X510UA:~/Desktop$
vella-chemla@vellachemla-X510UA:~/Desktop$
vella-chemla@vellachemla-X510UA:~/Desktop$
vella-chemla@vellachemla-X510UA:~/Desktop$ python Morley-dernier.py
Normes des cotes 14.863746806168091
Normes des cotes 14.863746806168082
Normes des cotes 14.863746806168091
```

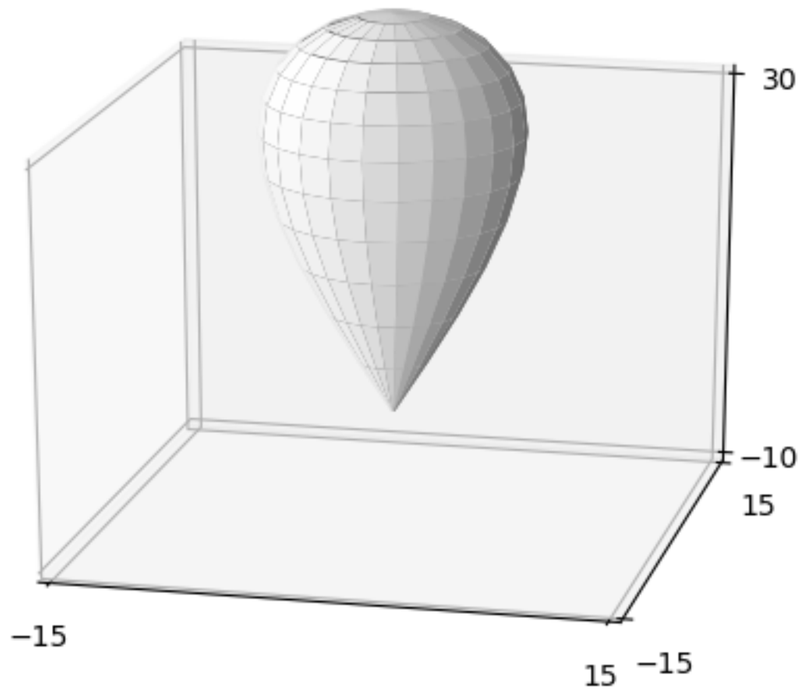
emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
a=[8.83, 47.89]
b=[72.74, 8.55]
c=[90.72, 86.63]
ab = vecteur(a,b) ; ac = vecteur(a,c)
ba = vecteur(b,a) ; bc = vecteur(b,c)
ca = vecteur(c,a) ; cb = vecteur(c,b)
anglea=angle(ab,ac)  ; angleb=angle(bc,ba)  ; anglec=angle(ca,cb)
aprime = add(a, rotation(ab,anglea/3.0))  ; aseconde = add(a, rotation(ab,anglea*2.0/3.0))
bprime = add(b, rotation(bc,angleb/3.0)) ; bseconde = add(b, rotation(bc,angleb*2.0/3.0))
cprime = add(c, rotation(ca,anglec/3.0)) ; cseconde = add(c, rotation(ca,anglec*2.0/3.0))
p=intersecte(a,aseconde,b,c) ; q=intersecte(a,aprime,b,c)
r=intersecte(c,cseconde,a,b) ; s=intersecte(c,cprime,a,b)
t=intersecte(b,bseconde,c,a) ; u=intersecte(b,bprime,c,a)
n=intersecte(a,aseconde,c,cprime)
o=intersecte(c,cseconde,b,bprime)
x=intersecte(b,bseconde,a,aprime)
print('Normes des cotes %3.15f '% norme(vecteur(n,o)))
print('Normes des cotes %3.15f '% norme(vecteur(o,x)))
print('Normes des cotes %3.15f '% norme(vecteur(x,n)))

fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
matplotlib.pyplot.plot([a[0],b[0],c[0],a[0]],[a[1],b[1],c[1],a[1]], 'g', alpha=0.7)
matplotlib.pyplot.fill([a[0],p[0],q[0]],[a[1],p[1],q[1]], 'g', 2, alpha=0.4)
matplotlib.pyplot.fill([c[0],r[0],s[0]],[c[1],r[1],s[1]], 'g', 2, alpha=0.4)
matplotlib.pyplot.fill([b[0],t[0],u[0]],[b[1],t[1],u[1]], 'g', 2, alpha=0.4)
matplotlib.pyplot.fill([n[0],o[0],x[0]],[n[1],o[1],x[1]], 'g', 2, alpha=0.8)
matplotlib.pyplot.xlim(0,100)
matplotlib.pyplot.ylim(0,100)
ax.set_aspect('equal')
matplotlib.pyplot.show()
```

-:--- **Morley-dernier.py** Bot L52 (Python)

Figure 1



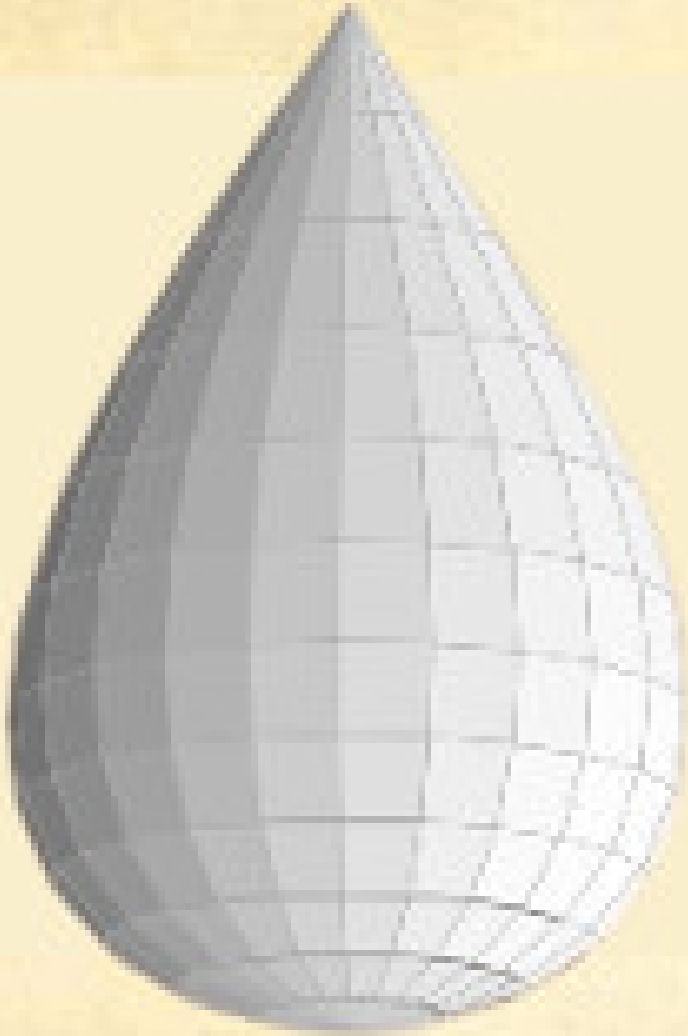
emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
import math
#from matplotlib import colors

alpha = 0.3
r, R = 40, 0.5
#color = 'dodgerblue'
#rgb = colors.colorConverter.to_rgb(color)
#rgba = rgb + (alpha,)
u = np.linspace(0, 2 * np.pi, 25)
v = np.linspace(0, np.pi, 25)
theta, phi = np.meshgrid(u,v)
k = 1./(2.*math.sqrt(2.))
X = (k/2)*(R + r*np.sin(2*phi)) * np.cos(theta)
Y = (k/2)*(R + r*np.sin(2*phi)) * np.sin(theta)
Z = R+r*np.sin(phi)

fig = plt.figure()
ax = fig.gca(projection = '3d')
ax.set_xlim3d(-15, 15)
ax.set_ylim3d(-15, 15)
ax.set_zlim3d(-10, 30)
ax.set_xticks([-15,15])
ax.set_yticks([-15,15])
ax.set_zticks([-10,30])
ax.plot_surface(X, Y, Z, color = 'g')
plt.show()
```

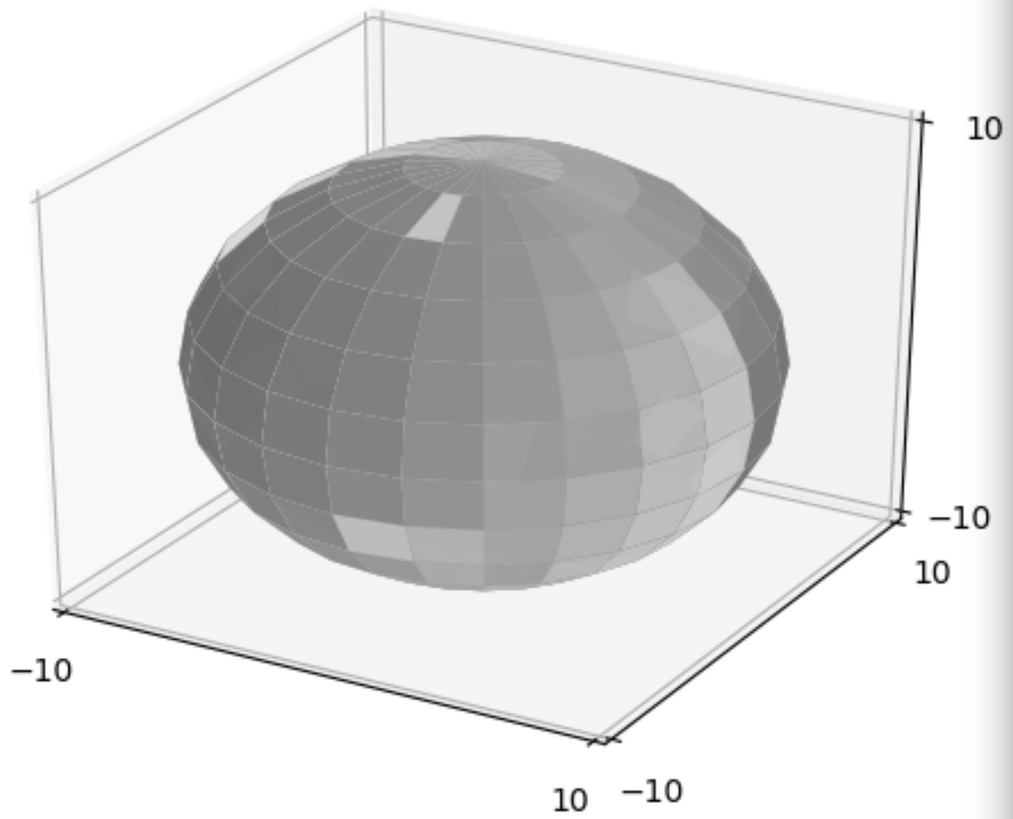
-:--- **larme.py** All L28 (Python)

Wrote /home/vella-chemla/Desktop/larme.py



Ceci n'est pas une poire.

Figure 1



emacs25@vellachemla-X510UA

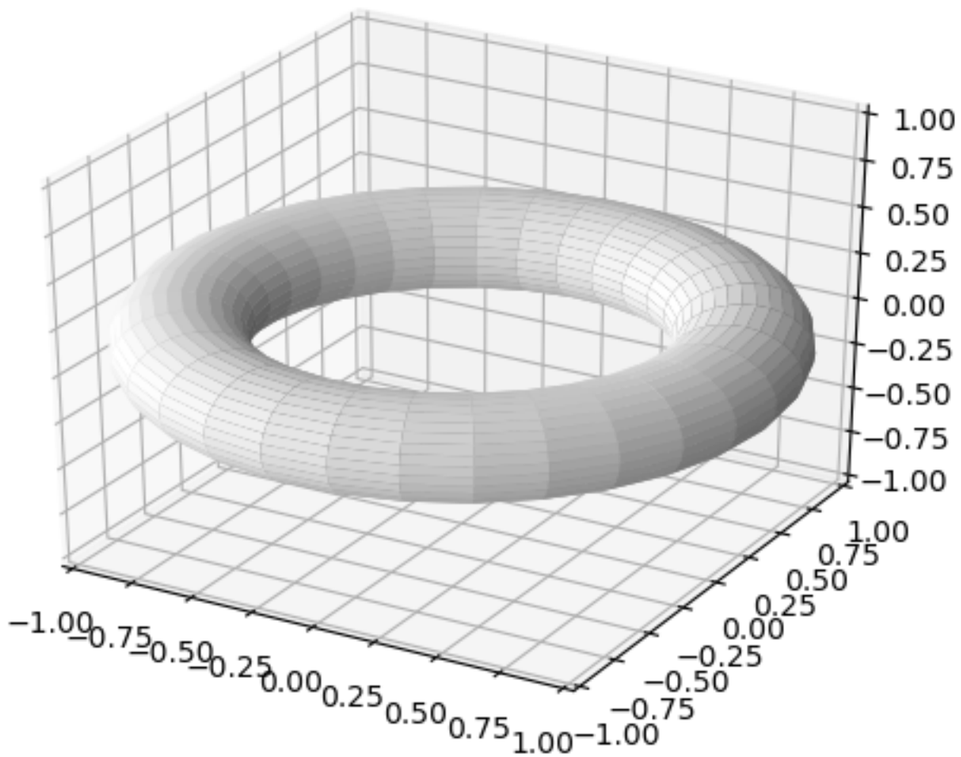
File Edit Options Buffers Tools Python Help

```
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
import math
from matplotlib import colors

alpha = 0.75
r, R = 10, 0.5
color = 'gray'
rgb = colors.colorConverter.to_rgb(color)
rgba = rgb + (alpha,)
u = np.linspace(0, 2*np.pi, 25)
v = np.linspace(0, 2*np.pi, 25)
theta, phi = np.meshgrid(u,v)
X = r * np.cos(phi) * np.cos(theta)
Y = r * np.sin(phi) * np.cos(theta)
Z = r * np.sin(theta)

fig = plt.figure()
ax = fig.gca(projection = '3d')
ax.set_xlim3d(-10, 10)
ax.set_ylim3d(-10, 10)
ax.set_zlim3d(-10, 10)
ax.set_xticks([-10,10])
ax.set_yticks([-10,10])
ax.set_zticks([-10,10])
ax.plot_surface(X, Y, Z, color = 'w', alpha = 0.8)
plt.show()
```

-:--- **sphere.py** All L27 (Python)

Wrote /home/vella-chemla/Desktop/la-geometre/sphere.py

Figure 1



emacs25@vellachemla-X510UA

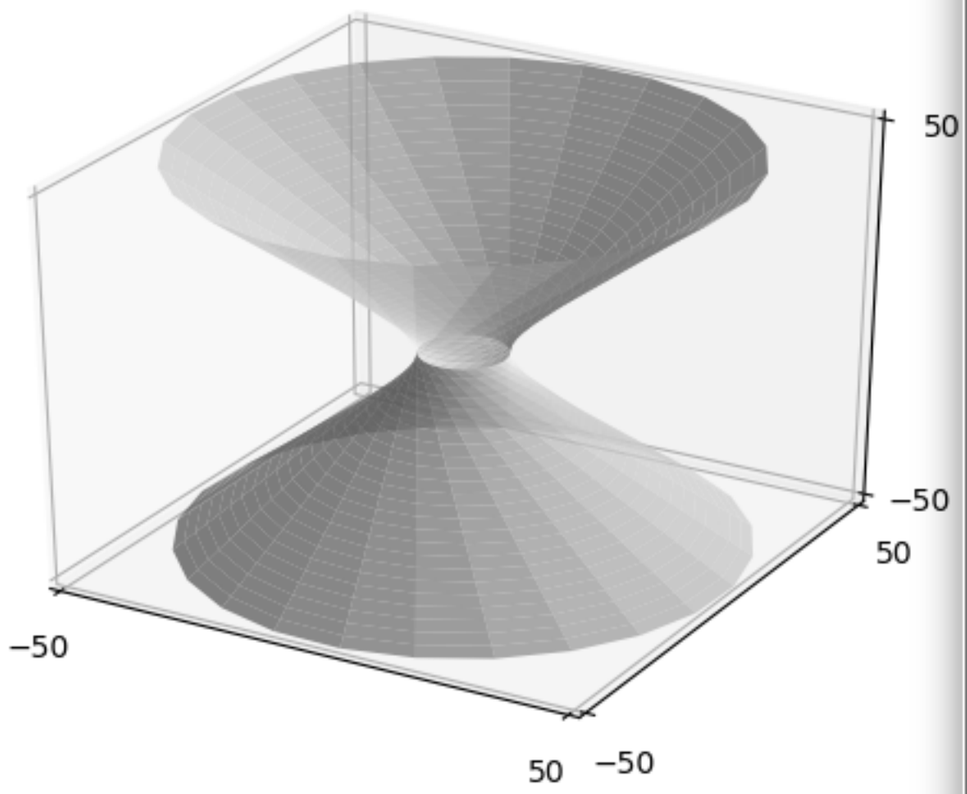
File Edit Options Buffers Tools Python Help

```
#code trouvé sur la toile
#------------------------
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt

# Generate torus mesh
angle = np.linspace(0, 2 * np.pi, 32)
theta, phi = np.meshgrid(angle, angle)
r, R = .25, 1.
#k = 1./(2.*sqrt(2.))
X = (R + r * np.cos(phi)) * np.cos(theta)
Y = (R + r * np.cos(phi)) * np.sin(theta)
Z = r * np.sin(phi)

# Display the mesh
fig = plt.figure()
ax = fig.gca(projection = '3d')
ax.set_xlim3d(-1, 1)
ax.set_ylim3d(-1, 1)
ax.set_zlim3d(-1, 1)
ax.plot_surface(X, Y, Z, color = 'w', rstride = 1, cstride = 1)
plt.show()
```

U:--- **torus.py** All L2 (Python)

Wrote /home/vella-chemla/Desktop/la-geometre/torus.py

Figure 1



emacs25@vellachemla-X510UA

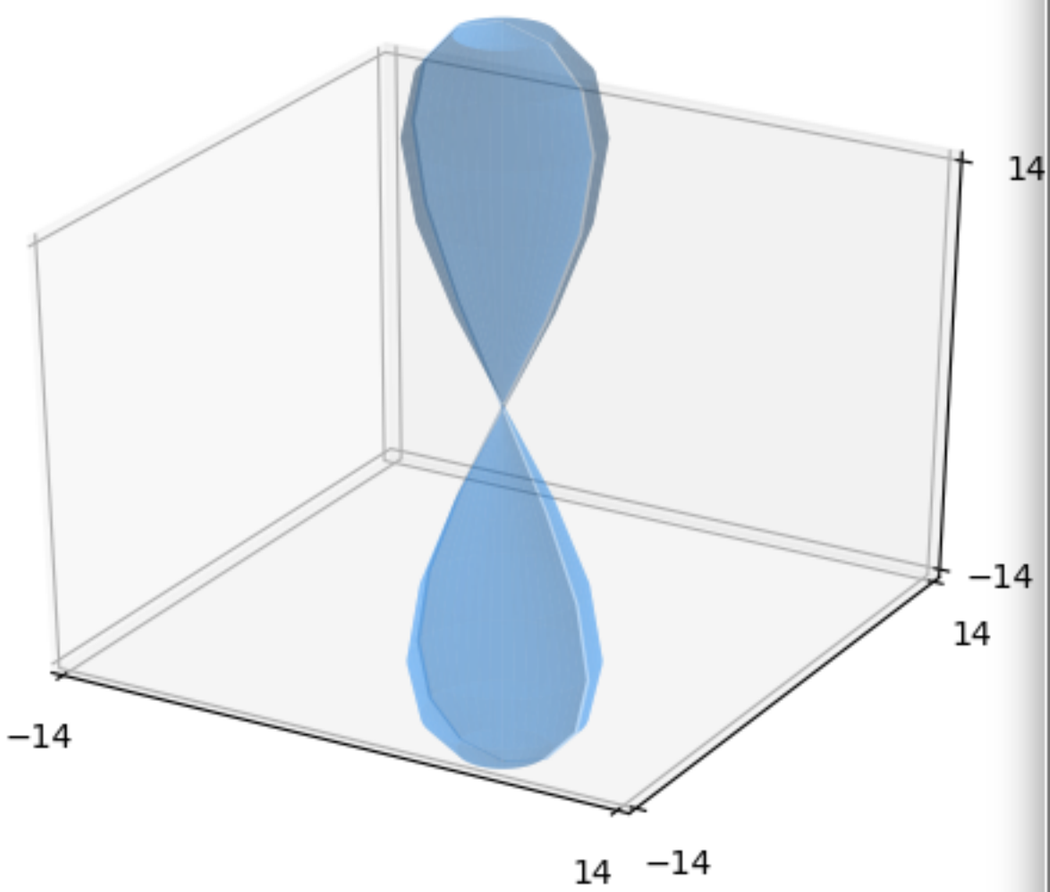
File Edit Options Buffers Tools Python Help

```
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
import math
from matplotlib import colors

alpha = 0.75
r1, r2, r3, R = 8, 8, 8, 0.5
color = 'gray'
rgb = colors.colorConverter.to_rgb(color)
rgba = rgb + (alpha,)
u = np.linspace(0, 2*np.pi, 25)
v = np.linspace(0, 2*np.pi, 25)
theta, phi = np.meshgrid(u,v)
X = r1 * (np.cos(theta) - phi*np.sin(theta))
Y = r2 * (np.sin(theta) + phi*np.cos(theta))
Z = r3 * phi

fig = plt.figure()
ax = fig.gca(projection = '3d')
ax.set_xlim3d(-50, 50)
ax.set_ylim3d(-50, 50)
ax.set_zlim3d(-50, 50)
ax.set_xticks([-50,50])
ax.set_yticks([-50,50])
ax.set_zticks([-50,50])
ax.plot_surface(X, Y, Z, color = 'w', alpha = 0.75)
ax.plot_surface(X, Y, (-1)*Z, color = 'w', alpha = 0.75)
plt.show()
```

-:--- **hyperboloide.py** All L28 (Python)

Wrote /home/vella-chemla/Desktop/la-geometre/hyperboloide.py

Figure 1



emacs25@vellachemla-X510UA

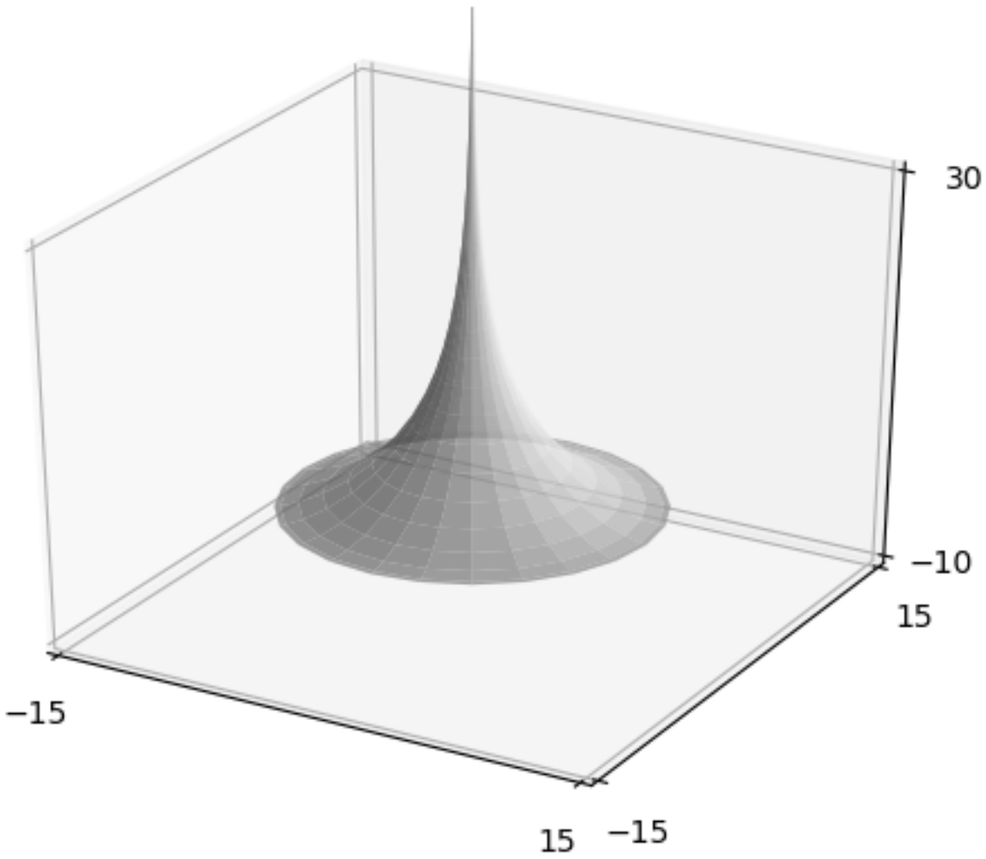
File Edit Options Buffers Tools Python Help

