

S'enhardir : il se passe quoi sur la droite critique ?¹, Denise Vella-Chemla, 23 novembre 2024

Quel est notre objectif courant ? On cherche une opération qui permettrait de rassembler certains termes, par paquets (de 2 peut-être ?), dans la somme bien connue

$$\sum_{n=1}^{\infty} \frac{1}{n^s}.$$

Dans cette vaine tentative, on trouve dans le site de Gérard Villemin comment trouver la partie réelle et la partie imaginaire d'un entier élevé à une puissance complexe "directement" ainsi :

$$\begin{aligned} n^{\frac{1}{2}+ib} &= (\cosh(a \ln n) + \sinh(a \ln n)) \cdot (\cos(b \ln n) + i \sin(b \ln n)) \\ \mathbf{a} &= (\cos(i a \ln n) - i \sin(i a \ln n)) \cdot (\cos(b \ln n) + i \sin(b \ln n)) \end{aligned}$$

On vérifie cela par programme au cas où quelque chose apparaisse. Voici le programme utilisé :

```
import mpmath ; from mpmath import * ; import math ; from math import cos,sin,cosh,sinh,log
lesa = [0.48,0.49,0.5,0.51,0.52]
lesb = [14.11,14.12,14.13,14.14,14.15]
for n in range(1,25):
    #for a in lesa:
    for a in [0.5]:
        #for b in lesb:
        for b in [14.13]:
            res = n**(a+1j*b)
            print(n, '--> ', res)
            print('lesh = ',cosh(a*log(n))+sinh(a*log(n)))
            print('le cos = ',cos(b*log(n)))
            print('le sin = ',sin(b*log(n)))
            print('du coup, partie reelle = ',(cosh(a*log(n))+sinh(a*log(n)))*cos(b*log(n)))
            print(' et partie imaginaire = ',(cosh(a*log(n))+sinh(a*log(n)))*sin(b*log(n)))
        print('')
    print('')
```

Le résultat du programme est fourni en annexe. On voit apparaître les racines carrées comme parties réelles en sommant côté hyperbolique (le cosh et le sinh), ce qui est normal, on a $\exp(\frac{1}{2} \ln x) = \sqrt{x}$ par exemple.

On réalise par cette expérimentation l'existence de "nombres bases" qui permettent de calculer les racines carrées, ou les racines cubiques en les élevant à des puissances correspondant aux logarithmes népériens. On a par exemple

$$\begin{aligned} \text{Base_racine_2} &= \exp(1/2) = 1.6487212707... \\ \text{Base_racine_3} &= \exp(1/3) = 1.39561242509... \end{aligned}$$

On a $\text{Base_racine_2}^{\ln n} = \sqrt{n}$, ou bien $\text{Base_racine_3}^{\ln n} = \sqrt[3]{n}$, etc.

¹ou on ne nous dit pas tout ;-)

Le nombre base $\text{Base_racine}_k = \exp\left(\frac{1}{k}\right)$, élevé à la puissance le logarithme népérien d'un nombre n permet d'obtenir la racine k -ième de ce nombre n .

$$\text{Base_racine}_k^{\ln n} = \sqrt[k]{n}.$$

Le programme est fourni ci-après. Il permet d'observer cette propriété des nombres bases.

D'autre part, le programme calcule pour chaque partie imaginaire z d'un zéro non trivial de la fonction ζ de Riemann la fonction suivante :

$$f(z) = \exp\left(W\left(-\left(\frac{z}{2}\right)^2\right)\left(\frac{2\pi}{180}\right)\right)$$

Comme on réalise qu'il y a comme un tempo de valse [1, 1, 2, 3, 3, 4, 5, 5, 6, etc.], on met en regard des images obtenues par la fonction f des multiples de $2/3$ avec un tout petit (shift) décalage pour être raccord en fin de liste. La fonction qui a dû être utilisée $W(z)$ s'appelle fonction de Lambert, elle permet de trouver la solution y , lorsqu'on connaît x , d'une équation de la forme $y \ln y = x$.

```
import math ; from math import exp, pi, log,sqrt ; from scipy.special import lambertw

zeros = [ 14.1347251417346937904572519835625, 21.0220396387715549926284795938969,
25.0108575801456887632137909925628, 30.4248761258595132103118975305840,
32.9350615877391896906623689640747, 37.5861781588256712572177634807053,
40.9187190121474951873981269146334, 43.3270732809149995194961221654068,
48.0051508811671597279424727494277, 49.7738324776723021819167846785638,
52.9703214777144606441472966088808, 56.4462476970633948043677594767060,
59.3470440026023530796536486749922, 60.8317785246098098442599018245241,
65.1125440480816066608750542531836, 67.0798105294941737144788288965221,
69.5464017111739792529268575265547, 72.0671576744819075825221079698261,
75.7046906990839331683269167620305, 77.1448400688748053726826648563047,
79.3373750202493679227635928771161, 82.9103808540860301831648374947706,
84.7354929805170501057353112068276, 87.4252746131252294065316678509191,
88.8091112076344654236823480795095, 92.4918992705584842962597252418105,
94.6513440405198869665979258152080, 95.8706342282453097587410292192467,
98.8311942181936922333244201386224]

print('exp(0.5)**ln(2) = ',exp(0.5)**log(2))
print('sqrt(2) = ',sqrt(2))
print('exp(0.5)**ln(3) = ',exp(0.5)**log(3))
print('sqrt(3) = ',sqrt(3))
print('exp(0.5)**ln(5) = ',exp(0.5)**log(5))
print('sqrt(5) = ',sqrt(5))
print('la base des racines carrees est : ',exp(0.5),' : elevee a la puissance
log(n), elle fournit la racine carree de n.')
print('')
print('exp(1/3)**log(2) = ',exp(1/3)**log(2))
print('sqrt[3](2) = ',2**(1/3))
print('exp(1/3)**log(3) = ',exp(1/3)**log(3))
print('sqrt[3](3) = ',3**(1/3))
print('exp(1/3)**log(5) = ',exp(1/3)**log(5))
print('sqrt[3](5) = ',5**(1/3))
print('la base des racines cubiques est : ',exp(1/3),' : elevee a la puissance
log(n), elle fournit la racine cubique de n.')
```

```

premier = 1
for z in zeros:
    print(z, ' --> ', exp(lambertw(-((z/2)**2)*(2*pi)/180)), ' ', premier*(2/3+0.08))
    premier = premier+1