```
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
import math
from matplotlib import colors

alpha = 0.3
r, R = 25, 0.5
color = 'dodgerblue'
rgb = colors.colorConverter.to_rgb(color)
rgba = rgb + (alpha,)
u = np.linspace(0, 2 * np.pi, 25)
v = np.linspace(0, np.pi, 25)
theta, phi = np.meshgrid(v, u)
k = 1./(2.*math.sqrt(2.))
X = (k/2)*(R + r*np.sin(2*phi)) * np.cos(theta)
Y = (k/2)*(R + r*np.sin(2*phi)) * np.sin(theta)
Z = R+r*np.sin(phi)

fig = plt.figure()
ax = fig.gca(projection = '3d')
ax.set_xlim3d(-14, 14)
ax.set_ylim3d(-14, 14)
ax.set_zlim3d(-14,14)
ax.set_xticks([-14,14])
ax.set_yticks([-14,14])
ax.set_zticks([-14,14])
ax.plot_surface(X, Y, Z, color = rgba)
plt.show()
```

-:--- **larme.py** All L14 (Python)

Wrote /home/vella-chemla/Desktop/larme.py

Figure 1



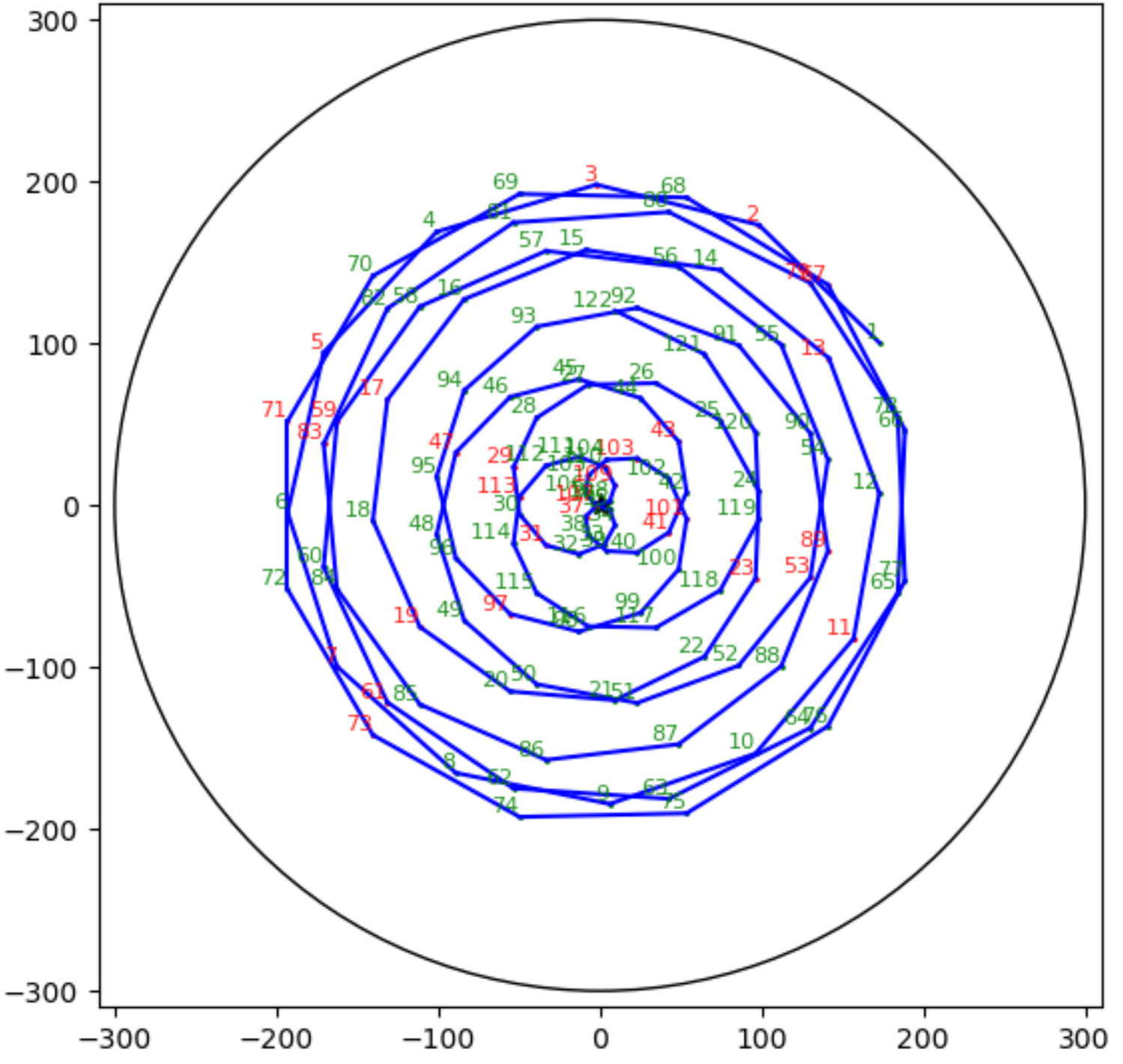
emacs25@vellachemla-X510UA

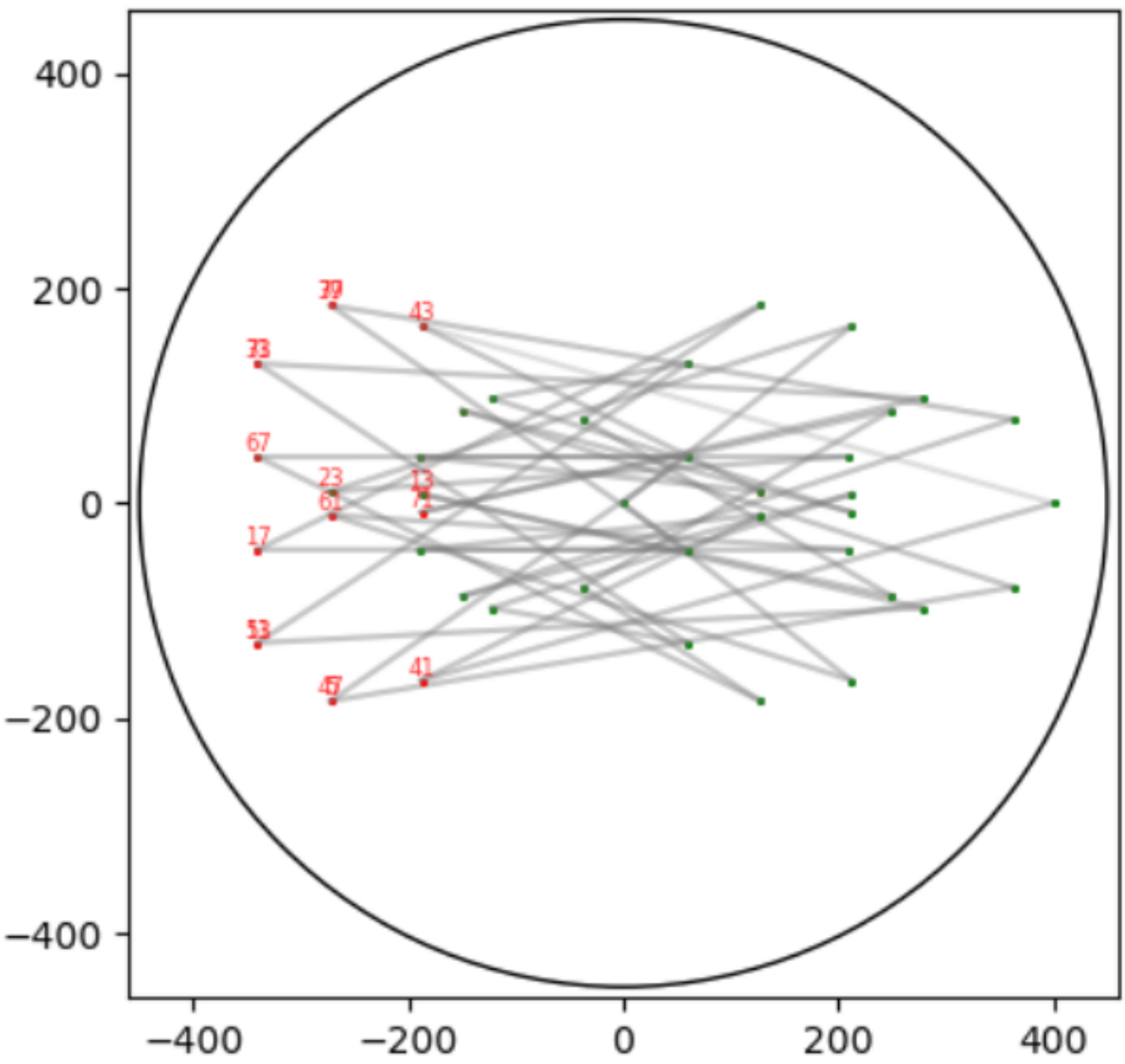
File Edit Options Buffers Tools Python Help

```
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
import math
from matplotlib import colors

alpha = 0.75
r, R = 10, 0.5
color = 'gray'
rgb = colors.colorConverter.to_rgb(color)
rgba = rgb + (alpha,)
u = np.linspace(0, 2*np.pi, 25)
v = np.linspace(0, 2*np.pi, 25)
theta, phi = np.meshgrid(u,v)
X = r*np.cos(phi) / np.cosh(theta)
Y = r*np.sin(phi) / np.cosh(theta)
Z = r*(theta-np.tanh(theta))

fig = plt.figure()
ax = fig.gca(projection = '3d')
ax.set_xlim3d(-15, 15)
ax.set_ylim3d(-15, 15)
ax.set_zlim3d(-10, 30)
ax.set_xticks([-15,15])
ax.set_yticks([-15,15])
ax.set_zticks([-10,30])
ax.plot_surface(X, Y, Z, color = 'w', alpha = 0.75)
plt.show()
```

-:--- **Beltrami.py** All L8 (Python)

Wrote /home/vella-chemla/Desktop/la-geometre/Beltrami.py


300
200
100
0
−100
−200
−300
−300
−200
−100
0
100
200
300



$$n = 84;\ E = \{2,2,3,7\}$$

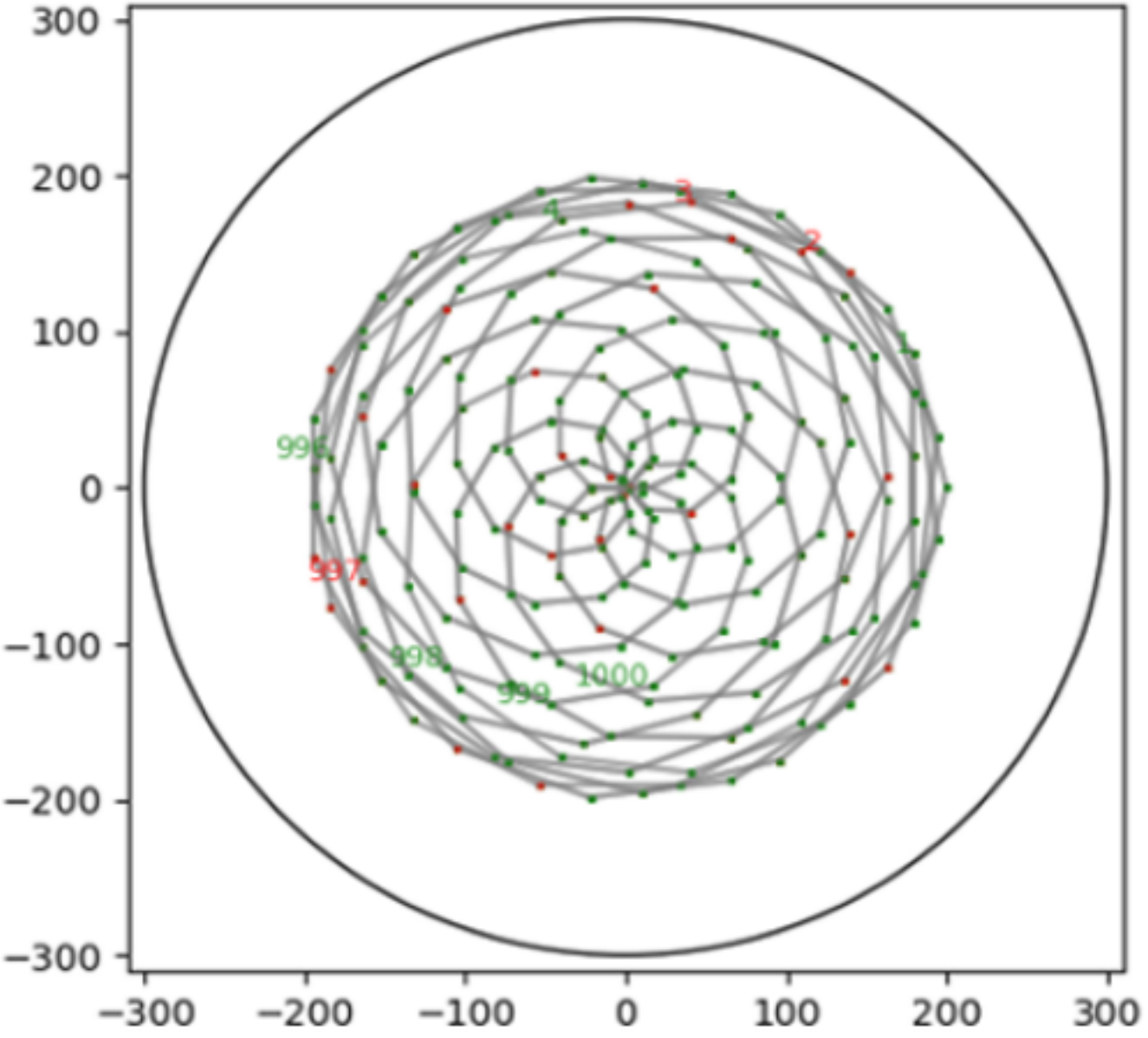
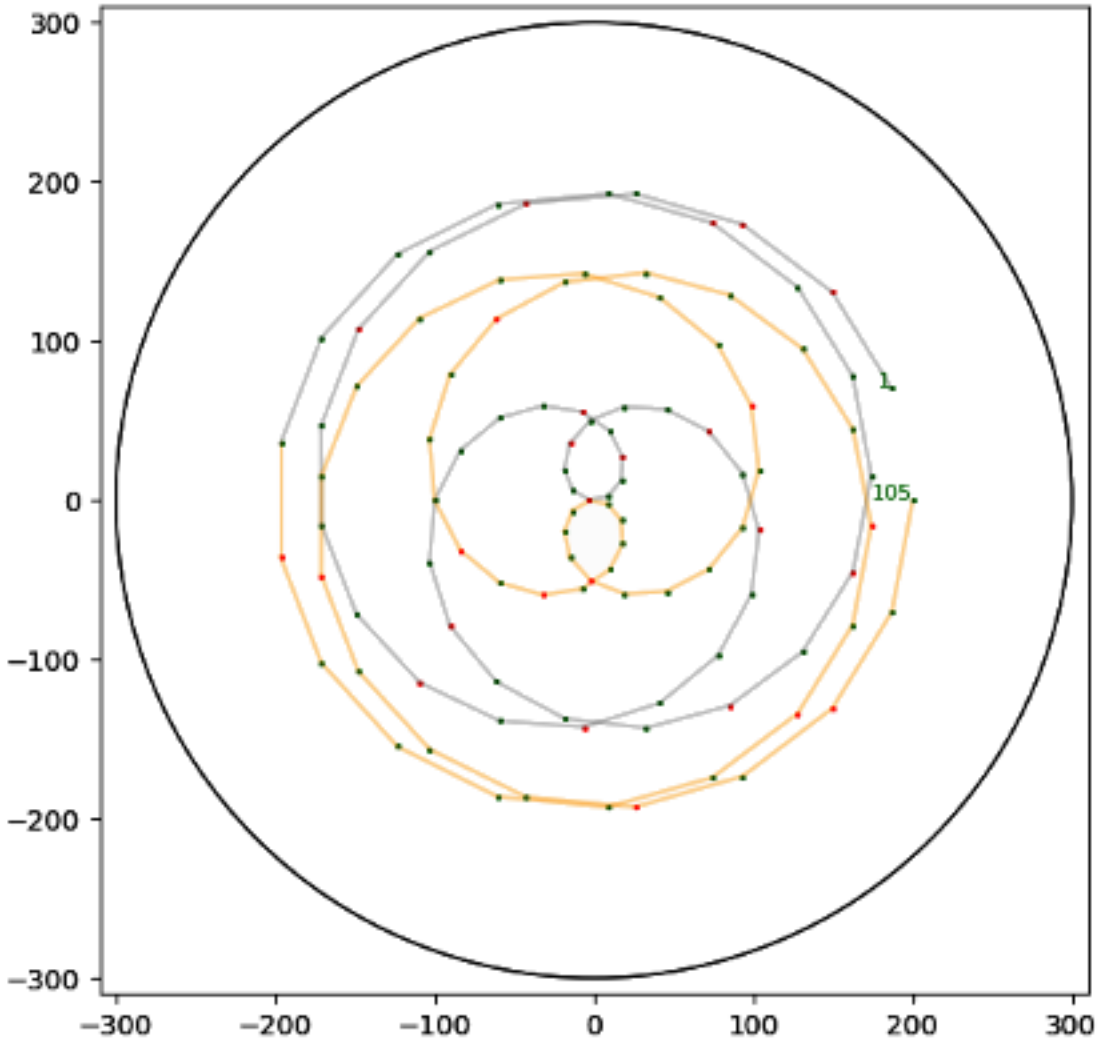
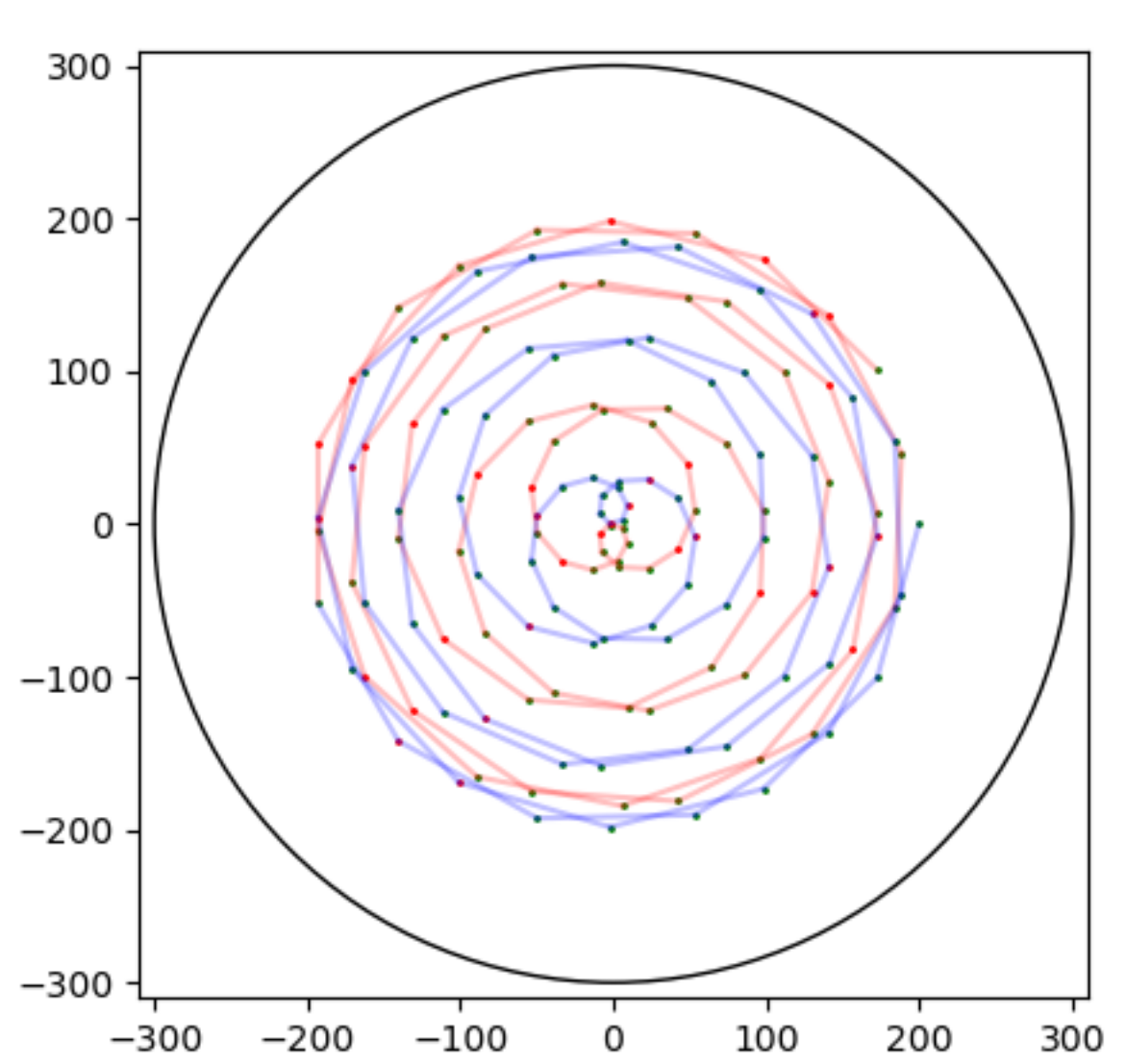
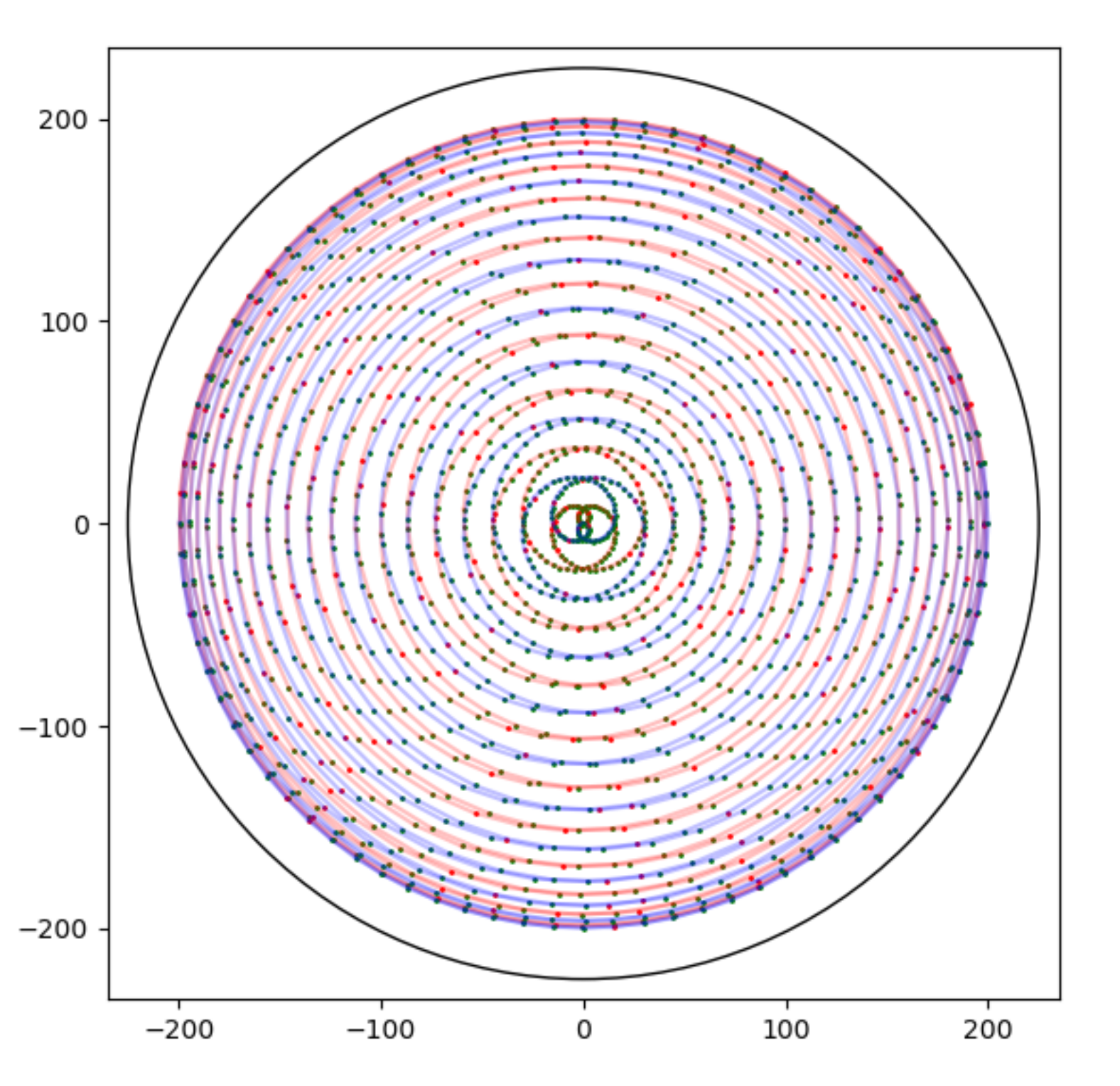
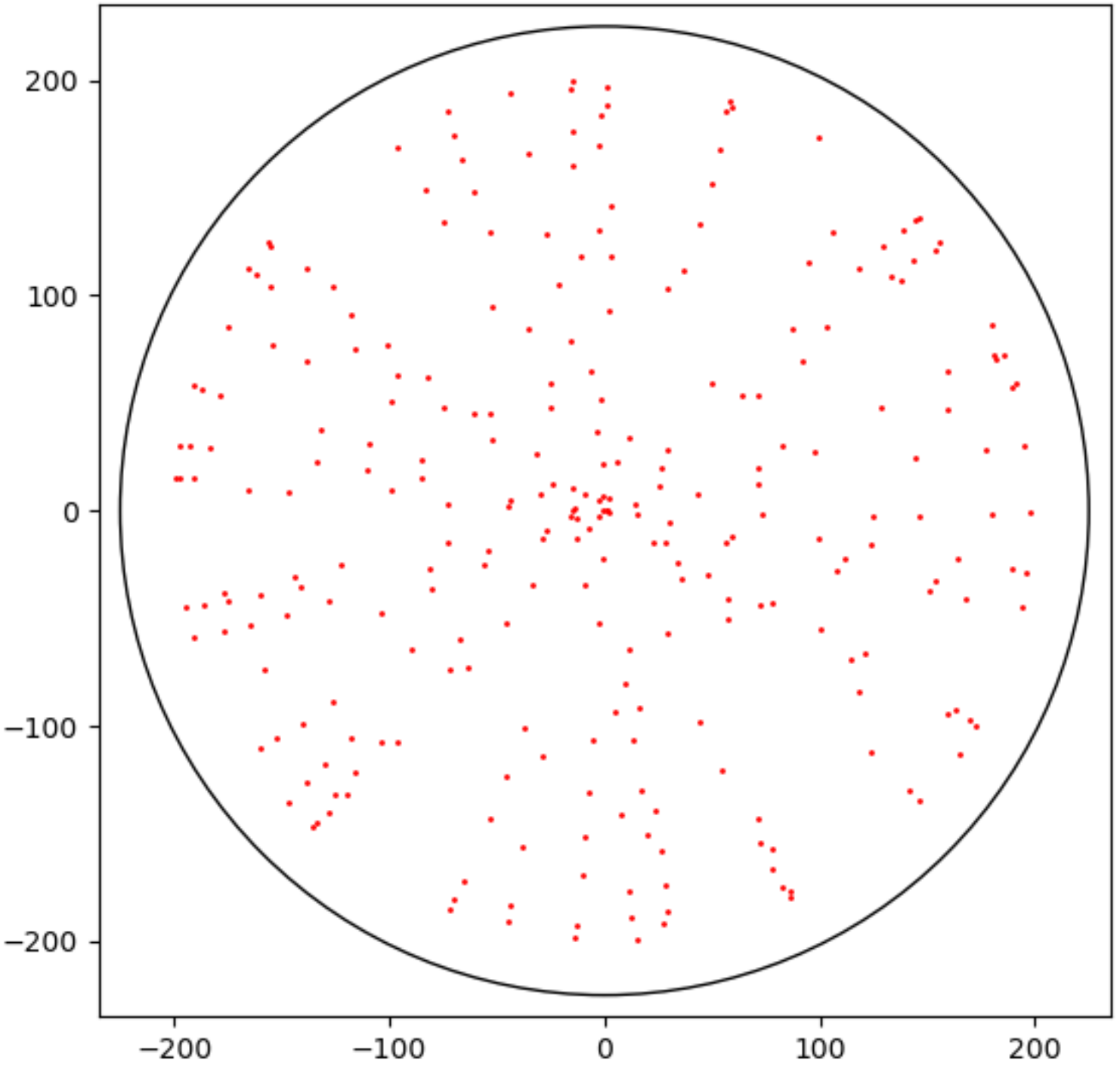


Fig. 5 : E={11,19} périodicité 8.


300
200
100
0
−100
−200
−300
−300
−200
−100
0
100
200
300
1
105


300
200
100
0
−100
−200
−300
−300
−200
−100
0
100
200
300


200
100
0
−100
−200
−200
−100
0
100
200

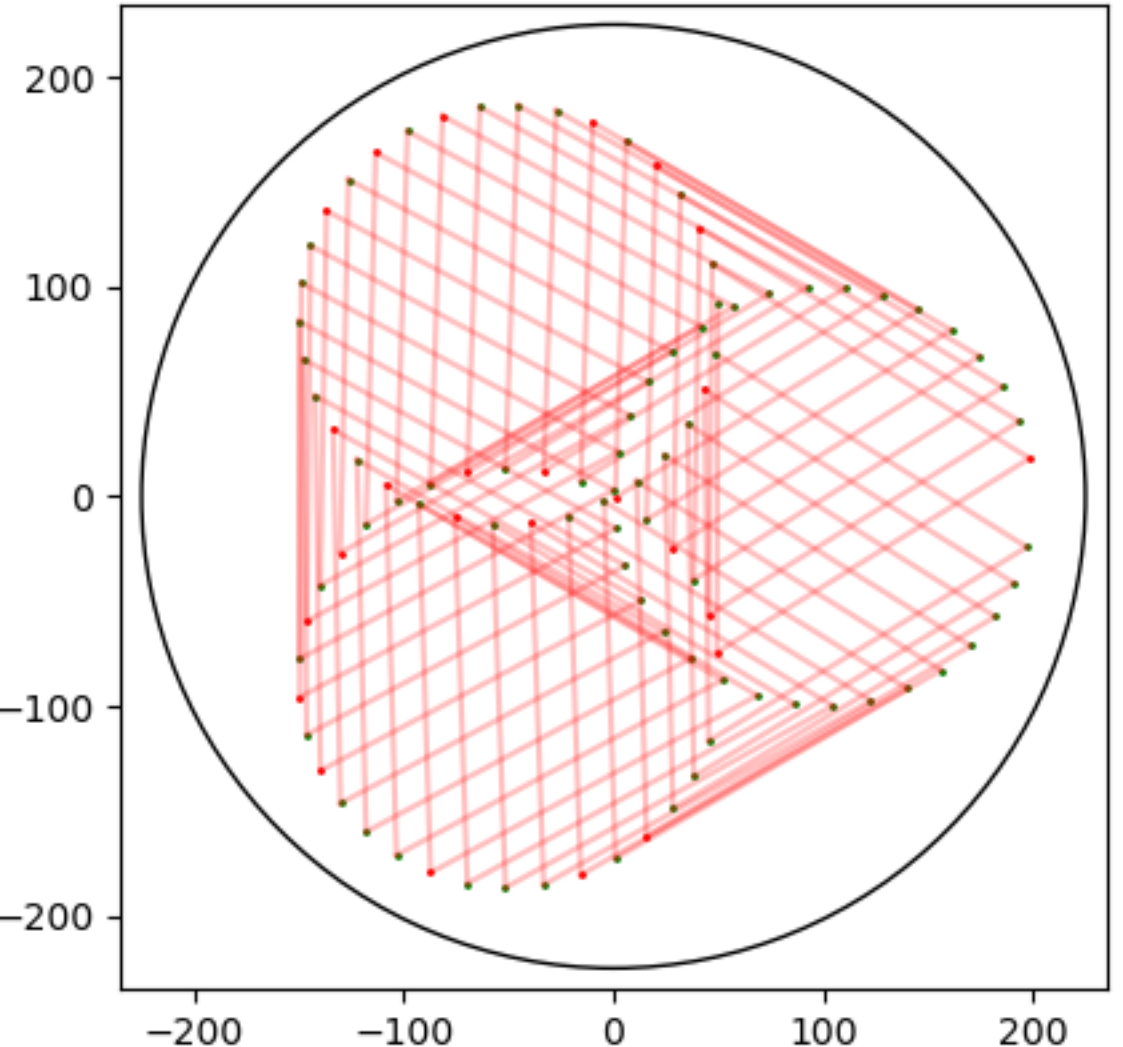

200
100
0
−100
−200
−200
−100
0
100
200

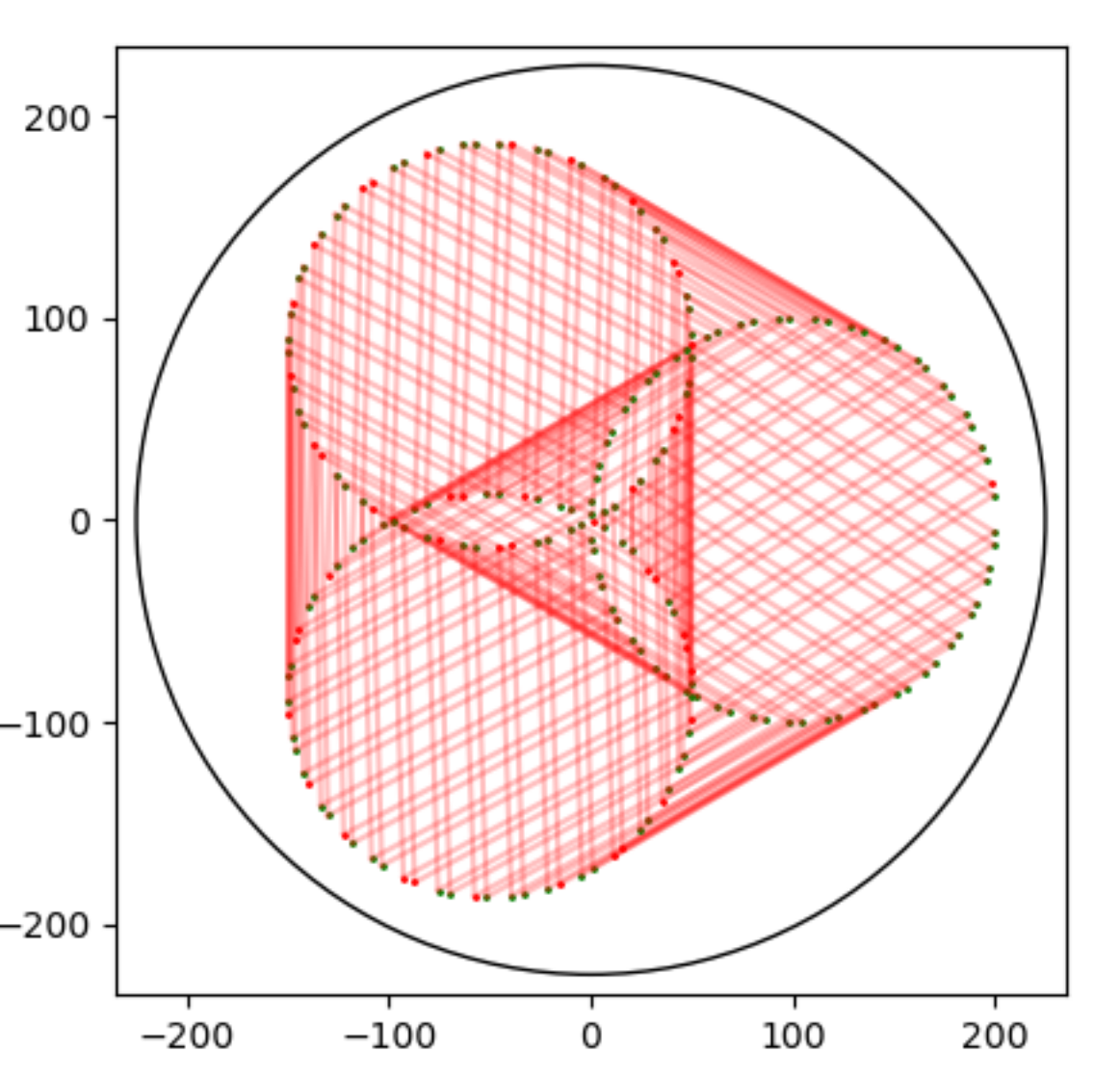
emacs25@vellachemla-X510UA

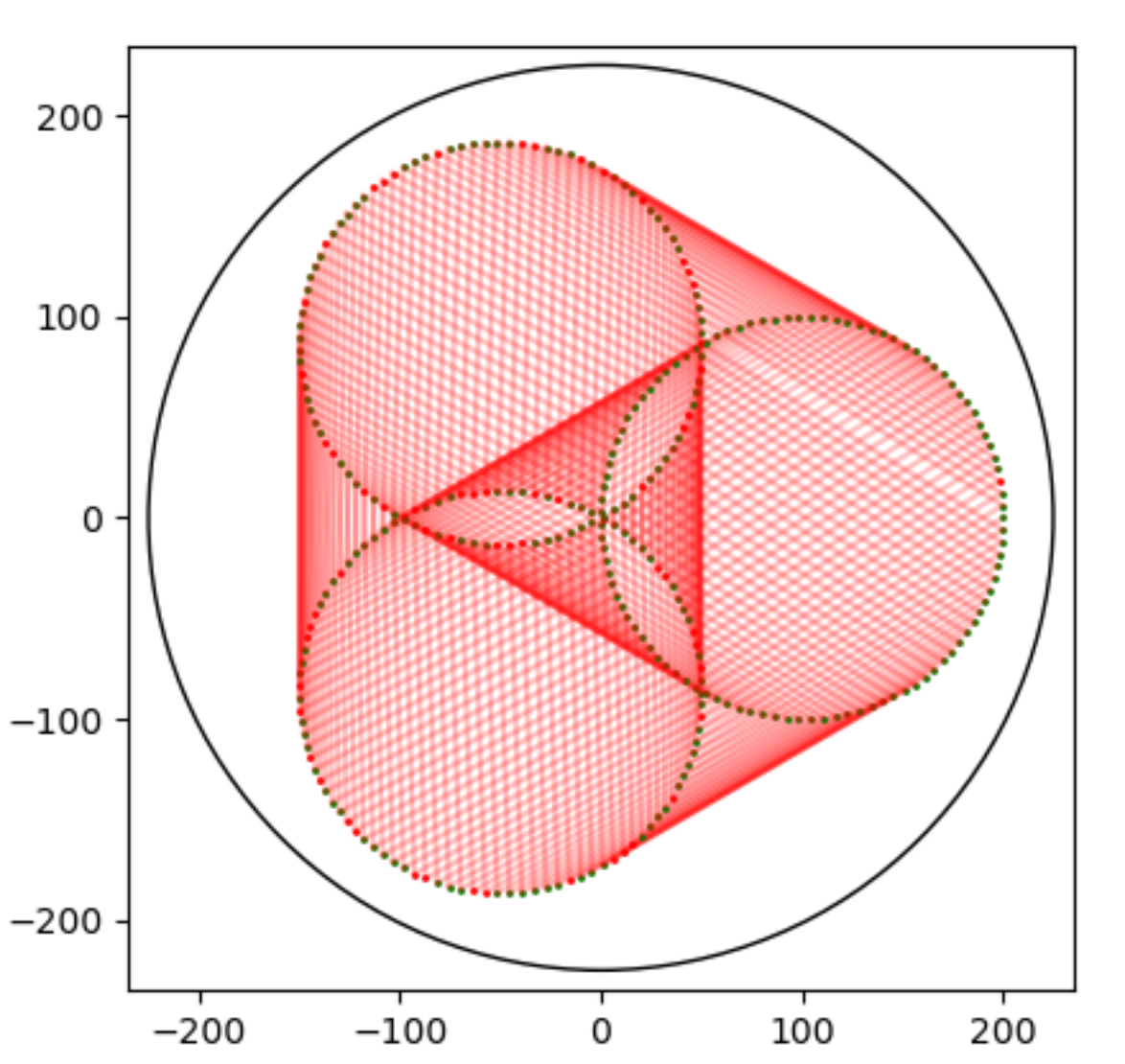
File Edit Options Buffers Tools Python Help

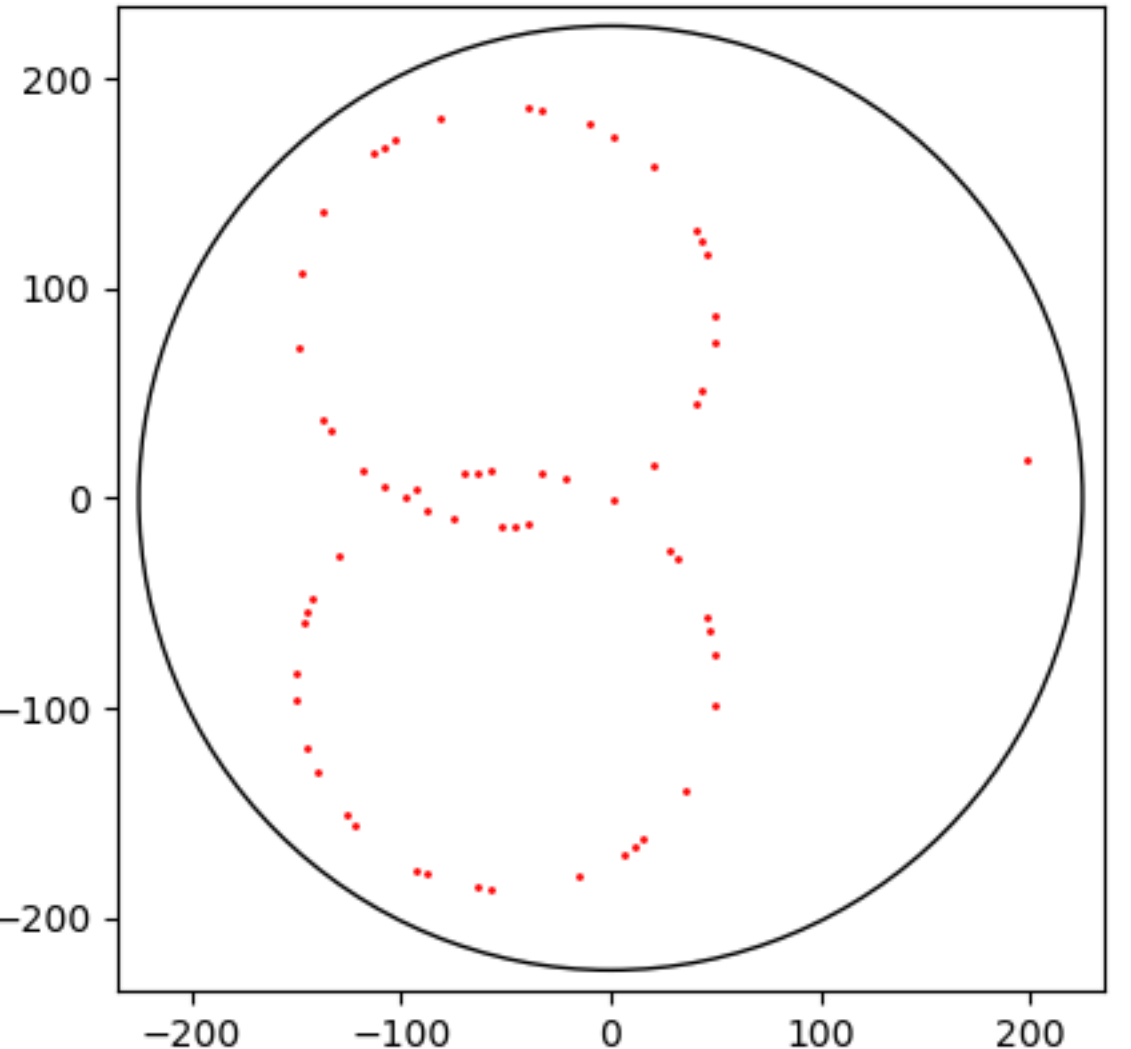
```
n = 42*42 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [41,43]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)

#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= n:
  x0, y0 = 0, 0
  for element in Ens:
    t = 2*pi*(x % element)/element
    x0 = x0 + xmax*math.cos(t)
    y0 = y0 + xmax*math.sin(t)
  #print(str(x0)+'   '+str(y0))
  if (prime(x)):
    plt.plot(x0, y0, 'r', marker='o', markersize=1)
    #ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
  else:
    plt.plot(x0, y0, 'g', marker='o', markersize=1)
    #ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
  if (x != 1):
    if (x <= 21*42):
      ax.plot([xprec,x0],[yprec,y0],'red', alpha=0.25)
    else:
      ax.plot([xprec,x0],[yprec,y0],'blue', alpha=0.25)
  xprec = x0
  yprec = y0
  ax.set_aspect('equal')
  x = x+1
plt.show()
```

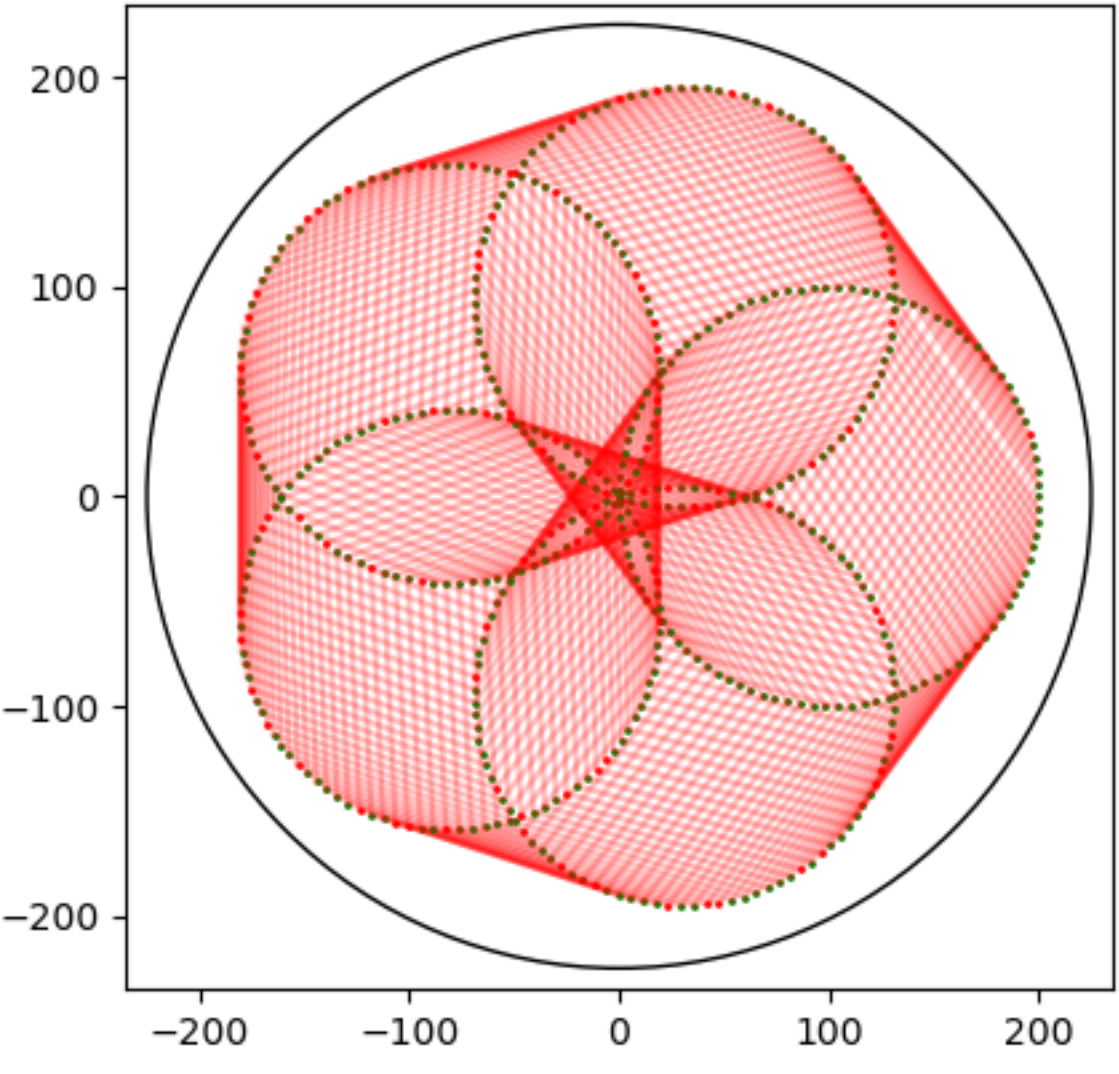
-:--- **test.py** 28% L18 (Python)

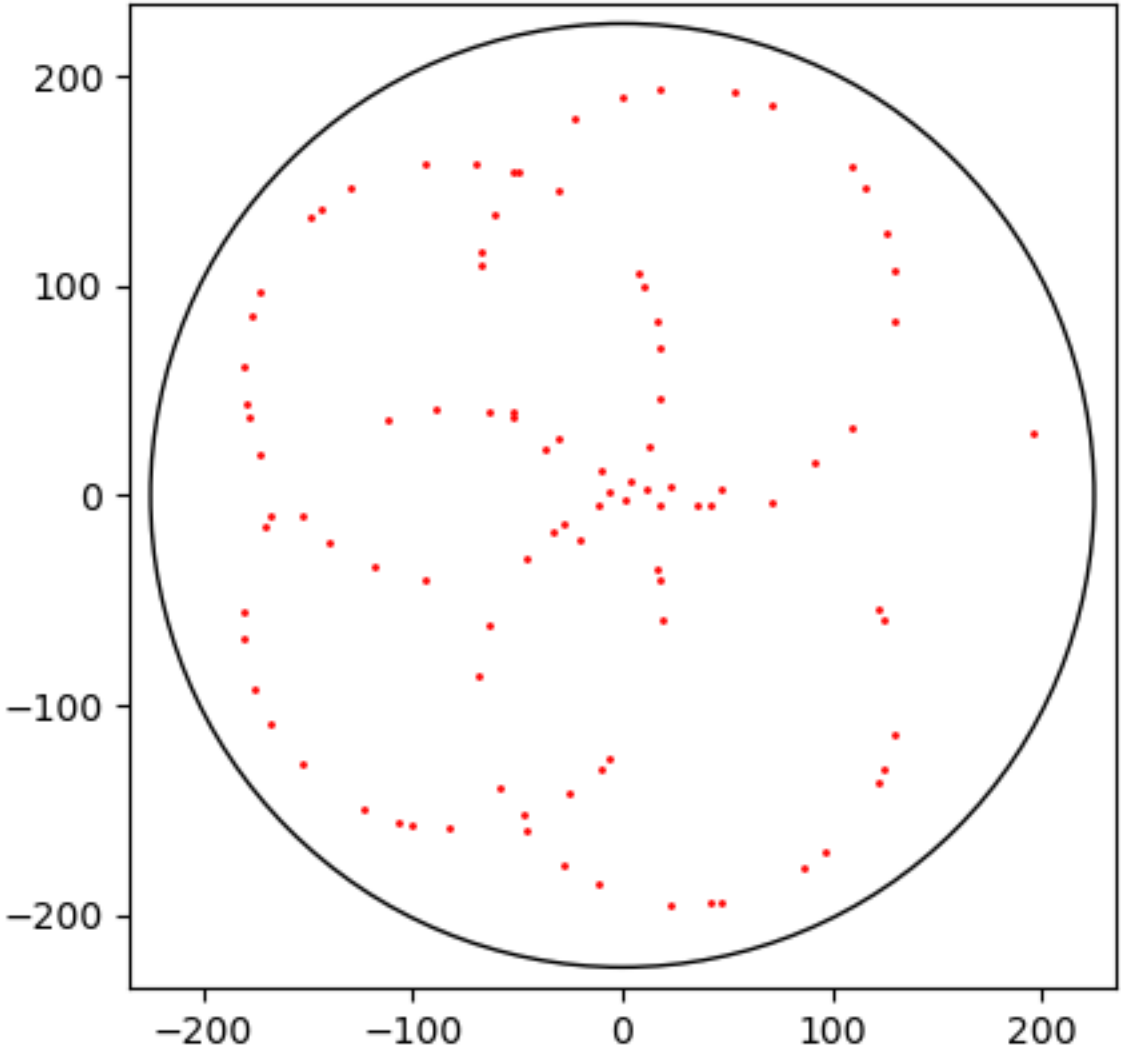

200
100
0
−100
−200
−200
−100
0
100
200

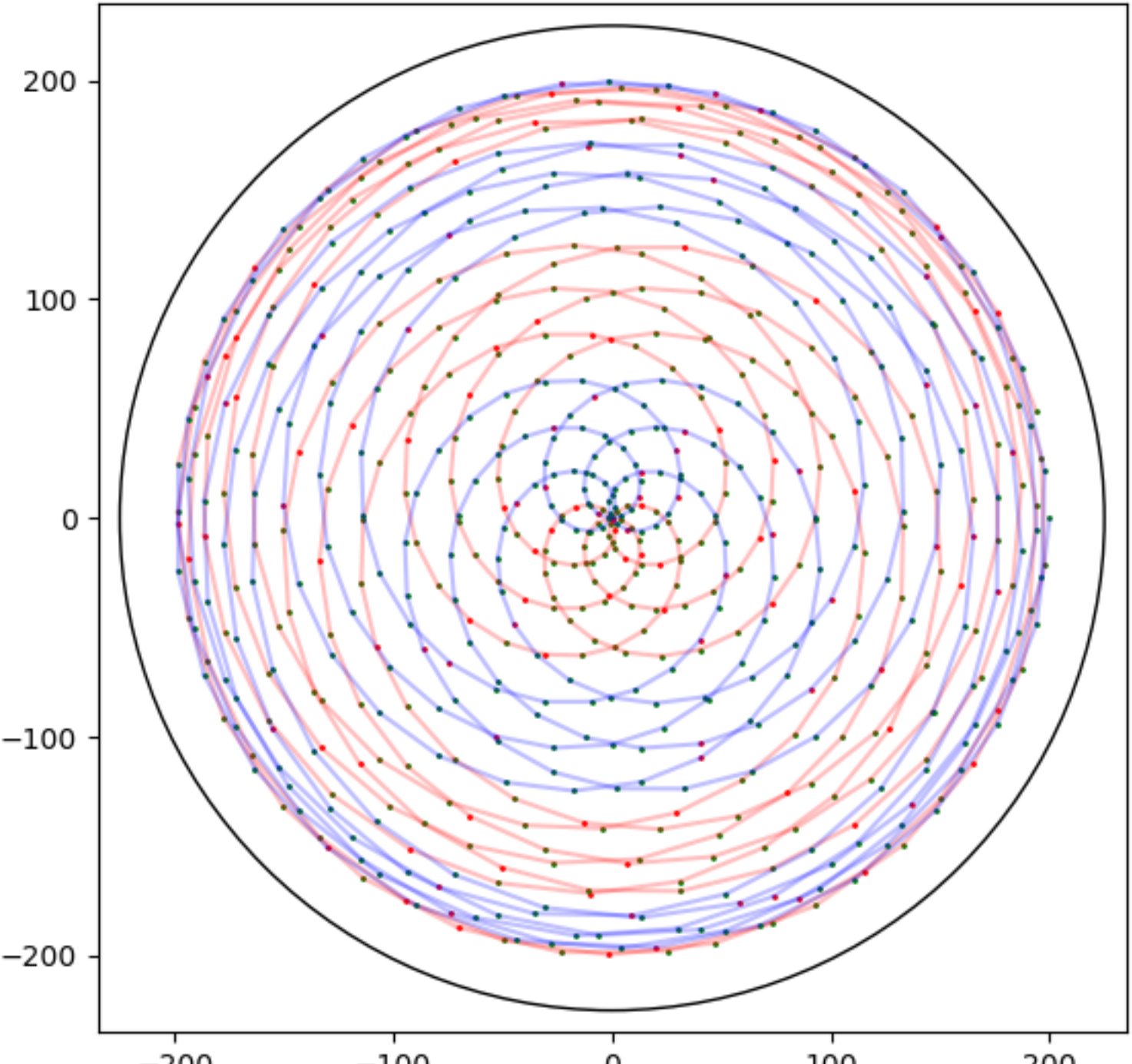

200
100
0
−100
−200
−200
−100
0
100
200


200
100
0
−100
−200
−200
−100
0
100
200


200
100
0
−100
−200
−200
−100
0
100
200


200
100
0
−100
−200
−200
−100
0
100
200


200
100
0
−100
−200
−200
−100
0
100
200


200
100
0
−100
−200
−200
−100
0
100
200

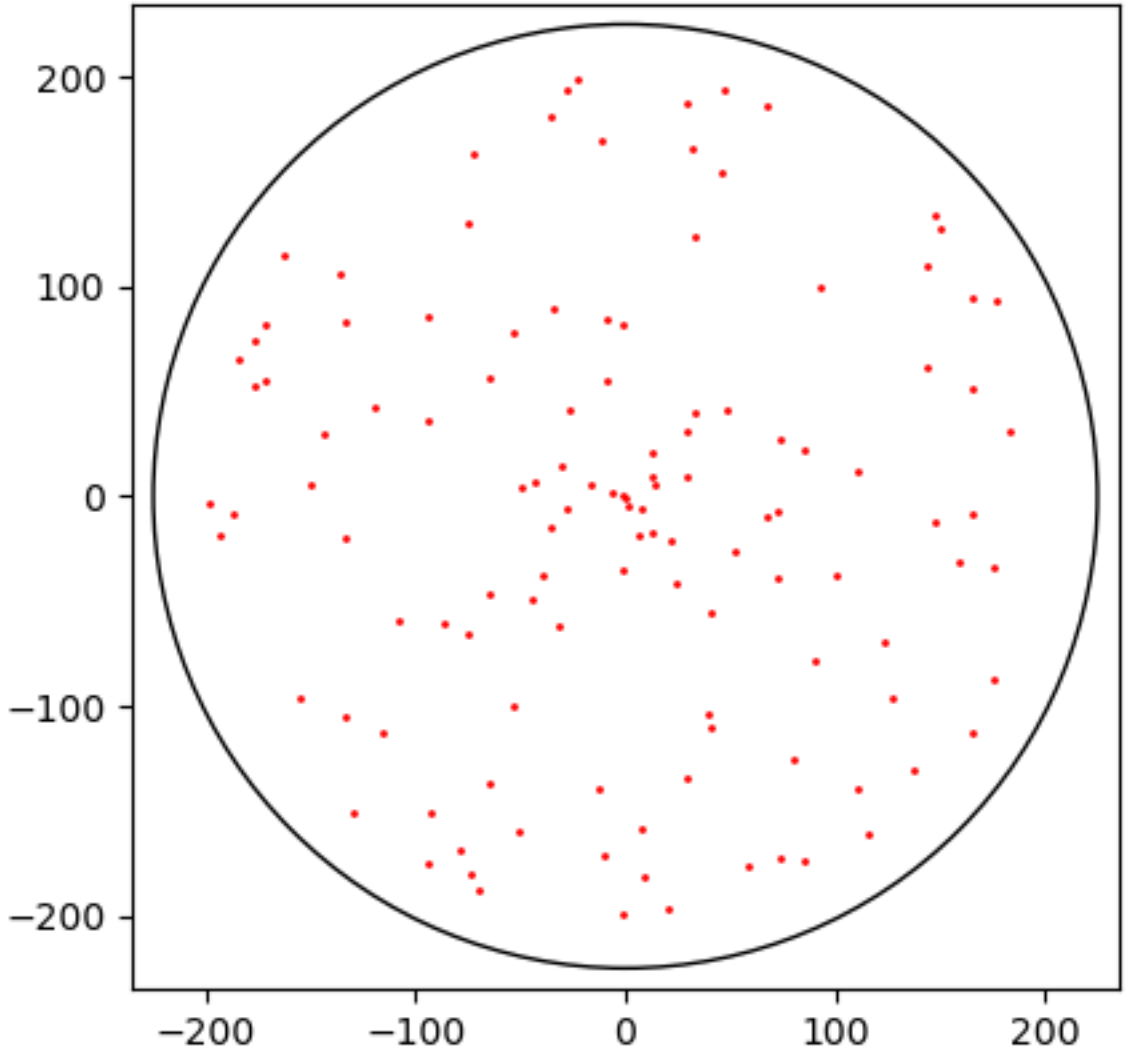
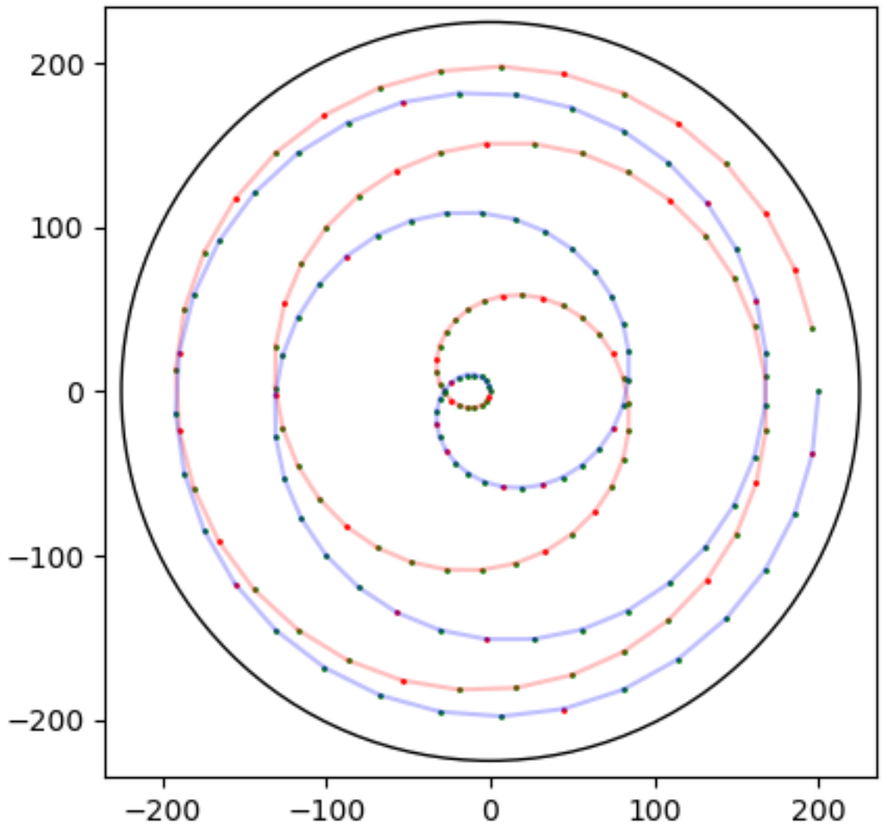

200
100
0
−100
−200
−200
−100
0
100
200

Figure 1



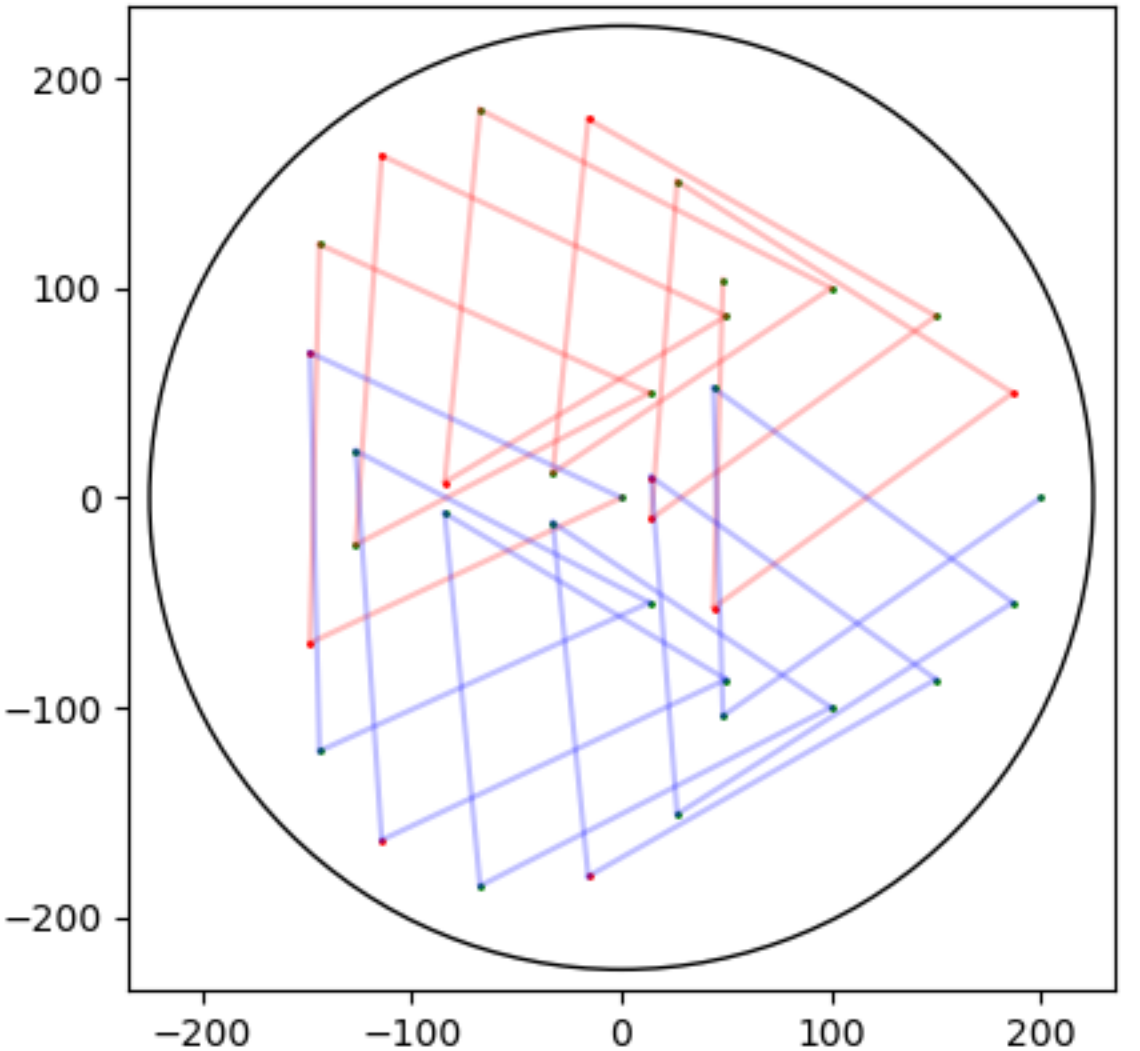
emacs25@vellachemla-X510UA

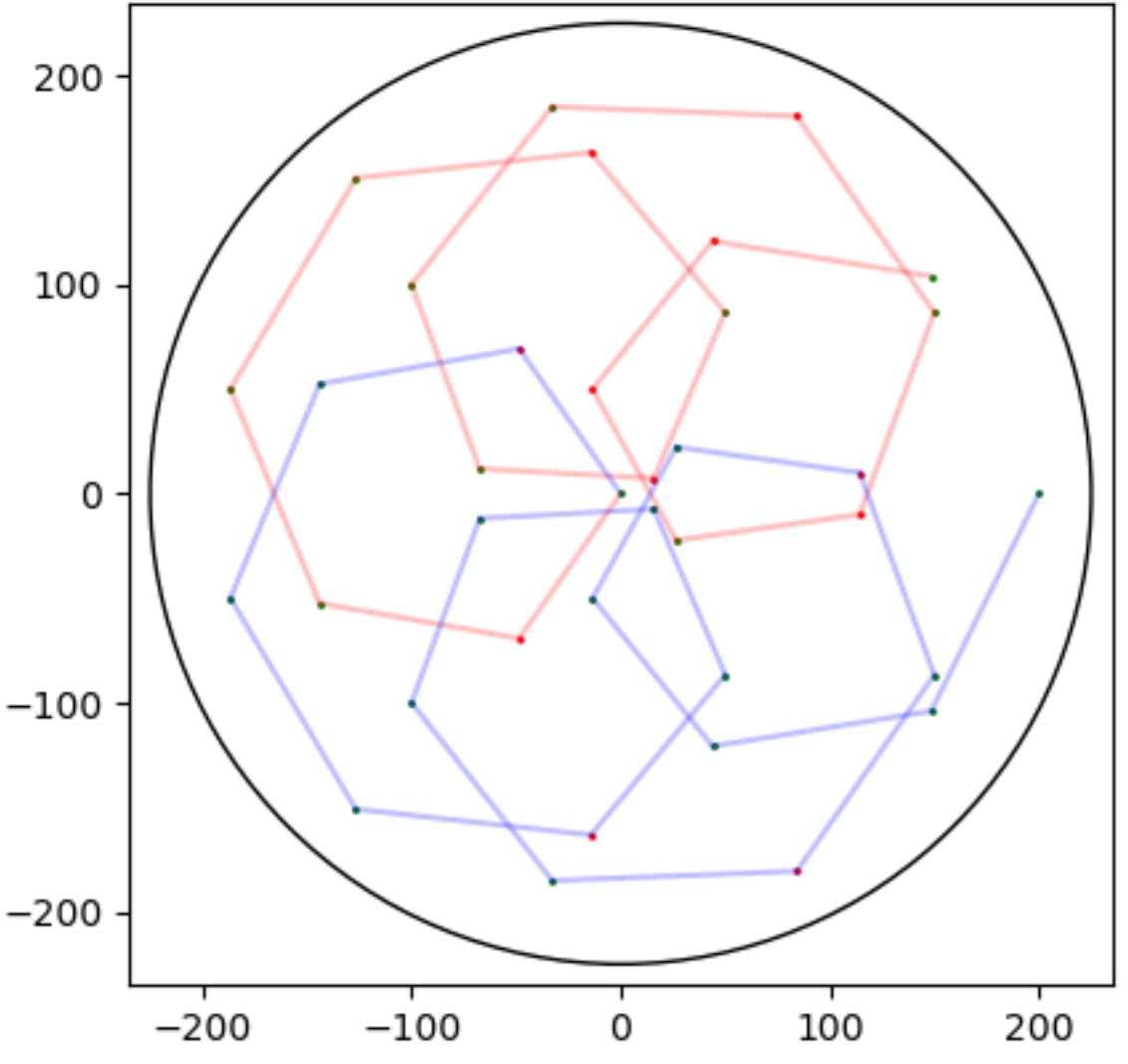
File Edit Options Buffers Tools Python Help

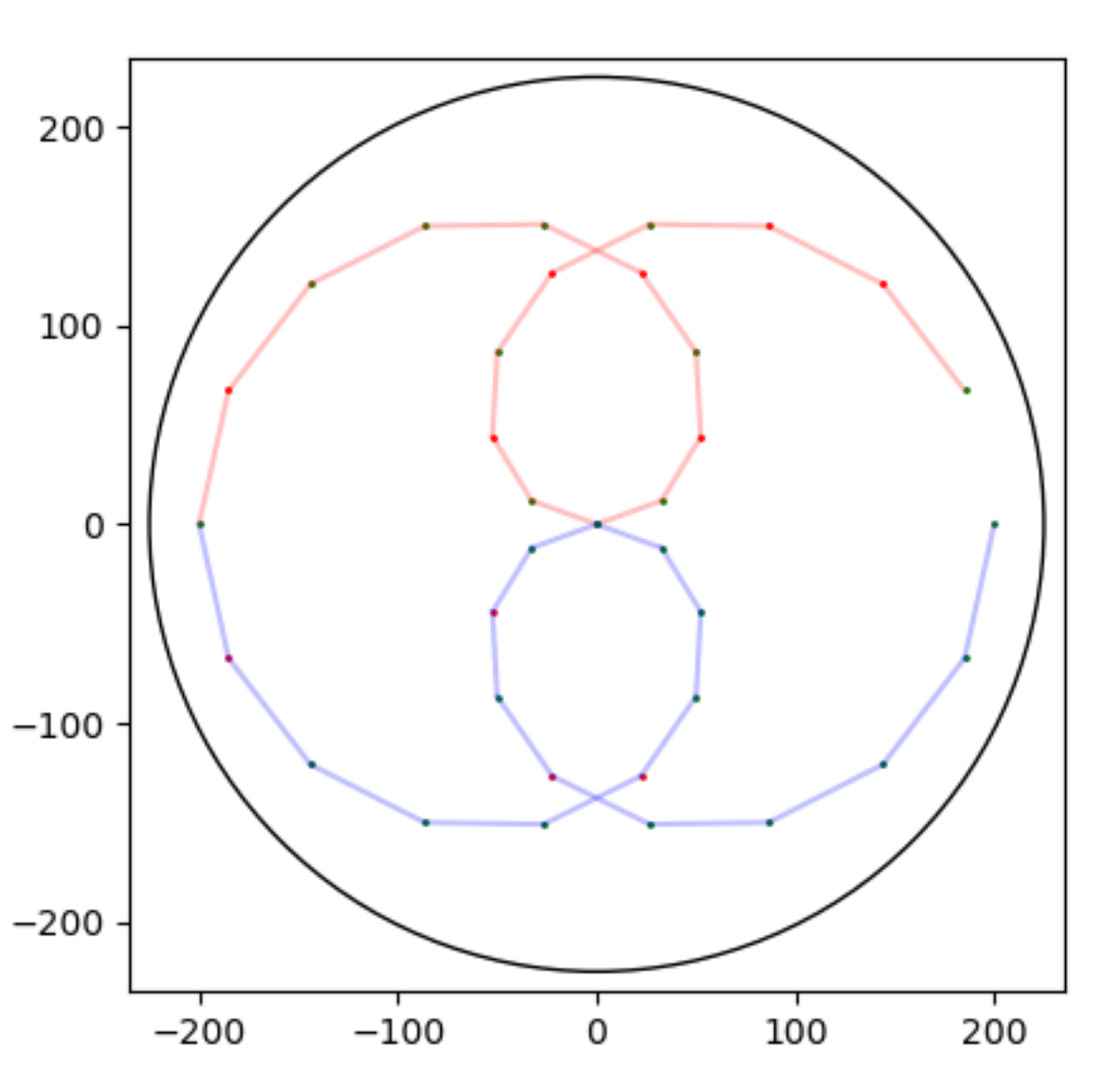
```
n = 30*36 ; x = 1 ; xmax = 100 ; fig = plt.figure() ; ax = fig.gca() ;
Ens = [30,36]
nbprime = len(Ens) + 0.25 ; print(str(nbprime))
ax.set_xlim((-1)*nbprime*xmax-10,nbprime*xmax+10) ;
ax.set_ylim((-1)*nbprime*xmax-10,nbprime*xmax+10)
cercle = plt.Circle((0,0), nbprime*xmax, fill=False)
ax.add_artist(cercle)

#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x <= 180:
  x0, y0 = 0, 0
  for element in Ens:
    t = 2*pi*(x % element)/element
    x0 = x0 + xmax*math.cos(t)
    y0 = y0 + xmax*math.sin(t)
  #print(str(x0)+'   '+str(y0))
  if (prime(x)):
    plt.plot(x0, y0, 'r', marker='o', markersize=1)
    #ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
  else:
    plt.plot(x0, y0, 'g', marker='o', markersize=1)
    #ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
  if (x != 1):
    if (x <= 90):
      ax.plot([xprec,x0],[yprec,y0],'red', alpha=0.25)
    else:
      ax.plot([xprec,x0],[yprec,y0],'blue', alpha=0.25)
  xprec = x0
  yprec = y0
  ax.set_aspect('equal')
  x = x+1
plt.show()
```

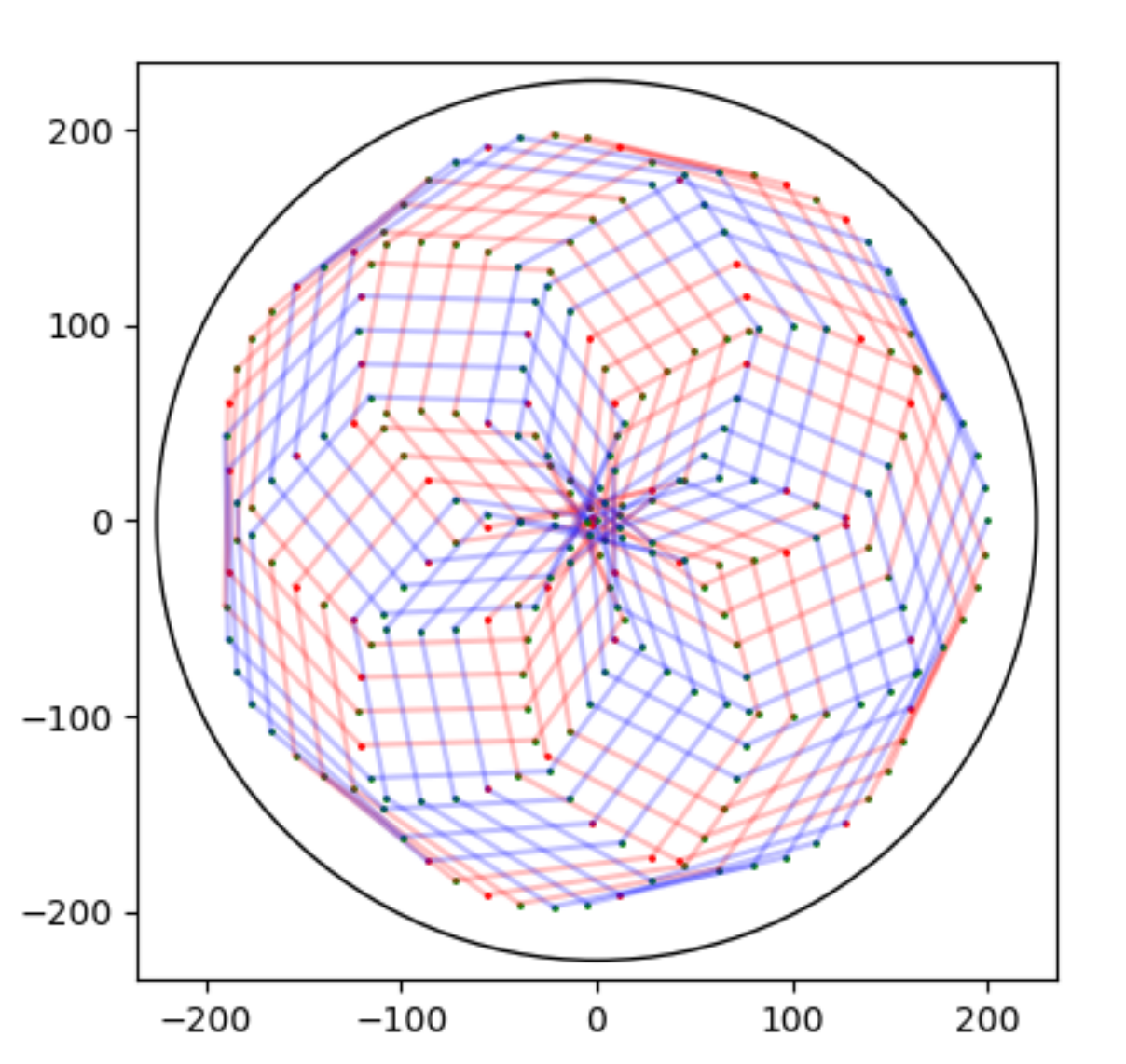
-:--- **test-curiosite-30-36.py** 28% L42 (Python)

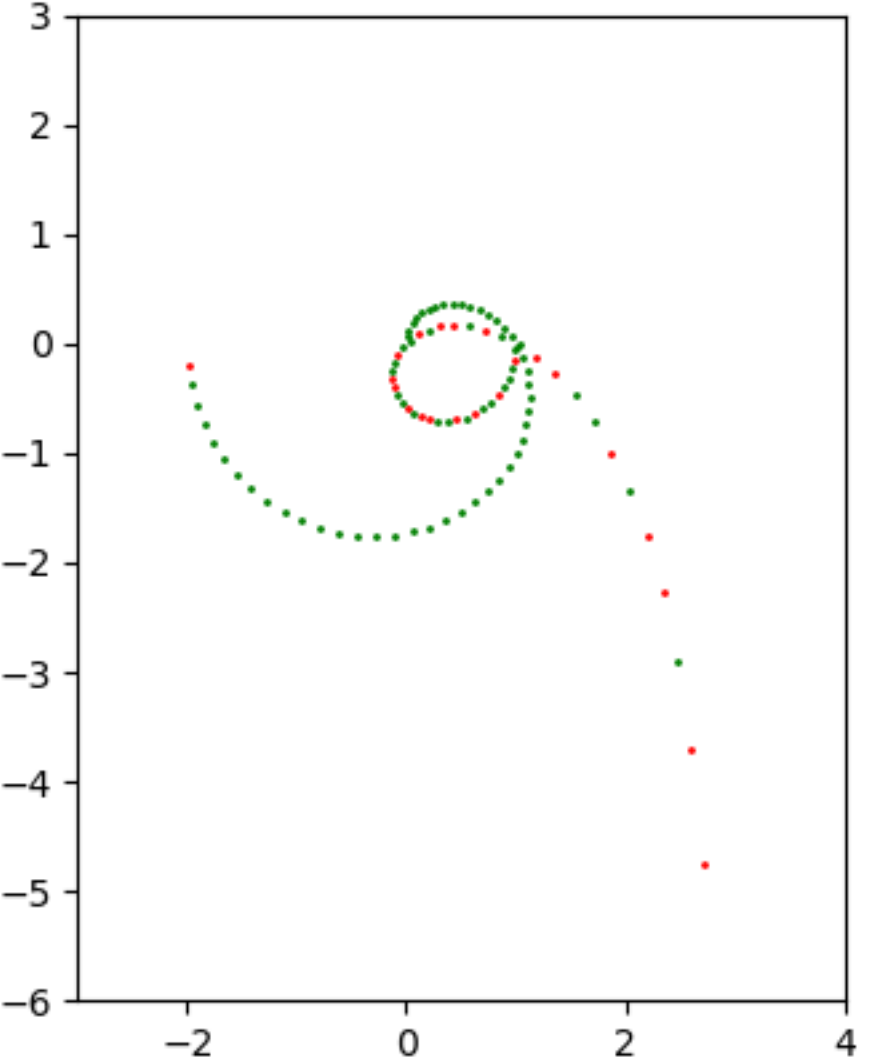
Wrote /home/vella-chemla/Desktop/test-curiosite-30-36.py

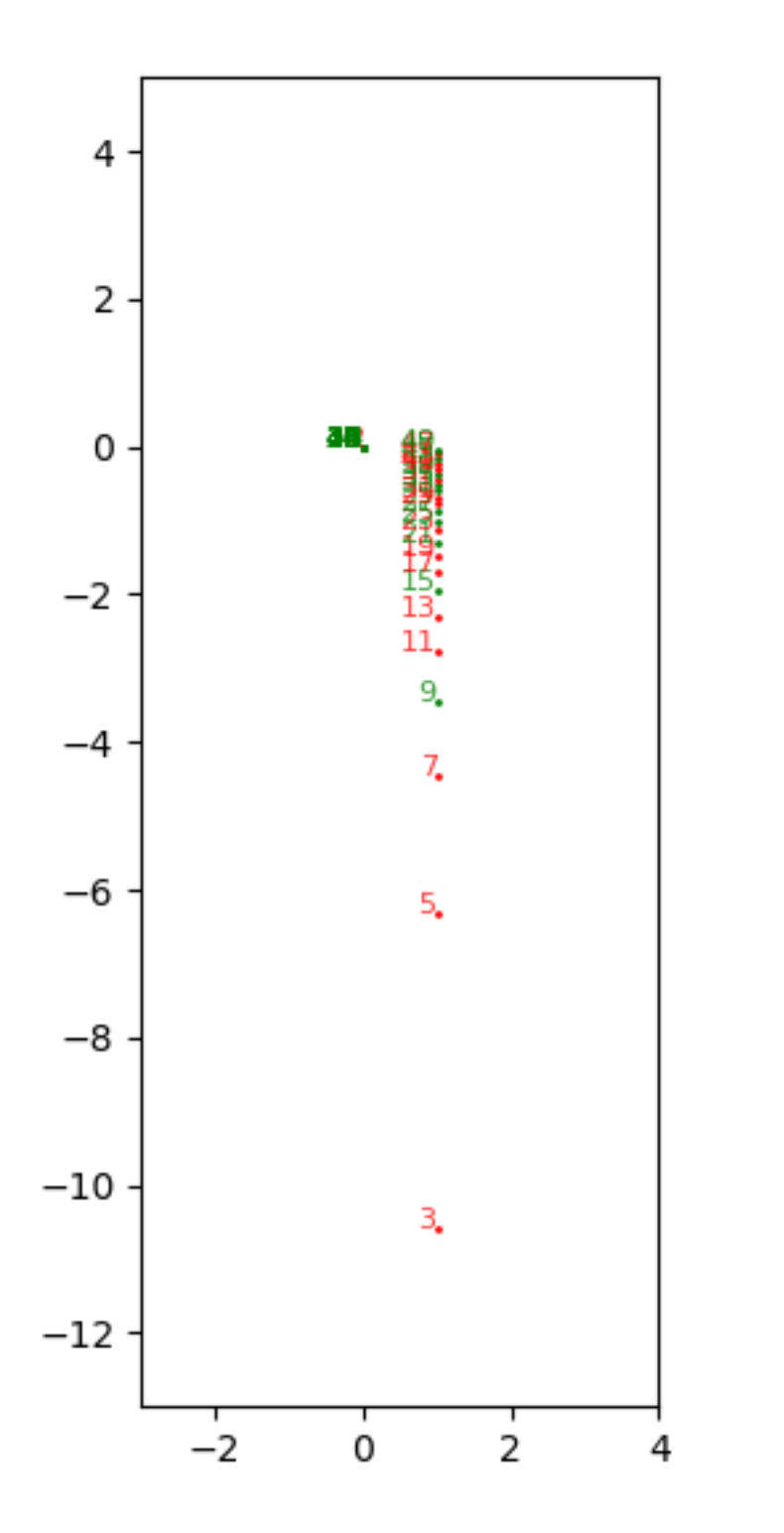

200
100
0
−100
−200
−200
−100
0
100
200


200
100
0
−100
−200
−200
−100
0
100
200


200
100
0
−100
−200
−200
−100
0
100
200


200
100
0
−100
−200
−200
−100
0
100
200


3
2
1
0
−1
−2
−3
−4
−5
−6
−2
0
2
4


4
2
0
−2
−4
−6
−8
−10
−12
−2
0
2
4

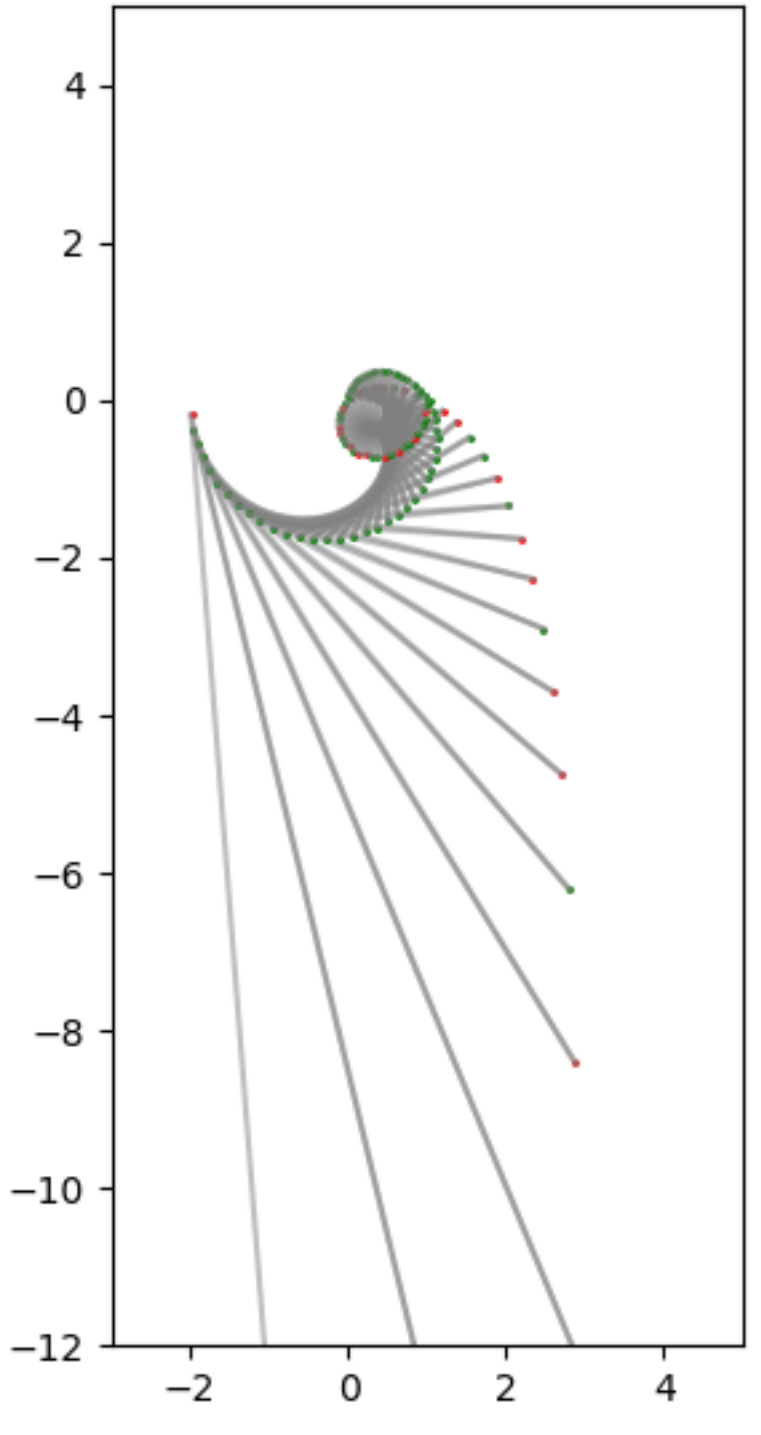
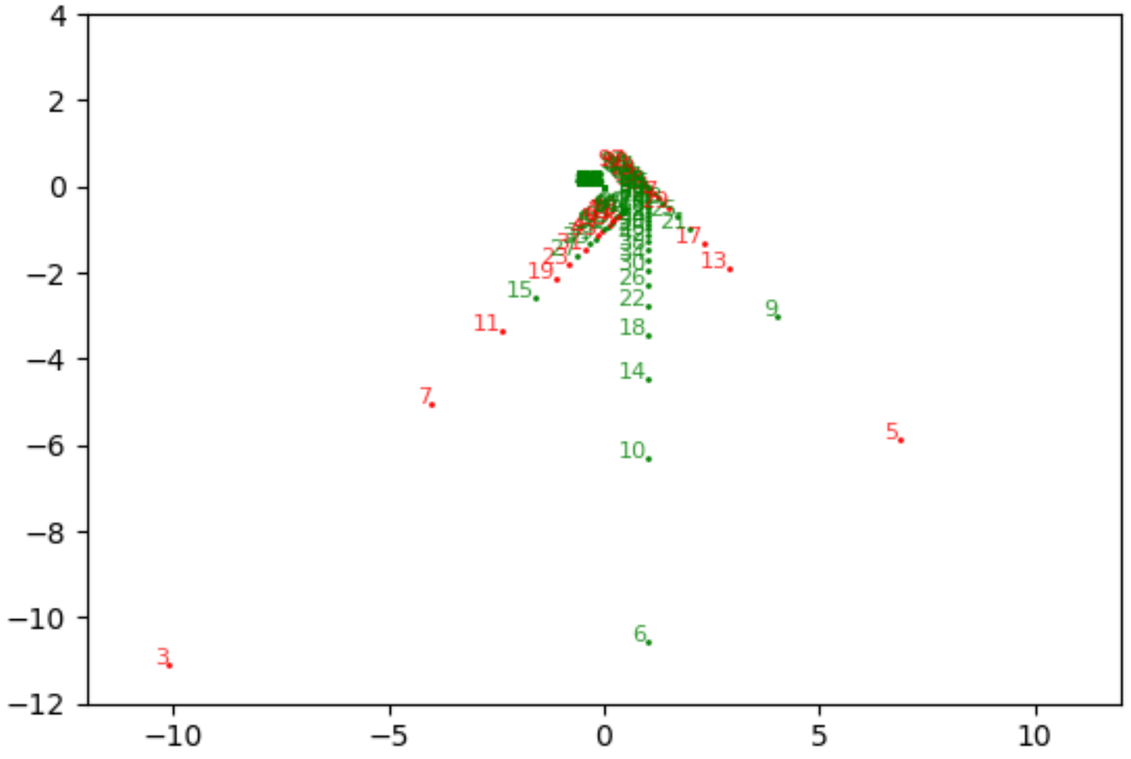

4
2
0
−2
−4
−6
−8
−10
−12
−2
0
2
4

Figure 1

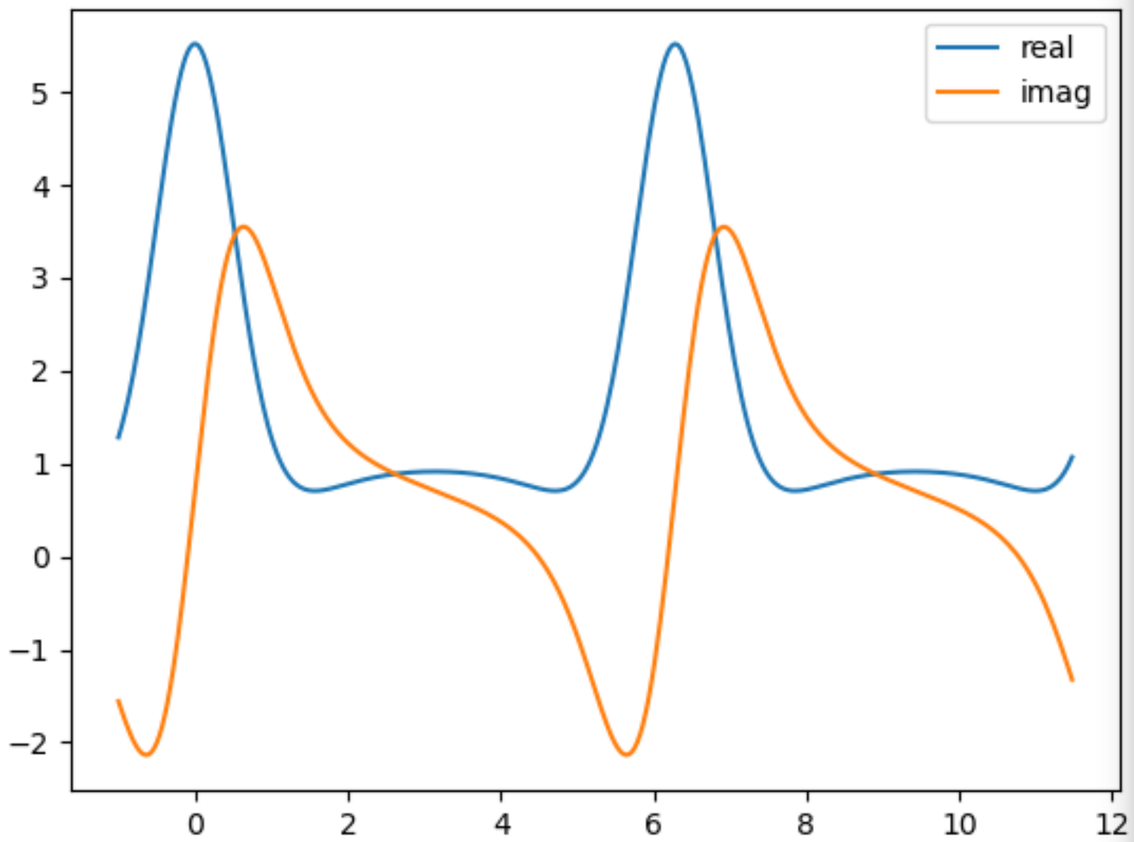


emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
  if (atester in [0,1]): return False ;
  if (atester in [2,3,5,7]): return True ;
  while (pastrouve):
    if ((k * k) > atester): return True
    else:
      if ((atester % k) == 0): return False
      else: k=k+1

fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')

n = 100
m = 50
for k in range(1,n):
    t = exp(-j*math.pi*k/n)
    z = (1.0 - t**m)/(1.0 - t)
    if True:
        c = 'r' if prime(k) else 'g'
        plt.plot(z.real, z.imag, color=c, marker='o', markersize=1)
        ax.text(z.real, z.imag, str(k), color=c, fontsize=8, ha='right', alpha=
.8)

xmin, xmax, ymin, ymax = ax.axis()
print(xmin, xmax, ymin, ymax)
ax.set_xlim(-12, 12) ;
ax.set_ylim(-12, 4)
plt.show()
```

Figure 1



emacs25@vellachemla-X510UA

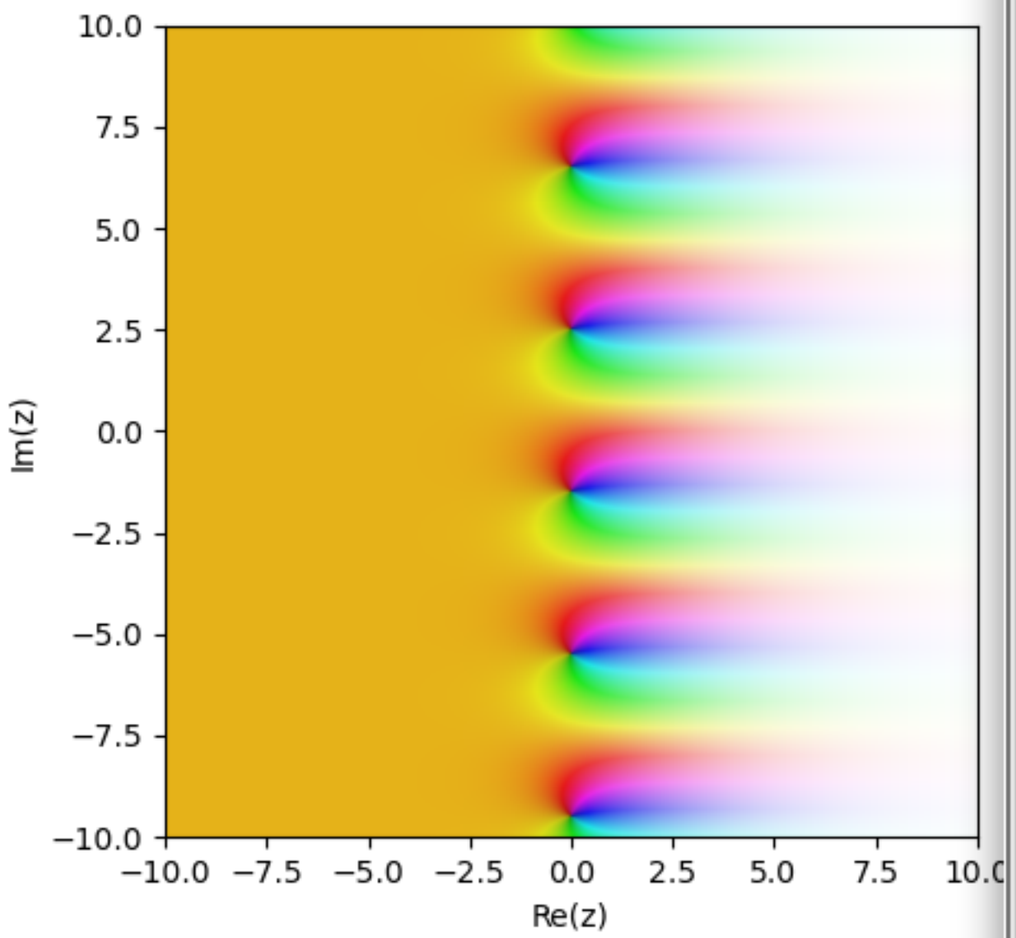
File Edit Options Buffers Tools Python Help

```
import numpy
import matplotlib.pyplot as plt
import mpmath
import cmath

def fctfr(z):
    return (mpmath.exp(mpmath.pi*z/2)+mpmath.exp(mpmath.pi*mpmath.j/4)).real

def fctfi(z):
    return (mpmath.exp(mpmath.pi*z/2)+mpmath.exp(mpmath.pi*mpmath.j/4)).imag

#mp.cplot(lambda z:exp(pi*z/2)+exp(pi*j/4),[-10,10],[-10,10], points =100000)
abs=[] ; y = [] ; z = [] ; x = -1
for t in range(500):
    x = x+t*0.0001
    print(x)
    abs.append(x)
    y.append(fctfr(cmath.exp(1j*x)))
    z.append(fctfi(cmath.exp(1j*x)))
print(y)
print(z)
plt.plot(abs, y, label="real")
plt.plot(abs, z, label="imag")
plt.legend()

plt.show()
```

-:--- **fonction-f.py** All L10 (Python)

Wrote /home/vella-chemla/Desktop/revenir-aux-operateurs/fonction-f.py

Figure 1



8.6039 y=-4.52381 [0.997, 0.97, 0.991]

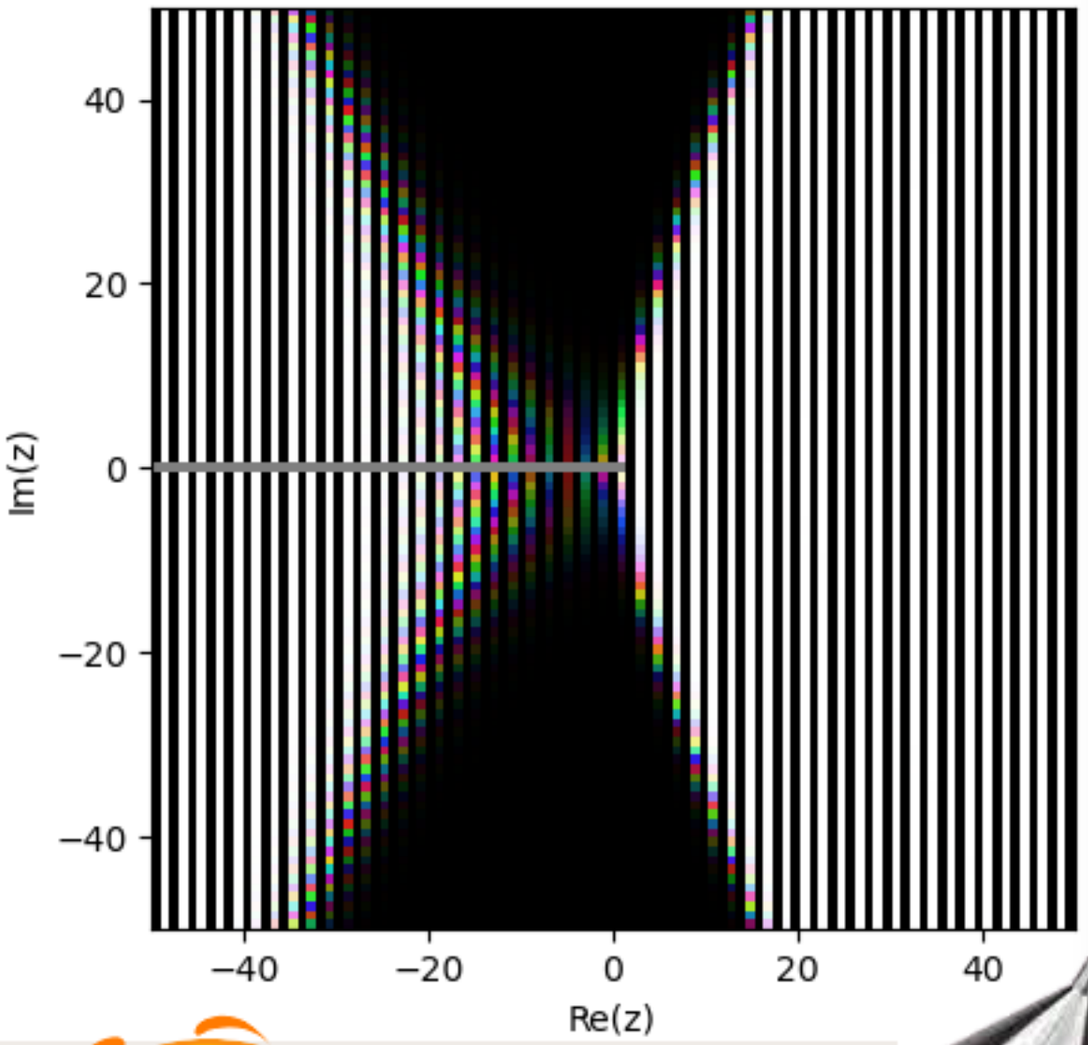
emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
import mpmath
from mpmath import *

mpmath.cplot(lambda z:exp(pi*z/2)+exp(pi*j/4),[-10,10],[-10,10], points =1000000
)
```

-:--- **fonction-f.py** All L5 (Python)

Figure 1



emacs25@vellachemla-X510UA

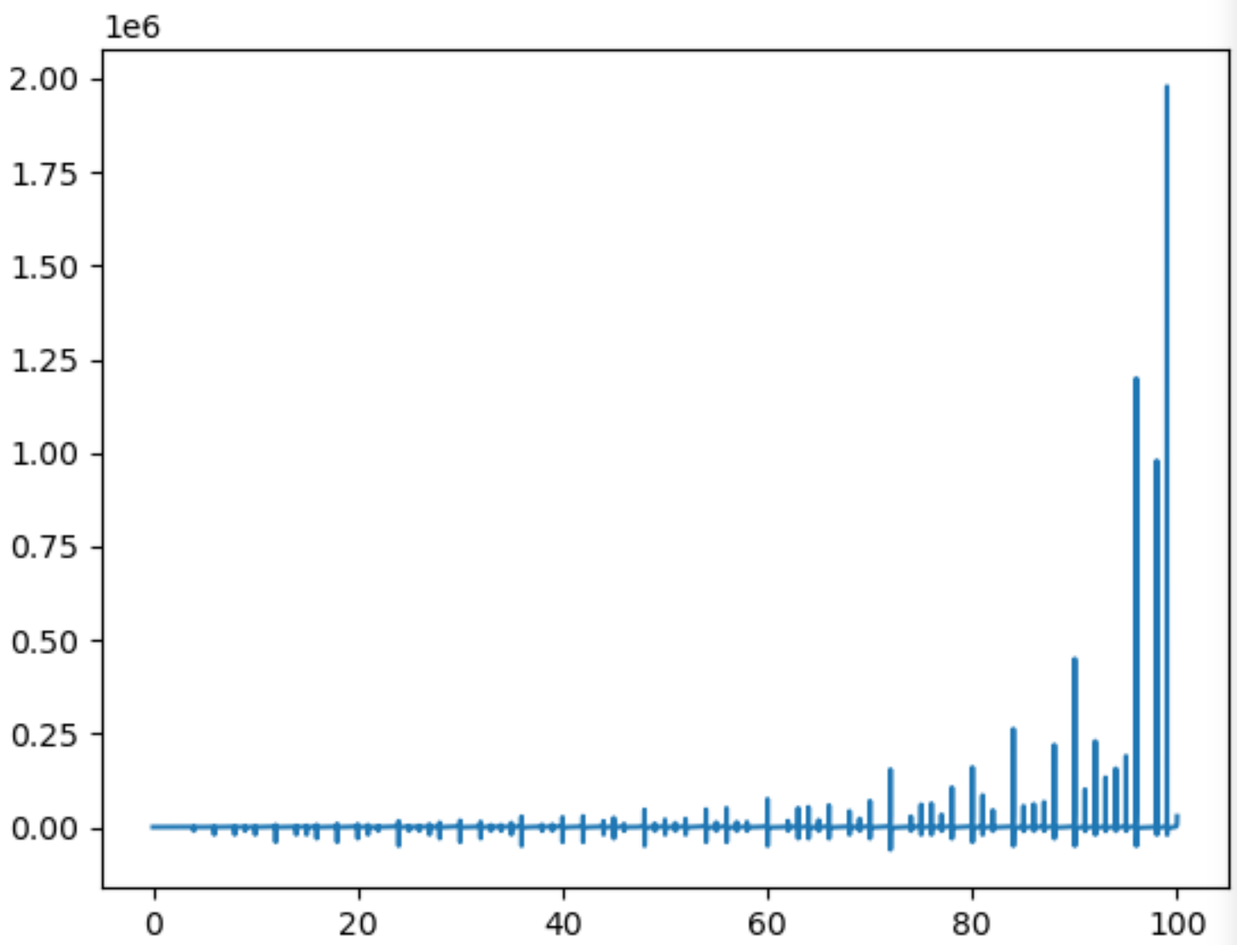
File Edit Options Buffers Tools Python Help

```
import mpmath
from mpmath import exp, pi, zeta, j, sin, gamma
from scipy import integrate
import numpy
from numpy import *

def sommeinfinie(s):
    somme = 0.0
    n = 1
    res = (n**(s-1))*((-j)**(s-1)+(j**(s-1)))
    while (n <= 1000):
        somme = somme+res
        n=n+1
        res = (n**(s-1))*((-j)**(s-1)+(j**(s-1)))
    return somme

def zetaautredef(s):
    bout1 = (2*pi)**s
    bout2 = 2*mpmath.sin(pi*s)*mpmath.gamma(s-1)
    return bout1*sommeinfinie(s.real)/bout2

mpmath.cplot(zetaautredef,[-50,50],[-50,50], points = 10000)
```

-:--- **testzetadef3.py** All L1 (Python)

5 1

Figure 1



emacs25@vellachemla-X510UA

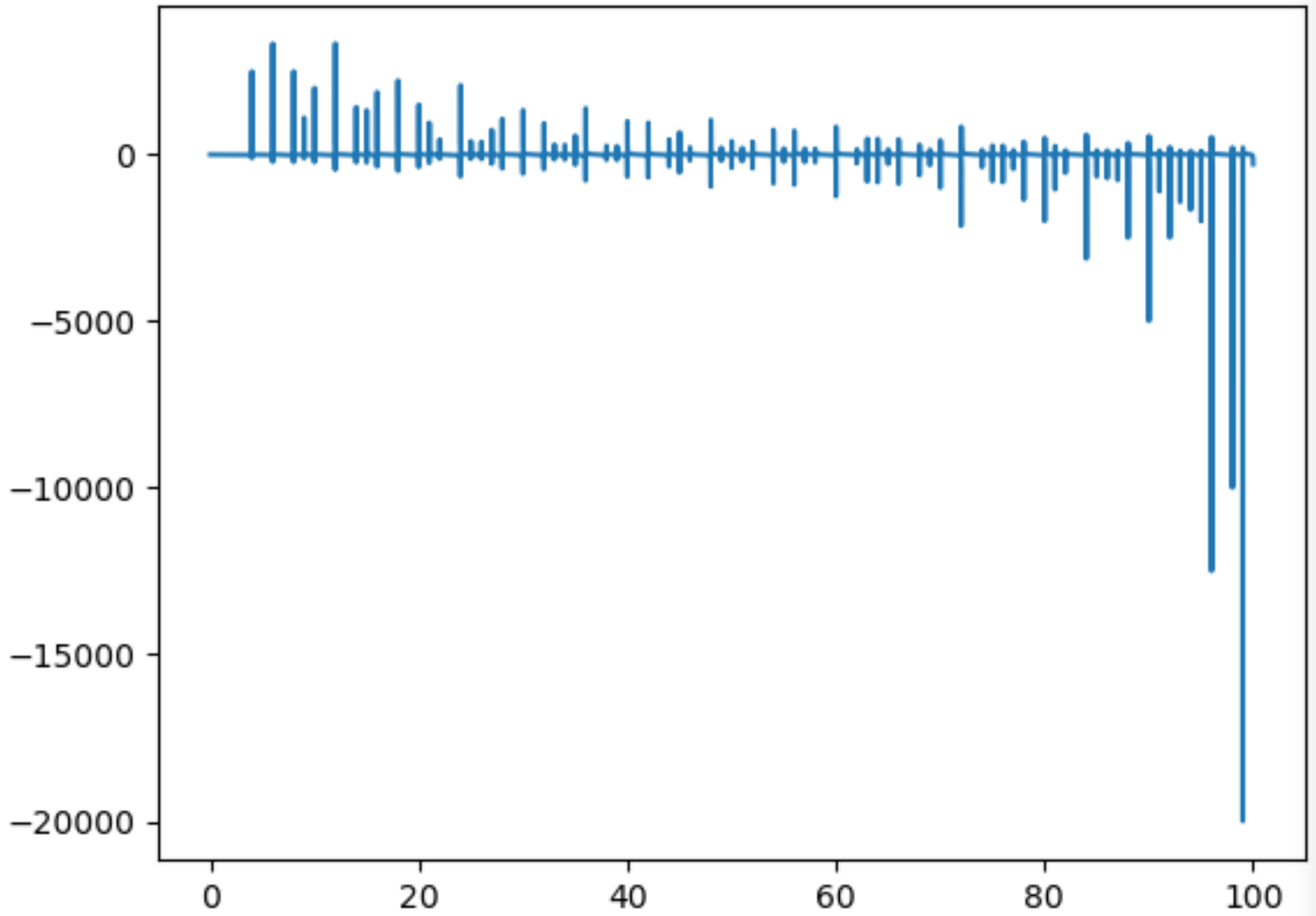
File Edit Options Buffers Tools Python Help

```python
import matplotlib.pyplot as plt
import numpy as np

n = 100
x = np.linspace(0,n,n*n)
rac = int(np.sqrt(n))
y1 = 0.0
for y in range(2, rac):
    multiple = 2*y
    while multiple <= n:
        y1=y1+1.0/(1.0-x/float(multiple))
        multiple = multiple+y
plt.plot(x, y1)
plt.show()
```

-:--- **mesdiracs3.py**   All L4   (Python)

Wrote /home/vella-chemla/Desktop/voir-avec-jacques/mesdiracs3.py

Figure 1



emacs25@vellachemla-

File Edit Options Buffers Tools Python Help

```
import matplotlib.pyplot as plt
import numpy as np

n = 100
x = np.linspace(0,n,n*n)
rac = int(np.sqrt(n))
y1 = 0.0
for y in range(2,rac):
    multiple = 2*y
    while multiple <= n:
        y1 = y1+1.0/(x-float(multiple))
        multiple = multiple+y
plt.plot(x, y1)
plt.show()
```

-:--- **mesdiracs.py** All L15 (Python)

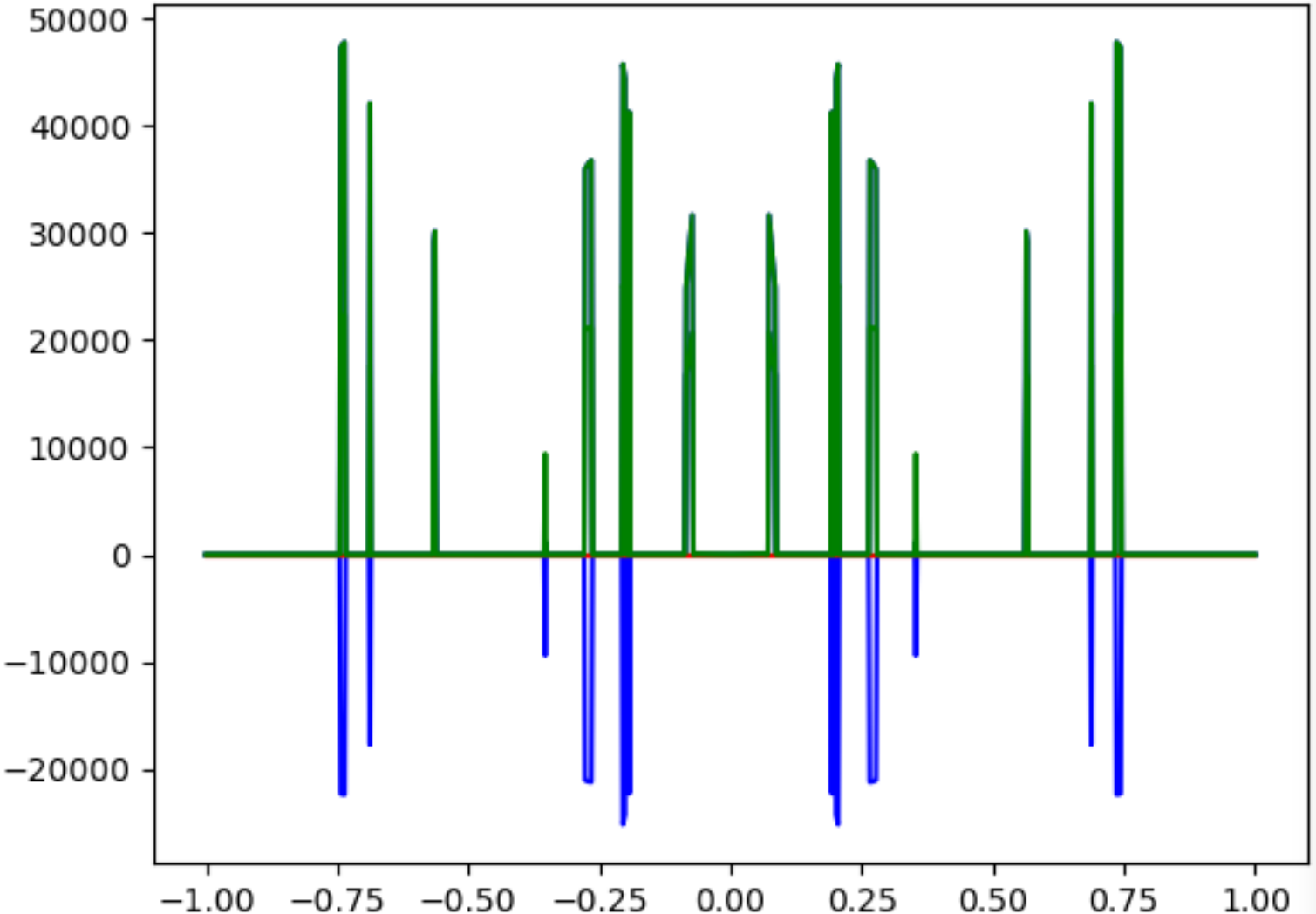
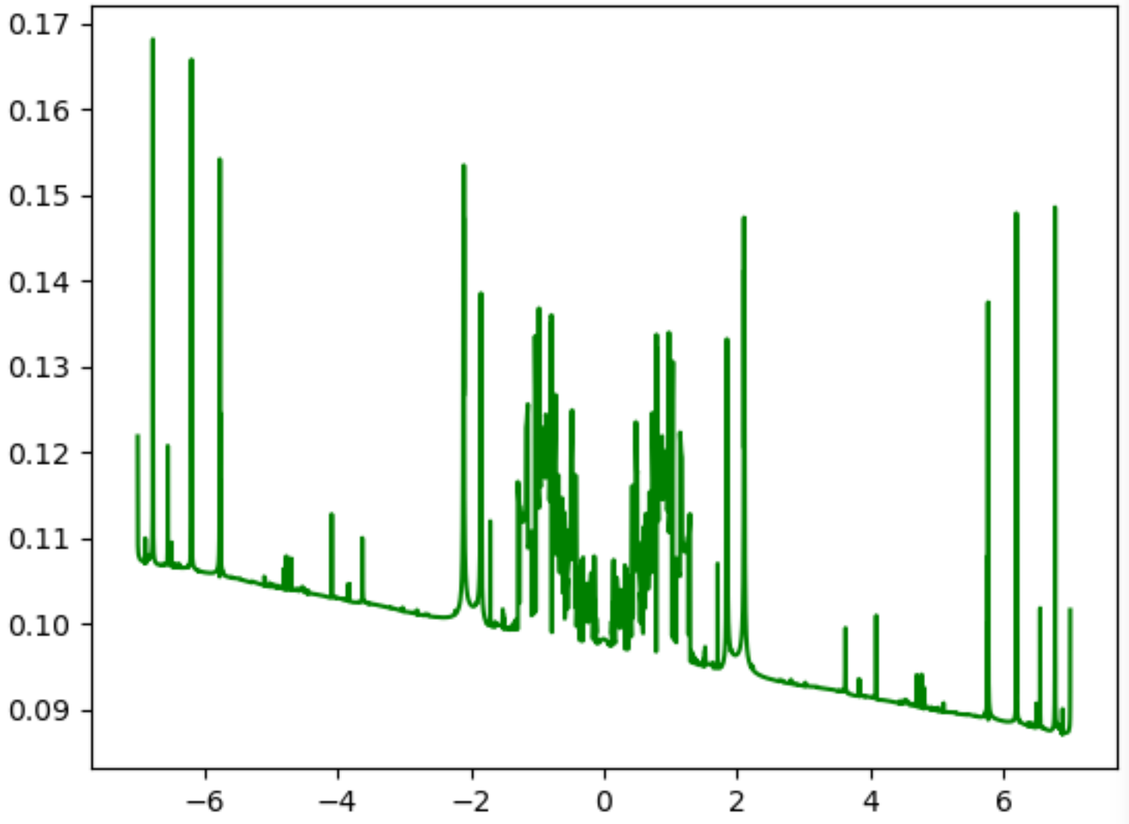

50000
40000
30000
20000
10000
0
−10000
−20000
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

Figure 1



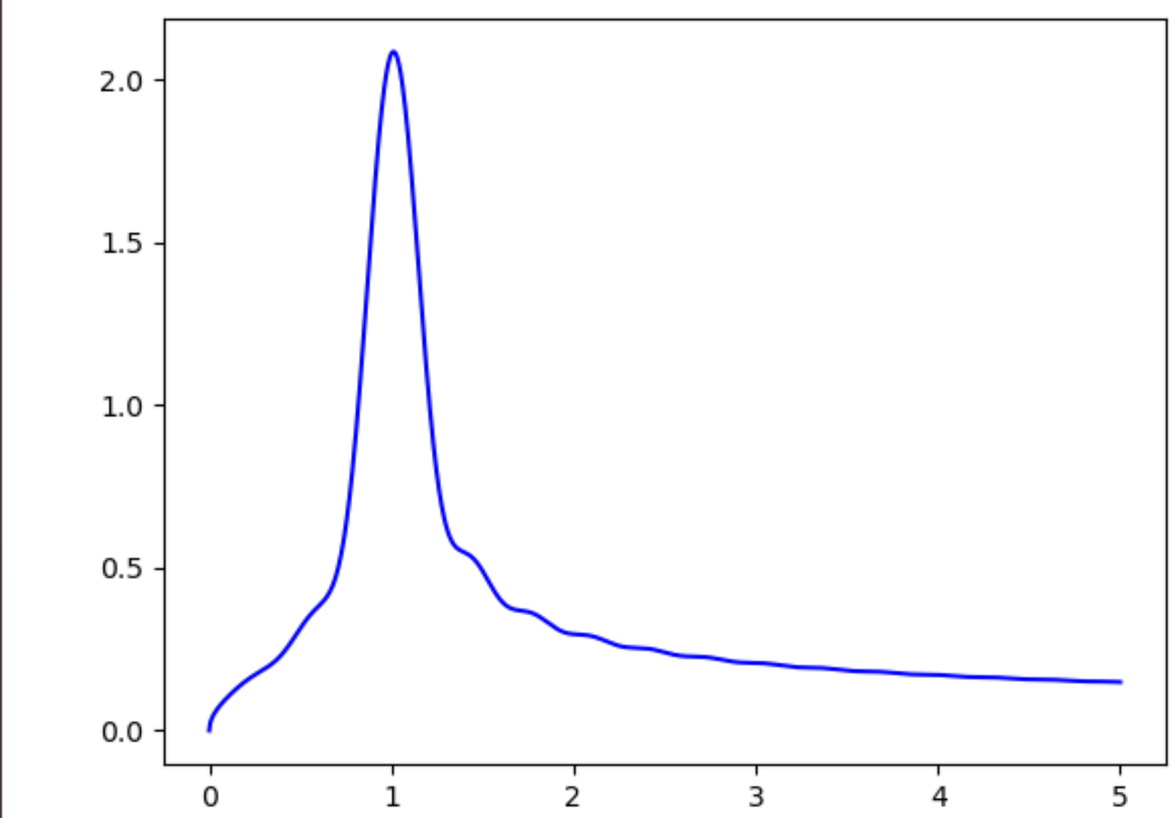
emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
    return quad(lambda rho: delta(rho) * 2 * np.cos(t * np.log(rho))/rho, 1, np.inf)

xb = np.arange(-7,7,0.00001)
yb = fcttheta(xb)
ybprime = 2*scipy.misc.derivative(fcttheta, xb, n=1, dx=0.01)
print(plt.ylim())
print("ybprime")
print(ybprime)

xh = np.linspace(-7, 7, 14000)
yh = [delta_hat(r) for r in xh]
print("yh")
print(yh)
print("real(yh)")
print(np.real(yh))

tabres1 = []
print("tableau yh")
for k1 in range(14000):
    print(np.real(yh)[k1][1])
    tabres1.append(np.real(yh)[k1][1])
tabres2 = []
for k2 in ybprime:
    print("ausuivant")
    print(k2)
    tabres2.append(k2)

tabres = []
for k in range(14000):
    tabres.append(tabres1[k]+tabres2[k])
plt.plot(xh, tabres, 'g')
```

-:--- **zigouigoui.py** 42% L34 (Python)

Figure 1




```
(base) vella-chemla@vellachemla-X510UA:~/Desktop/ne-te-decourage-pa
(base) vella-chemla@vellachemla-X510UA:~/Desktop/ne-te-decourage-pa
(base) vella-chemla@vellachemla-X510UA:~/Desktop/ne-te-decourage-pa
(base) vella-chemla@vellachemla-X510UA:~/Desktop/ne-te-decourage-pa
(base) vella-chemla@vellachemla-X510UA:~/Desktop/ne-te-decourage-pa
ssai2pourvoirlim.py
limites pour la partie reelle
(-0.10428921163010954, 2.1900734442323)
```

emacs25@vellachemla-X510UA

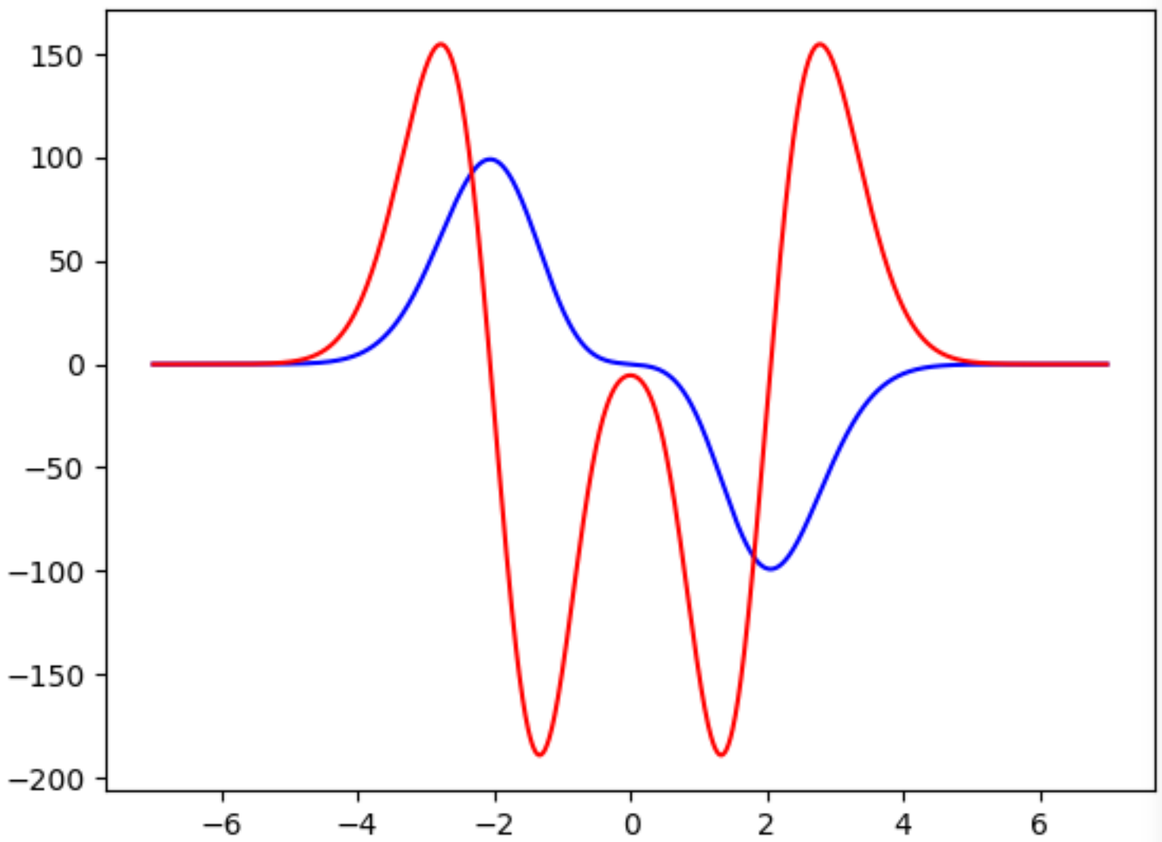
File Edit Options Buffers Tools Python Help

```
import numpy as np
from scipy.integrate import quad
import matplotlib.pyplot as plt

def SI(x):
    return quad(lambda t: np.sinc(t), 0, x)

def cmplx(t):
    return np.complex(t[0],t[1])

def delta(rho):
    r1 = 2.0 * np.pi * (rho + 1.0)
    r2 = 2.0 * np.pi * (rho - 1.0)
    z1 = cmplx(SI(r1))/r1
    z2 = cmplx(SI(r2))/r2
    return 2.0 * np.sqrt(rho) * (z1 + z2)

x = np.linspace(0, 5, 1000)
y = [delta(r) for r in x]

plt.plot(x, np.real(y), 'b')
print("limites pour la partie reelle")
print(plt.ylim())
plt.show()
```

-:--- **reessai2pourvoirlim.py** All L18 (Python)

Wrote /home/vella-chemla/Desktop/ne-te-decourage-pas/reessai2pourvoirlim.py

Figure 1



emacs25@vellachemla-X510UA

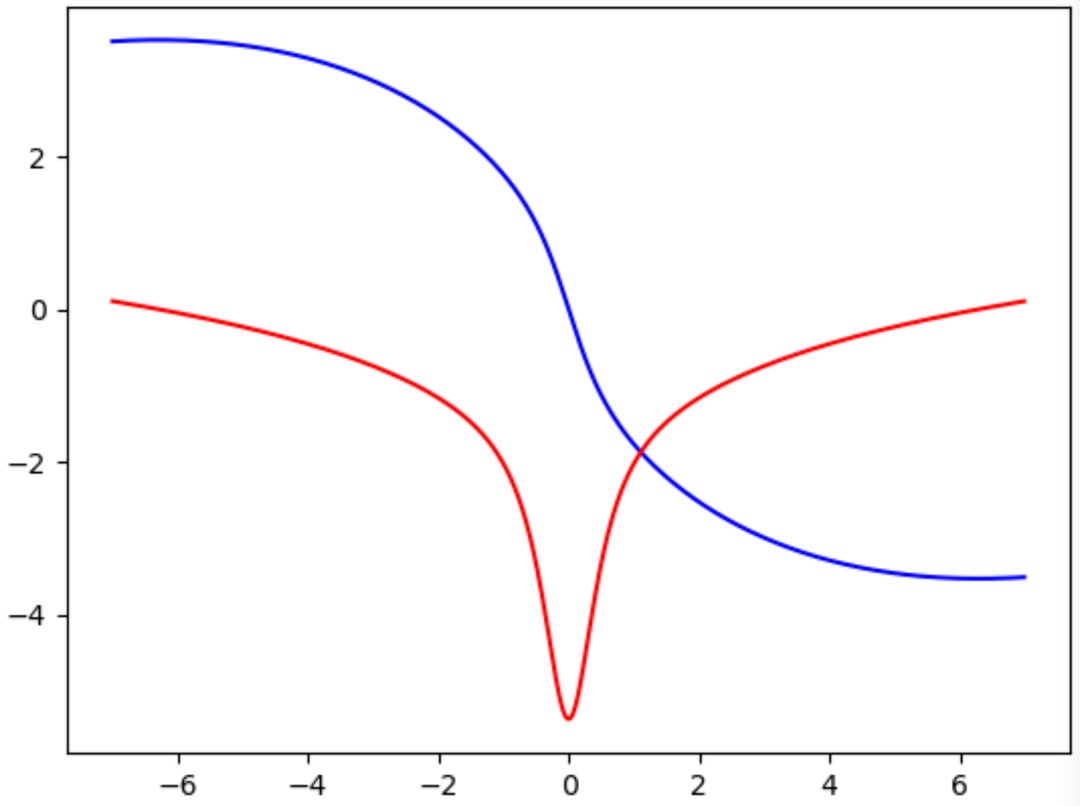
File Edit Options Buffers Tools Python Help

```
import matplotlib.pyplot as plt
import numpy as np
import scipy
import scipy.special as sp
import scipy.misc

def fchapeau(t):
    return ((1+4*t*t)**2)*np.exp(-t*t/2)

# un peu moche, pour pas multiplier des fcts, fchapeau caché dans fcttheta

def fcttheta(t):
    return (np.angle(sp.gamma(1/4+1j*t/2))-(np.log(np.pi)/2)*t)*fchapeau(t)

x = np.arange(-7,7,0.01)
y = fcttheta(x)
yprime = 2*scipy.misc.derivative(fcttheta, x, n=1, dx=0.01)
plt.plot(x,y,'b')
plt.plot(x, yprime, 'r')
pluspetiteordonnee, plusgrandeordonnee = plt.ylim()
print(pluspetiteordonnee)
plt.show()
```

U:--- **invchameau.py<2>** All L11 (Python)

Wrote /home/vella-chemla/Desktop/ne-te-decourage-pas/invchameau.py

Figure 1



emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
import matplotlib.pyplot as plt
import numpy as np
import scipy
import scipy.special as sp
import scipy.misc

def fcttheta(t):
    return np.angle(sp.gamma(1/4+1j*t/2))-(np.log(np.pi)/2)*t

x = np.arange(-7,7,0.01)
y = fcttheta(x)
yprime = 2*scipy.misc.derivative(fcttheta, x, n=1, dx=0.01)
plt.plot(x,y,'b')
plt.plot(x, yprime, 'r')
print(plt.ylim())
plt.show()
```

-:--- **deriveeriemannsiegel.py** All L17 (Python)

Fichier Édition Affichage Rechercher Terminal Aide

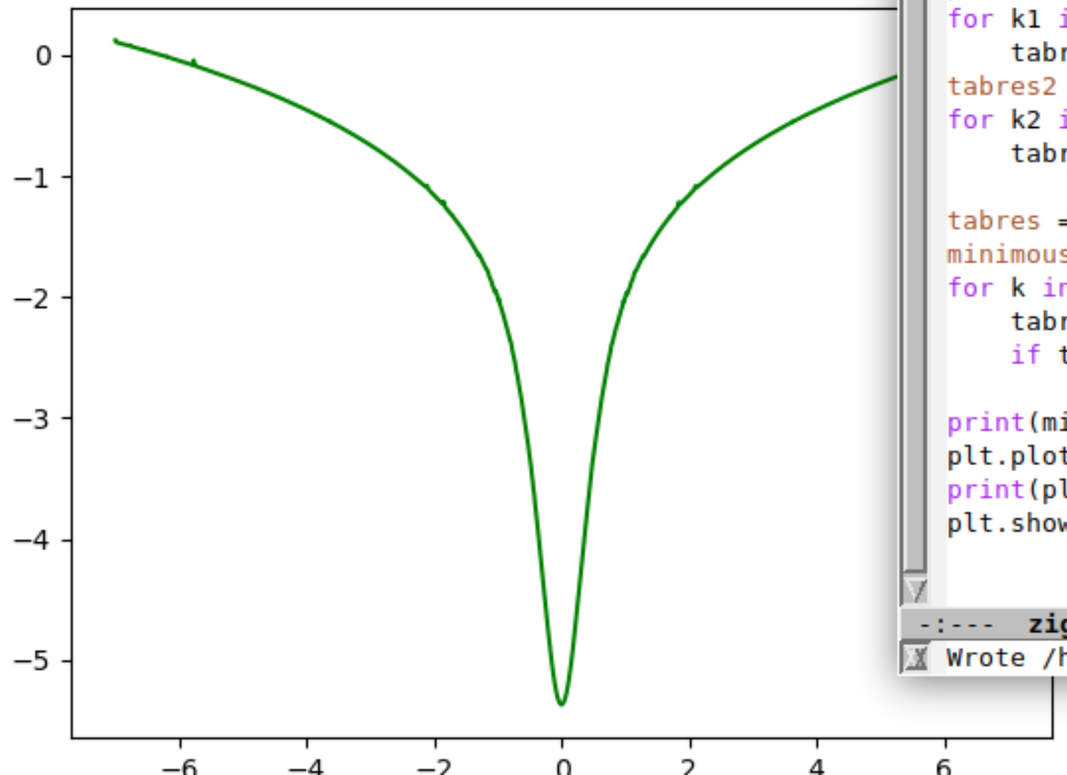
```
(base) vella-chemla@vellachemla-X510UA:~$ cd Desktop/ne-te-decoura
(base) vella-chemla@vellachemla-X510UA:~/Desktop/ne-te-decourage-p
iveeriemannsiegel.py
(-5.816775557129396, 3.976103703424913)
```

Terminal

Fichier Édition Affichage Rechercher Terminal Aide

```
ybprime
[0.10717933 0.10574728 0.10431318 ... 0.10287701 0.10431318 0.10574728]
/home/vella-chemla/anaconda3/lib/python3.8/site-packages/scipy/integrate/quadpac
k.py:465: ComplexWarning: Casting complex values to real discards the imaginary
part
  return _quadpack._qagie(func,bound,infbounds,args,full_output,epsabs,epsrel,li
mit)
zigouigoui.py:23: IntegrationWarning: The integral is probably divergent, or slo
wly convergent.
  return quad(lambda rho: delta(rho) * 2 *
f)
zigouigoui.py:23: IntegrationWarning: The
s been achieved.
  If increasing the limit yields no improv
  the integrand in order to determine the
  local difficulty can be determined (sing
  probably gain from splitting up the inte
  on the subranges.  Perhaps a special-pur
  return quad(lambda rho: delta(rho) * 2 *
f)
tableau yh
-5.370485098189474
(-5.645104269826774, 0.3965175061938411)
```

Figure 1



emacs25@vellachemla-X510

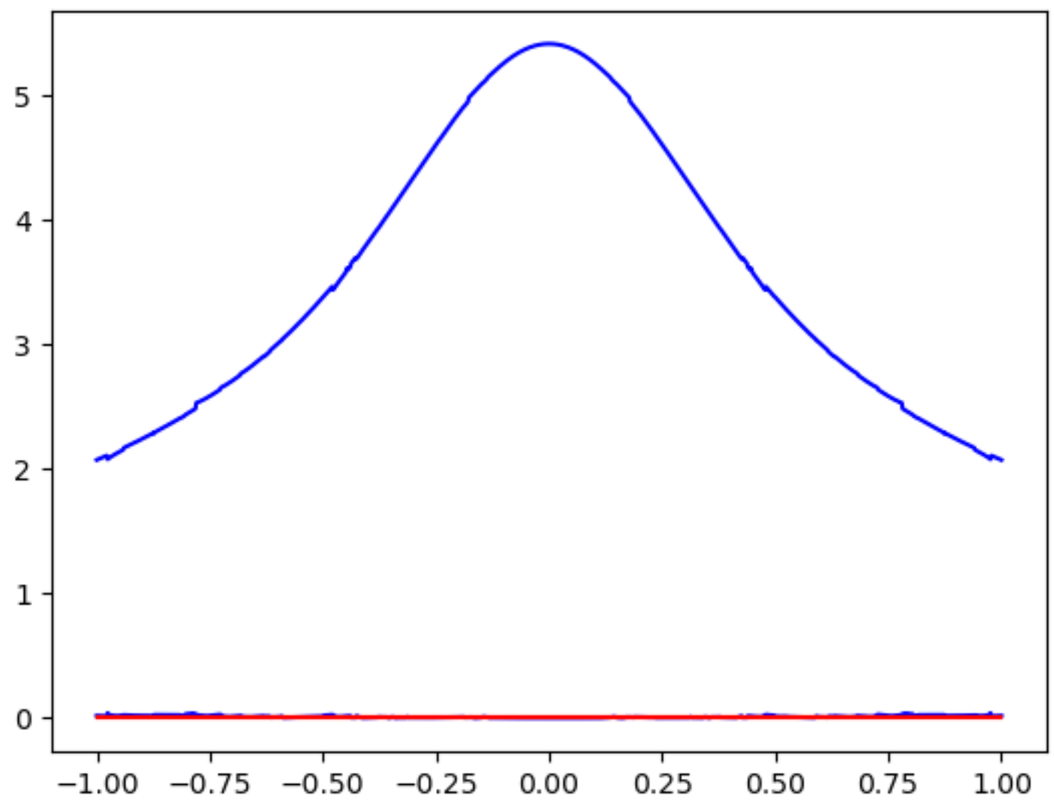
File Edit Options Buffers Tools Python Help

```
    return quad(lambda rho: delta(rho) * 2 * np.cos
inf)

xb = np.arange(-7,7,0.01)
yb = fctthetariemannsiegel(xb)
ybprime = 2*scipy.misc.derivative(fctthetariemannsi
print(plt.ylim())
print("ybprime")
print(ybprime)

xh = np.linspace(-7, 7, 1400)
yh = [delta_hat(r) for r in xh]

tabres1 = []
print("tableau yh")
for k1 in range(1400):
    tabres1.append(np.real(yh)[k1][1])
tabres2 = []
for k2 in ybprime:
    tabres2.append(k2)

tabres = []
minimousse = 0.3
for k in range(1400):
    tabres.append(tabres1[k]+tabres2[k])
    if tabres1[k]+tabres2[k] < minimousse:
        minimousse = tabres1[k]+tabres2[k]
print(minimousse)
plt.plot(xb, tabres, 'g')
print(plt.ylim())
plt.show()
```

-:--- **zigouigoui.py** Bot L34 (Python)

Wrote /home/vella-chemla/Desktop/ne-te-decourage-pa

Figure 1



emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

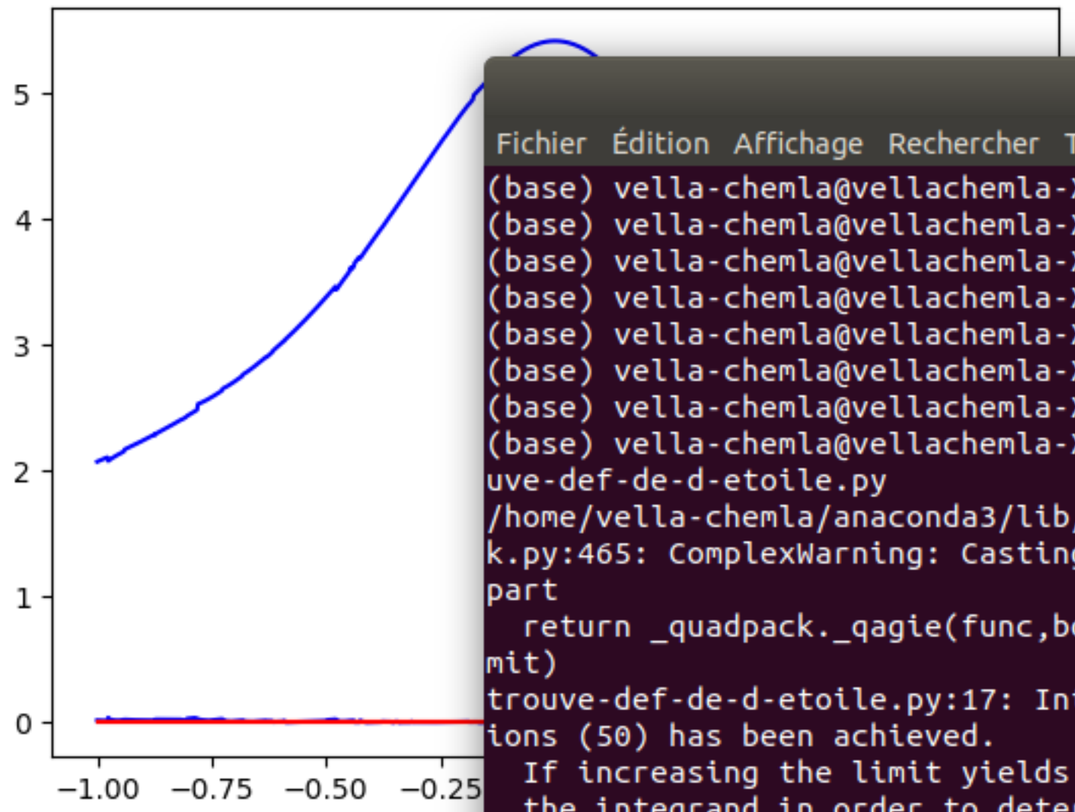
```
import numpy as np
from scipy.special import sici as SI
from scipy.integrate import quad
import matplotlib.pyplot as plt

def cmplx(t):
    return np.complex(t[0],t[1])

def delta(rho):
    r1 = 2.0 * np.pi * (rho + 1.0)
    r2 = 2.0 * np.pi * (rho - 1.0)
    z1 = cmplx(SI(r1))/r1
    z2 = cmplx(SI(r2))/r2
    return 2.0 * np.sqrt(rho) * (z1 + z2)

def delta_hat(t):
    return quad(lambda rho: delta(rho) * 2 * np.cos(t * np.log(rho))/rho, 1, np.
inf)

x = np.linspace(-1, 1, 1000)
y = [delta_hat(r) for r in x]

plt.plot(x, np.real(y), 'b', x, np.imag(y), 'r')
print(plt.ylim())
plt.show()
```

-:--- **trouve-def-de-d-etoile.py** All L23 (Python)

Wrote /home/vella-chemla/Desktop/ne-te-decourage-pas/trouve-def-de-d-etoile.py

```
  If increasing the limit yields no improvement it is advised to ana
  the integrand in order to determine the difficulties.  If the posi
  local difficulty can be determined (singularity, discontinuity) or
  probably gain from splitting up the interval and calling the integ
  on the subranges.  Perhaps a special-purpose integrator should be
  return quad(lambda rho: delta(rho) * 2 * np.cos(t * np.log(rho))/r
f)
(-0.2710550183614743, 5.69215538559096)
```

Figure 1



emacs25@vellachemla-X510UA

dit Options Buffers Tools Python Help

```
ort numpy as np
m scipy.special import sici as SI
m scipy.integrate import quad
```

Terminal

Fichier Édition Affichage Rechercher Terminal Aide

```
(base) vella-chemla@vellachemla-X510UA:~/Desktop/ne-te-decourage-pas$
(base) vella-chemla@vellachemla-X510UA:~/Desktop/ne-te-decourage-pas$
(base) vella-chemla@vellachemla-X510UA:~/Desktop/ne-te-decourage-pas$
(base) vella-chemla@vellachemla-X510UA:~/Desktop/ne-te-decourage-pas$
(base) vella-chemla@vellachemla-X510UA:~/Desktop/ne-te-decourage-pas$
(base) vella-chemla@vellachemla-X510UA:~/Desktop/ne-te-decourage-pas$
(base) vella-chemla@vellachemla-X510UA:~/Desktop/ne-te-decourage-pas$
(base) vella-chemla@vellachemla-X510UA:~/Desktop/ne-te-decourage-pas$ python tro
uve-def-de-d-etoile.py
/home/vella-chemla/anaconda3/lib/python3.8/site-packages/scipy/integrate/quadpac
k.py:465: ComplexWarning: Casting complex values to real discards the imaginary
part
  return _quadpack._qagie(func,bound,infbounds,args,full_output,epsabs,epsrel,li
mit)
trouve-def-de-d-etoile.py:17: IntegrationWarning: The maximum number of subdivis
ions (50) has been achieved.
  If increasing the limit yields no improvement it is advised to analyze
  the integrand in order to determine the difficulties.  If the position of a
  local difficulty can be determined (singularity, discontinuity) one will
  probably gain from splitting up the interval and calling the integrator
  on the subranges.  Perhaps a special-purpose integrator should be used.
  return quad(lambda rho: delta(rho) * 2 * np.cos(t * np.log(rho))/rho, 1, np.in
f)
```

```
(t * np.log(rho))/rho, 1, np.
```

-:--- **trouve-def-de-d-etoile.py** All L22 (Python)

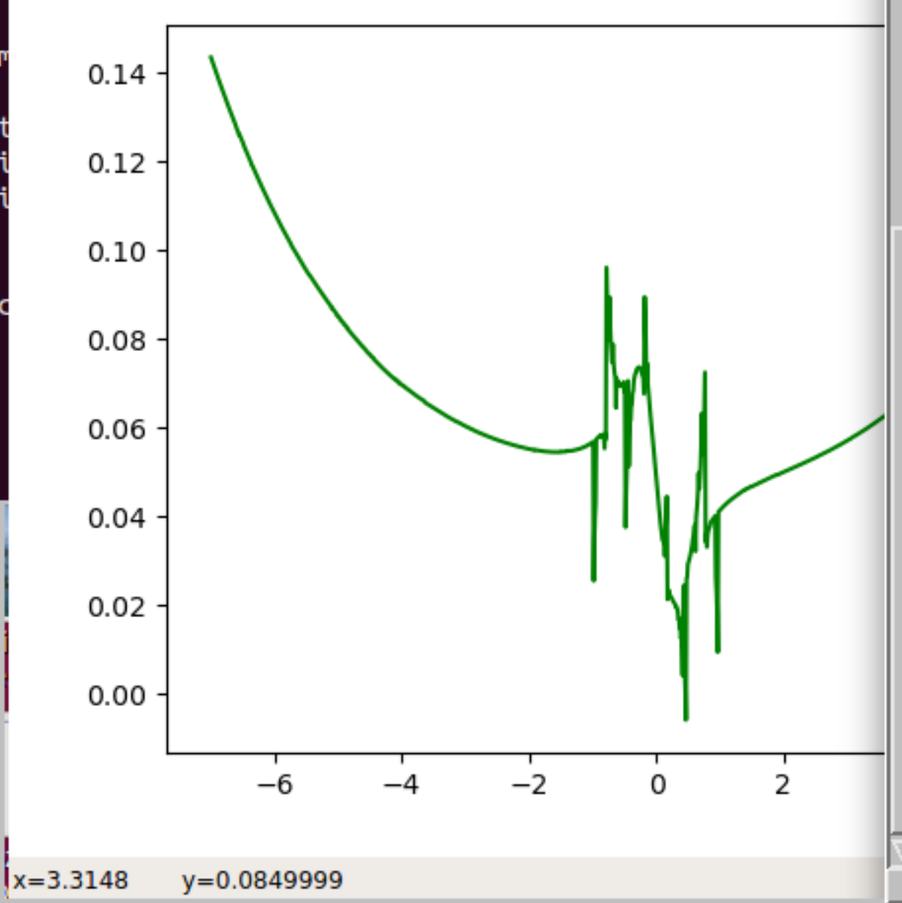
Wrote /home/vella-chemla/Desktop/ne-te-decourage-pas/trouve-def-de-d-etoile.py

Terminal

Fichier Édition Affichage Rechercher Terminal Aide

```
ybprime
[0.10717933 0.10574728 0.10431318 ... 0.10287701 0.10431318 0.10574728]
/home/vella-chemla/anaconda3/lib/python3.8/site-packages/scipy/integrate/quadpac
k.py:465: ComplexWarning: Casting complex values to real discards the imaginary
part
  return _quadpack._qagie(func,bound,infbounds,
mit)
zigouigoui.py:23: IntegrationWarning: The integ
wly convergent.
  return quad(lambda rho: delta(rho) * 2 * np.c
f)
zigouigoui.py:23: IntegrationWarning: The maxim
s been achieved.
  If increasing the limit yields no improvement
  the integrand in order to determine the diffi
  local difficulty can be determined (singulari
  probably gain from splitting up the interval
  on the subranges.  Perhaps a special-purpose
  return quad(lambda rho: delta(rho) * 2 * np.c
f)
tableau yh
-0.005919557842045453
(-0.013392390070540695, 0.15100991895635463)
```

Figure 1



emacs25@vellachemla-X510U

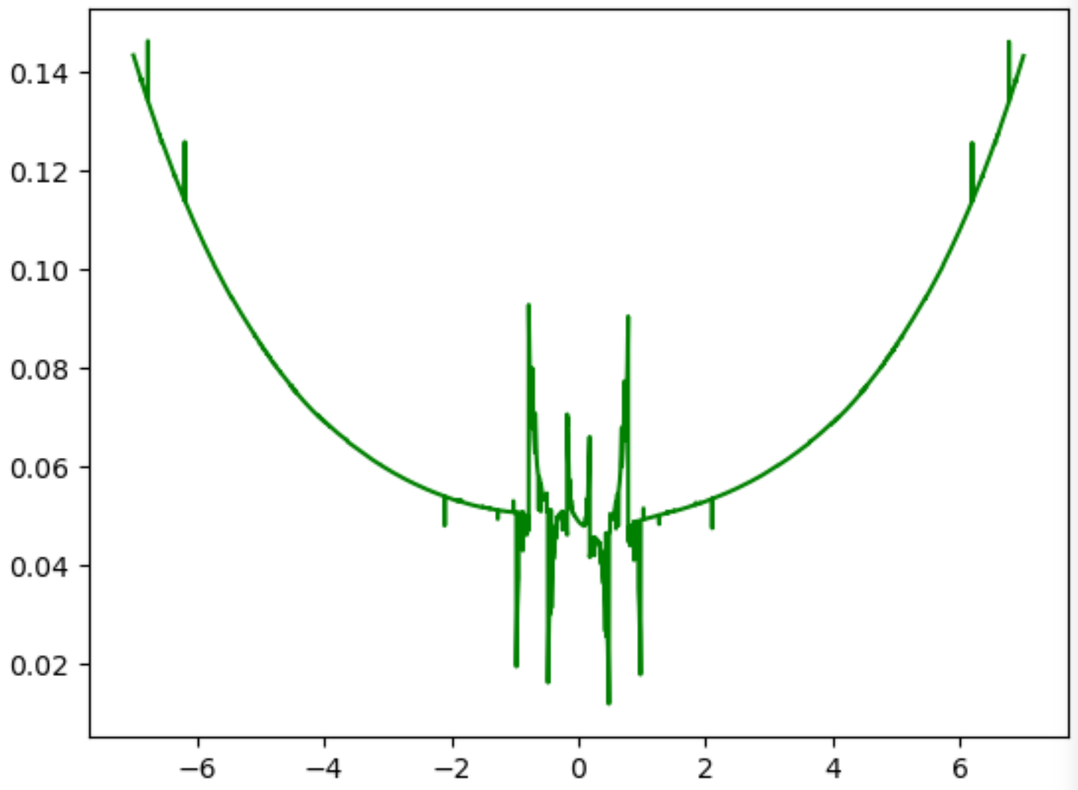
File Edit Options Buffers Tools Python Help

```
    return quad(lambda rho: delta(rho) * 2 * np.cos(
inf)

xb = np.arange(-7,7,0.01)
yb = fctthetariemannsiegel(xb)
ybprime = 2*scipy.misc.derivative(fctthetariemannsie
print(plt.ylim())
print("ybprime")
print(ybprime)

xh = np.linspace(-7, 7, 1400)
yh = [delta_hat(r) for r in xh]

tabres1 = []
print("tableau yh")
for k1 in range(1400):
    tabres1.append(np.real(yh)[k1][0])
tabres2 = []
for k2 in ybprime:
    tabres2.append(k2)

tabres = []
minimousse = 0.3
for k in range(1400):
    tabres.append(tabres1[k]+tabres2[k])
    if tabres1[k]+tabres2[k] < minimousse:
        minimousse = tabres1[k]+tabres2[k]
print(minimousse)
plt.plot(xb, tabres, 'g')
print(plt.ylim())
plt.show()
```

-:--- **zigouigoui.py** Bot L38 (Python)

Wrote /home/vella-chemla/Desktop/ne-te-decourage-pas

Figure 1




```
  return quad(lambda rho: delta(rho) * 2 * np.cos(t * np.log(rho))/rho
f)
tableau yh
0.011892124812775684
7478
(0.00516476069810951, 0.15316677122076533)
```

emacs25@vellachemla-X510UA

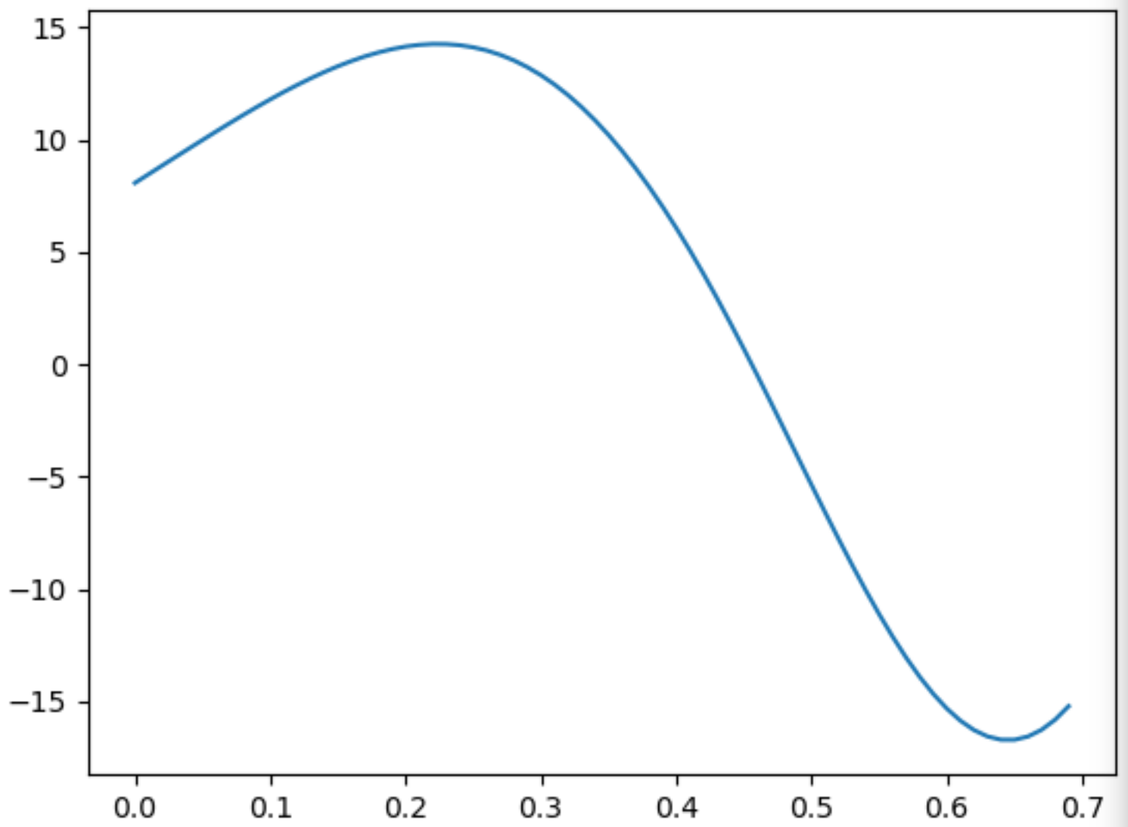
File Edit Options Buffers Tools Python Help

```
def delta_hat(t):
    return quad(lambda rho: delta(rho) * 2 * np.cos(t * np.log(rho))/rho, 1, np.
inf)

xb = np.arange(-7,7,0.001)
yb = fctthetariemannsiegel(xb)
ybprime = 2*scipy.misc.derivative(fctthetariemannsiegel, xb, n=1, dx=0.001)
print(plt.ylim())
print("ybprime")
print(ybprime)

xh = np.linspace(-7, 7, 14000)
yh = [delta_hat(r) for r in xh]

tabres1 = []
print("tableau yh")
for k1 in range(14000):
    tabres1.append(np.real(yh)[k1][0])
tabres2 = []
for k2 in ybprime:
    tabres2.append(k2)

tabres = []
minimousse = 0.3
gardek = 0
for k in range(14000):
    tabres.append(tabres1[k]+tabres2[k])
    if tabres1[k]+tabres2[k] < minimousse:
        minimousse = tabres1[k]+tabres2[k]
        gardek = k
print(minimousse)
print(gardek)
plt.plot(xb, tabres, 'g')
```

-:--- **zigouigoui.py** 39% L27 (Python Ovwrt)

Overwrite mode enabled in current buffer

Figure 1



emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
import matplotlib.pyplot as plt
import numpy as np
import scipy as sci
import scipy.special as sp
import scipy.misc
from scipy.special import sici as SI
from scipy.integrate import quad

def cmplx(t):
    return np.complex(t[0],t[1])

def delta(rho):
    r1 = 2.0 * np.pi * (rho + 1.0)
    r2 = 2.0 * np.pi * (rho - 1.0)
    z1 = cmplx(SI(r1))/r1
    z2 = cmplx(SI(r2))/r2
    return 2.0 * np.sqrt(rho) * (z1 + z2)

def deltaexp(rho):
    return delta(abs(np.exp(rho)))

def Qplus(x):
    return (-1)*scipy.misc.derivative(deltaexp, x, n=2, dx=1e-6)+0.25

x = np.arange(1e-5,np.log(2),0.01)
print("toto")
y = np.array([Qplus(xx) for xx in x])
print(y.real)
plt.plot(x,y.real)
plt.show()
```

-:--- **qplusdeltaexp.py** All L28 (Python)

Wrote /home/vella-chemla/Desktop/ne-te-decourage-pas/qplusdeltaexp.py

emacs25@vellachemla-X510

File Edit Options Buffers Tools Help

```
50 --> 13   19
52 --> 11   23
54 --> 11   13   17   23
56 --> 13   19
58 --> 11   17   29
60 --> 13   17   19   23   29
62 --> 19   31
64 --> 11   17   23
66 --> 13   19   23   29
68 --> 31
70 --> 11   17   23   29
72 --> 11   13   19   29   31
74 --> 13   31   37
76 --> 17   23   29
78 --> 11   17   19   31   37
80 --> 13   19   37
82 --> 11   23   29   41
84 --> 11   13   17   23   31   37   41
86 --> 13   19   43
88 --> 17   29   41
90 --> 11   17   19   23   29   31   37   43
92 --> 13   19   31
94 --> 11   23   41   47
96 --> 13   17   23   29   37   43
98 --> 19   31   37
100 --> 11   17   29   41   47
```

-:--- **resdiffdelog** All L1 (Fundamental)

5 1

emacs25@vellachemla-X510UA

File Edit Options Buffers Tools Python Help

```
import math
from math import log, floor, sqrt

def prime(atester):
  pastrouve = True ; k = 2 ;
  if (atester in [0,1]): return False ;
  if (atester in [2,3,5,7]): return True ;
  while (pastrouve):
    if ((k * k) > atester): return True
    else:
      if ((atester % k) == 0): return False
      else: k=k+1

nmax = 100
for n in range(50,nmax+2,2):
  print(str(n)+" --> ", end='')
  for x in range(3,n//2+1,2):
    rac = floor(sqrt(nmax))
    if (x > rac):
      t = 1.0
      for p in range(3, rac, 2):
        if prime(p):
          t = t*(math.log(x)-math.log(p)-math.log(math.floor(x/p)))
          t = t*(math.log(n-x)-math.log(p)-math.log(math.floor((n-x)/p)))
      if (t > 0.0000000001):
        print(str(x)+"   ", end='')
  print('')
```

-:--- **diffdelog.py** All L1 (Python)

5 1

Affectée (Denise Vella-Chemla, Toussaint 2020)

Je voudrais ici revenir sur des programmes qui m'ont plu, même s'ils sont inefficaces. Je les apprécie plutôt parce qu'ils me rappellent une idée, un concept, que j'ai réussi à implémenter.

1) Crible d'Eratosthène de deux façons

Ci-après, deux façons de mettre en œuvre le crible d'Eratosthène pour compter ou écrire les nombres premiers ;

a) le crible classique

Le premier programme est frugal, il n'utilise essentiellement qu'une ligne :

```
import time
import math
from math import sqrt

tps1 = time.time()
def premiers(n):
    return [p for p in range(2,n) if all([p%q for q in range(2,int(sqrt(p)))])]
pix=premiers(1000)
print(pix)
print("pix "+str(len(pix)))
print("Temps d execution : %s secondes..." % (time.time()-tps1))
```

b) par la Formule de Legendre

Le second programme est basé sur la formule de Legendre, qui fait intervenir la fonction de Möbius ; il s'agit d'appliquer le principe d'inclusion/exclusion (pour exprimer grossièrement l'idée, on enlève les multiples de 2, les multiples de 3, mais ayant de ce fait enlevé deux fois au lieu d'une seule fois les multiples de 6, qui sont à la fois des multiples de 2 et des multiples de 3, on rajoute le nombre des multiples de 6, etc...). L'idée derrière le second programme est la suivante : on prend l'ensemble des nombres premiers jusqu'à la racine de n, et on fabrique à partir des éléments de cet ensemble tous les produits possibles dont tous les facteurs sont différents. S'il y a k éléments dans l'ensemble, on peut utiliser, pour identifier chaque produit à calculer, un mot de k booléens qui dit pour chaque élément s'il est l'un des facteurs du produit en cours de calcul ou pas. Un raffinement supplémentaire consiste à utiliser les codes de Gray (algorithme de Knuth) pour énumérer tous les mots booléens par le "flip" d'un seul bit pour passer d'un mot à un autre. La formule de calcul du nombre de nombres premiers inférieurs ou égaux à x, notée $\pi(x)$ s'écrit alors $\pi(x) = \pi(\sqrt{x}) + \sum_{d|P} \mu(d) \left\lfloor \frac{x}{d} \right\rfloor - 1$

avec $P = \prod_{p \leq \sqrt{x}} p$.

La fonction de Möbius, $\mu(d)$, multiplicative, vaut -1 pour le produit d'un nombre impair de nombres premiers tous différents, elle vaut 1 pour le produit d'un nombre pair de nombres premiers tous différents, et elle vaut 0 sinon, sauf pour 1 pour lequel elle vaut 1 ($\mu(1) = 1$).

```python
import bisect, math, time
primes = [2]
def next_prime(n):
    while any([n % d == 0 for d in primes]): n += 1
    return n
def add_primes(n):
    r = math.sqrt(n)
    while primes[-1] < r: primes.append(next_prime(primes[-1] + 1))
def pi(n):
    # l'algorithme G de Knuth (code binaire de Gray) est utilis\'{e} pour g\'{e}n\'{e}rer
    # tous les sous-ensembles d'un ensemble
    m = bisect.bisect_right(primes, math.sqrt(n))
    a = [0 for _ in range(m)]
    s, mu = m, 1
    while (True):
        d = math.prod([primes[i] for i in range(m) if a[i]])
        s, mu = s + mu * math.floor(n/d), -mu
        j = 0 if mu == -1 else a.index(1) + 1
        if j == m: break
        a[j] = 1 - a[j]
    return s

add_primes(10000)
for k in range(1, 11):
    n = 1000*k
    t0 = time.time(); p = pi(n); t1 = time.time()
    print('n = {:5d}, pi = {:4d}, time = {:10.6f} s'.format(n, p, t1-t0))
```

2) Par le calcul des sommes de diviseurs “à la Euler”, en utilisant la récurrence de la séquence A000203 de l’OEIS

On peut aussi trouver les nombres premiers en utilisant une formule récurrente pour le calcul de la somme des diviseurs fournie par Euler dans son article *Découverte d’une loi tout extraordinaire des nombres par rapport à la somme de leurs diviseurs* [1]. Plutôt que d’implémenter l’algorithme tel que proposé par Euler [2], on peut utiliser la formule récurrente de la séquence A000203 de calcul de la somme des diviseurs, fournie par D. Giard dans la partie FORMULA en bas de page, sur le site de l’OEIS [3] qui se programme par exemple ainsi :

```c
#include <iostream>

int main (int argc, char* argv[]) {
  int n, k, somme ;
  int sigma[120] ;

  sigma[1] = 1 ;
  for (n = 2 ; n <= 100 ; ++n) {
    somme = 0 ;
    for (k=1 ; k < n ; k++)
      somme = somme+(-(n*n)+5*k*n-5*k*k)*sigma[k]*sigma[n-k] ;
    sigma[n] = (12*somme)/(n*n*(n-1)) ;
    if (sigma[n] == n+1)
      std::cout << n << "  " ;
  }
}
```

3) Par le nombre de résidus quadratiques dont Gauss a démontré qu’il est exactement égal à $\frac{p-1}{2}$ pour les nombres premiers

On peut aussi utiliser le fait que Gauss a démontré qu’un nombre premier p, contrairement à un nombre composé, est caractérisé par le fait qu’exactement la moitié (i.e. un ensemble de cardinal $\frac{p-1}{2}$) des nombres strictement compris entre 0 et p sont des résidus quadratiques de p (sont des

1. L. EULER. *Découverte d’une loi tout extraordinaire des nombres par rapport à la somme de leurs diviseurs*. Éd. Commentationes arithmeticae 2, p.639, 1849.
2. On trouve le programme initial, écrit en décembre 2006, ici http://denisevellachemla.eu/Algo-d-Euler-somme-div-Decouv-loi.pdf ou en annexe de http://denise.vella.chemla.free.fr/noel2006.pdf.
3. Online Encyclopedia of Integer Sequences, suite A000203, https://oeis.org/A000203.

carrés modulo p). On calcule les carrés successifs en ajoutant les nombres impairs successifs [4], et on marque les nombres qui sont effectivement des carrés modulaires dans un tableau de booléens.

```
import time
import numpy

pix = 0
tps1 = time.time()
nmax = 1000
marque = numpy.zeros(nmax)
for p in range(2,nmax):
      nbsol = 0
      somme = 0
      for x in range(1,p):
          marque[x] = False
      for x in range(0,int((p-1)/2)):
          somme = (somme +(2*x+1)) % p ;
          marque[somme] = True
      for x in range(1,p):
        if (marque[x]):
          nbsol = nbsol+1
      if (nbsol == (p-1)/2):
          print (p,end=' ')
          pix = pix+1
print("pix "+str(pix))
print("Temps d execution : %s secondes..." % (time.time()-tps1))
```

4) En comptant des quotients entiers ou pas

On peut aussi utiliser le fait que les divisions entières de n par tous les nombres qui lui sont inférieurs "tombent pile juste sur le quotient d'après" lorsqu'il y a divisibilité et dans ces cas seulement. Ce qui fait qu'un nombre premier, exactement selon la manière dont on le définit, n'a que 2 divisions qui "tombent juste", notamment relativement au nombre qui le précède, ce qui fait que la différence entre leurs deux sommes de quotients entiers est égale à 2.

```
import time

tps1=time.time()
pix = 0
somme = 0
for x in range(1,1001):
    sommeprec = somme
    somme = 0
    for k in range(1,x+1):
        somme = somme+int(x/k)
    if (somme-sommeprec == 2):
        print(x, end=' ')
        pix=pix+1
print("pix "+str(pix))
print("Temps d execution : %s secondes..." % (time.time()-tps1))
```

5) En utilisant la trace des puissances d'une matrice circulante

Enfin, voici un programme que j'affectionne particulièrement, même s'il est totalement inefficace : il s'agit d'utiliser des matrices circulantes. Il faut voir le fait de "barrer un nombre sur k lors de l'exécution du crible d'Eratosthène" comme équivalent au fait de calculer les puissances d'une matrice circulante de taille $k \times k$, dont des sous-matrices reviennent périodiquement, identiques à elles-mêmes, lorsqu'on élève la matrice initiale aux différentes puissances de 2 à k.

4. https://fr.wikipedia.org/wiki/Preuve$_s ans_m ots$.

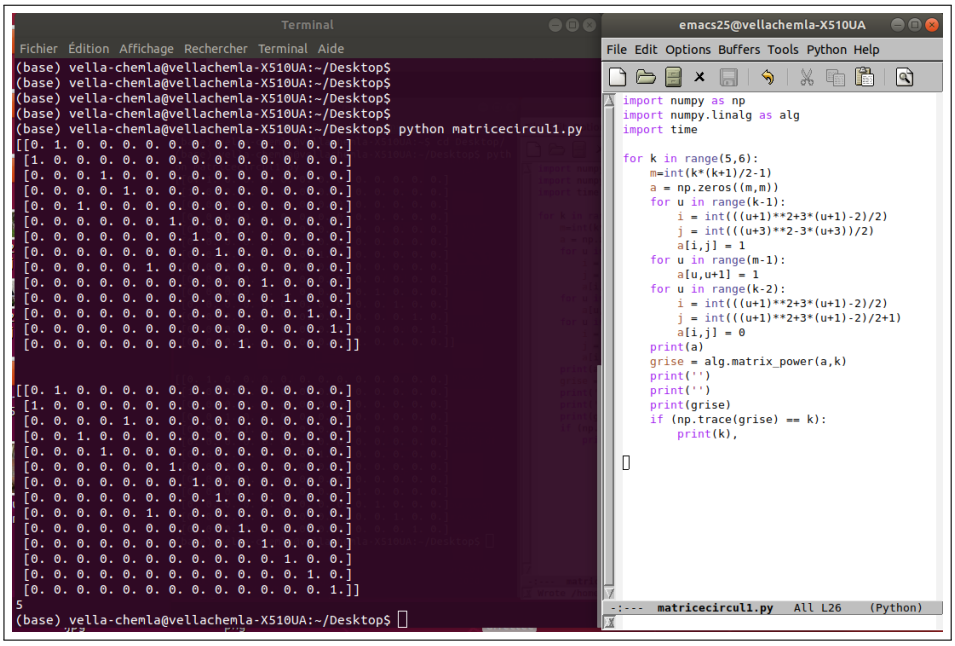
Voici la forme générale de la matrice G.

$$\begin{pmatrix}
0 & 1 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
1 & 0 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & 0 & 1 & 0 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & 0 & 0 & 1 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & 1 & 0 & 0 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots & 0 & 1 & 0 & 0 & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots & 0 & 0 & 1 & 0 & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots & 0 & 0 & 0 & 1 & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots & 1 & 0 & 0 & 0 & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & 0 & 1 & 0 & 0 & 0 & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & 0 & 0 & 1 & 0 & 0 & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & 0 & 0 & 0 & 1 & 0 & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & 0 & 0 & 0 & 0 & 1 & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & 1 & 0 & 0 & 0 & 0 & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots
\end{pmatrix}$$

Tous les ... sont des 0.

Selon cette modélisation, un nombre k s'avère premier si, lorsqu'on élève "sa" matrice de la forme ci-dessus (qui correspond en quelque sorte à une factorielle puisqu'elle contient toutes les sous-matrices possibles des tailles comprises entre 2 et k) à la puissance k, on obtient une matrice de trace k. La copie d'écran d'exécution montre la matrice initiale et la matrice élevée à la puissance k pour $k = 5$.

```python
import numpy as np
import numpy.linalg as alg
import time

for k in range(5,6):
    m=int(k*(k+1)/2-1)
    a = np.zeros((m,m))
    for u in range(k-1):
        i = int(((u+1)**2+3*(u+1)-2)/2)
        j = int(((u+3)**2-3*(u+3))/2)
        a[i,j] = 1
    for u in range(m-1):
        a[u,u+1] = 1
    for u in range(k-2):
        i = int(((u+1)**2+3*(u+1)-2)/2)
        j = int(((u+1)**2+3*(u+1)-2)/2+1)
        a[i,j] = 0
    print(a)
    grise = alg.matrix_power(a,k)
    print('')
    print('')
    print(grise)
    if (np.trace(grise) == k):
        print(k),
```



6) Calcul exact de $\pi(x)$ par des séquences fractales de valuations p-adiques (au sens simple, qui n'ont rien à voir avec la notion de corps p-adiques)

Pour terminer, la formule de calcul exact de $\pi(n)$ que j'avais proposée en septembre 2018, basée sur l'utilisation des valuations p-adiques [5].

On a pour tout entier $m \geq 2$:

$$\begin{aligned} f(m) &= \sum_{n=2}^{\sqrt{m}} v(m,n) \\ \pi(m) &= \sum_{n=2}^{\sqrt{m}} f(m) \\ &= \sum_{n=2}^{\sqrt{m}} \Big\lfloor \sum_{n=1}^{\sqrt{m}} v(m,n) \Big\rfloor \end{aligned}$$

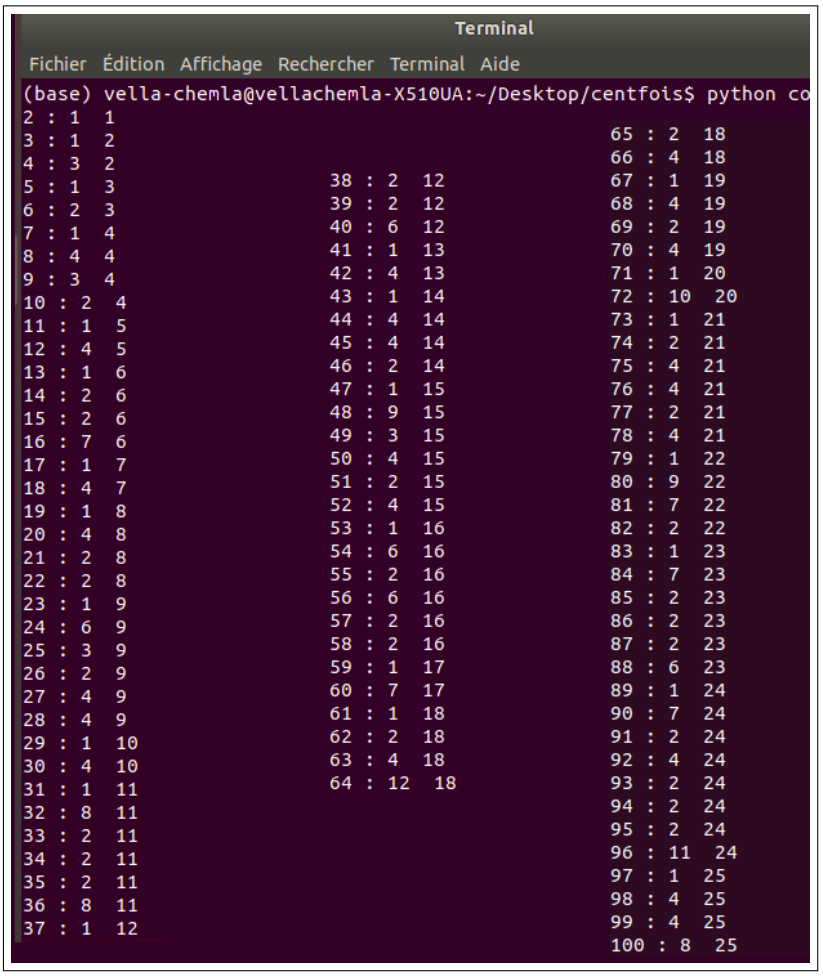
Voici le programme de calcul de la formule et son résultat.

```
from math import floor, sqrt

def vp(n, p):
    if (p == 1):
        return 1
    if ((n % p) != 0):
        return 0
    else:
        return vp(n/p,p)+1

nmax = 100 ;
pix = 0 ;
for m in range(2, nmax+1):
    print(str(m)+" : ", end='')
    somme = 0
    rac = floor(sqrt(m))
    for n in range(1, rac+1):
        somme = somme+vp(m, n)
    print(str(somme)+"  ", end=''),
    pix = pix+floor(1.0/float(somme))
    print(str(pix)+"  "),
```

On rappelle quelques propriétés de la valuation p-adique :

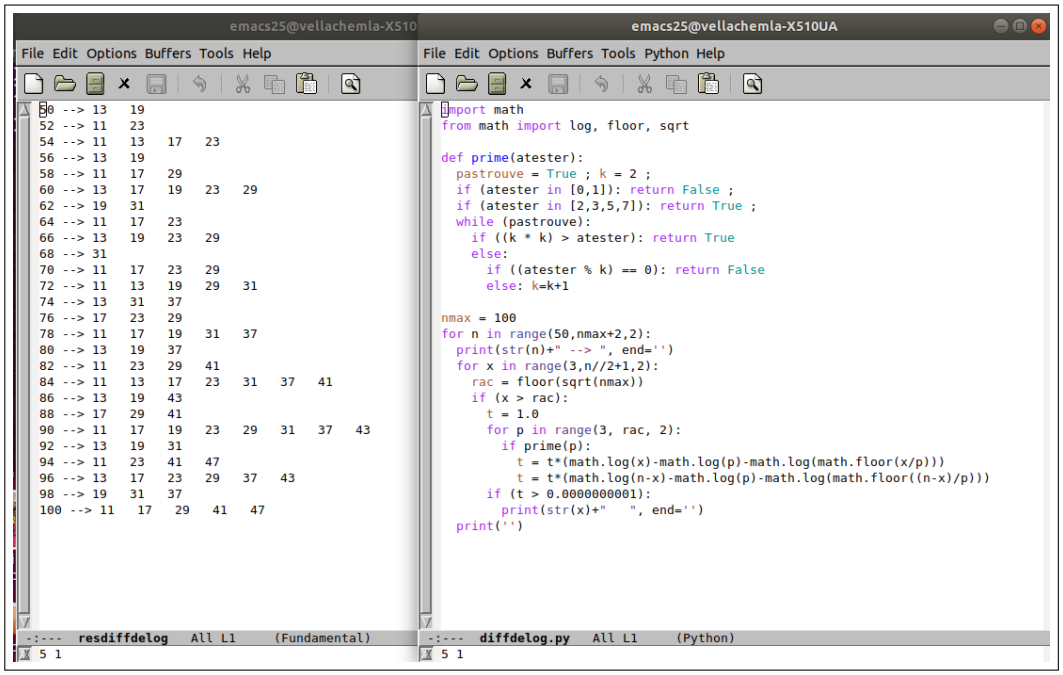
- Soient m et n deux entiers ; m divise n si $v_p(m) \leq v_p(n)$ pour tout nombre premier p ;
- si a et b sont des entiers non nuls,
  $v_p(pgcd(a,b)) = min(v_p(a), v_p(b))$
  $v_p(ppcm(a,b)) = max(v_p(a), v_p(b))$;
- si a et b sont des entiers non nuls et p un nombre premier quelconque,
  $v_p(ab) = v_p(a) + v_p(b)$
  $v_p(a+b) \geq min(v_p(a), v_p(b))$.

5. découvertes dans http://denise.vella.chemla.free.fr/fevrier2006.pdf, j'ai travaillé une première fois sur la formule proposée ici dans ces deux notes http://denise.vella.chemla.free.fr/fracto.pdf et http://denise.vella.chemla.free.fr/fractosimple.pdf.



7) En passant par les log

Trouver les décomposants de Goldbach en faisant un "détour par les log" (quand a divise b, log(b)-log(a)-log(floor(b/a)) est nul, et 0 est absorbant pour la multiplication)



8) Avec ce que j'ai envie d'appeler des diracs (?) : des sortes de pointes de peigne