```

Voici son résultat :

14.134725141734695	-->	1.0772375996941315	0.7466666666666666
21.022039638771556	-->	1.918950321922641	1.4933333333333332
25.01085758014569	-->	2.485787905112672	2.2399999999999998
30.424876125859512	-->	3.3427514619545198	2.9866666666666664
32.93506158773919	-->	3.7730944772557673	3.7333333333333333
37.586178158825675	-->	4.623993042930546	4.4799999999999995
40.9187190121475	-->	5.2754106660147295	5.2266666666666667
43.327073280915	-->	5.767411052076091	5.9733333333333333
48.00515088116716	-->	6.773033808741858	6.7199999999999999
49.7738324776723	-->	7.170157261188014	7.4666666666666666
52.970321477714464	-->	7.911057195740718	8.2133333333333333
56.44624769706339	-->	8.750192709322919	8.9599999999999999
59.34704400260235	-->	9.476820016840236	9.7066666666666665
60.83177852460981	-->	9.857909171062472	10.4533333333333333
65.1125440480816	-->	10.991060813218892	11.2
67.07981052949417	-->	11.528786717393535	11.9466666666666665
69.54640171117398	-->	12.217950949632758	12.6933333333333332
72.0671576744819	-->	12.939310177227325	13.4399999999999998
75.70469069908393	-->	14.010383679015998	14.1866666666666666
77.1448400688748	-->	14.444197041546738	14.9333333333333332
79.33737502024937	-->	15.115206731375853	15.6799999999999998
82.91038085408603	-->	16.235821766491938	16.4266666666666666
84.73549298051705	-->	16.82112136372026	17.1733333333333332
87.42527461312523	-->	17.69948257769329	17.9199999999999998
88.80911120763446	-->	18.15866459715959	18.6666666666666664
92.49189927055849	-->	19.40463662233388	19.4133333333333333
94.65134404051989	-->	20.151331589735214	20.1599999999999997
95.87063422824531	-->	20.578169388811915	20.9066666666666666
98.83119421819369	-->	21.630211644248604	21.6533333333333332

Annexe : résultat du premier programme, pour vérifier les formules avec cosinus et sinus hyperboliques

```

1 --> (1+0j)
lesh = 1.0
le cos = 1.0
le sin = 0.0
du coup, partie reelle = 1.0
et partie imaginaire = 0.0

```

2 --> (-1.3188208179886864-0.5105993047778777j)
lesh = 1.414213562373095
le cos = -0.9325471435697897
le sin = -0.3610482308775741
du coup, partie reelle = -1.3188208179886864
et partie imaginaire = -0.5105993047778777

3 --> (-1.7026318514933363+0.31787541314227746j)
lesh = 1.7320508075688772
le cos = -0.9830149577905086
le sin = 0.18352545534645742
du coup, partie reelle = -1.7026318514933363
et partie imaginaire = 0.31787541314227746

4 --> (1.4785766999206957+1.3467779855832307j)
lesh = 2.0
le cos = 0.7392883499603479
le sin = 0.6733889927916153
du coup, partie reelle = 1.4785766999206957
et partie imaginaire = 1.3467779855832307

5 --> (-1.6357872631371633-1.5245327250532306j)
lesh = 2.23606797749979
le cos = -0.7315463034206066
le sin = -0.681791761428404
du coup, partie reelle = -1.6357872631371633
et partie imaginaire = -1.5245327250532306

6 --> (2.407773296076462+0.4501419272863697j)
lesh = 2.449489742783178
le cos = 0.9829693319477565
le sin = 0.183769672280769
du coup, partie reelle = 2.407773296076462
et partie imaginaire = 0.4501419272863697

7 --> (-1.883459943740704+1.8581115790835234j)
lesh = 2.6457513110645907
le cos = -0.7118809450699439
le sin = 0.702300163780647
du coup, partie reelle = -1.883459943740704
et partie imaginaire = 1.8581115790835234

8 --> (-1.262313829719481-2.531119079636306j)
lesh = 2.82842712474619
le cos = -0.44629533449010284
le sin = -0.8948857326007423
du coup, partie reelle = -1.2623138297194807
et partie imaginaire = -2.5311190796363054

9 --> (2.7979104434392528-1.0824496064452904j)
lesh = 3.0
le cos = 0.9326368144797509
le sin = -0.36081653548176346
du coup, partie reelle = 2.7979104434392528
et partie imaginaire = -1.0824496064452904

10 --> (1.3788849469027102+2.845817334827573j)
lesh = 3.16227766016838
le cos = 0.4360417063532902
le sin = 0.899926458284515
du coup, partie reelle = 1.3788849469027105
et partie imaginaire = 2.8458173348275735

11 --> (-2.5887751822714606+2.0732204551507216j)
lesh = 3.3166247903554
le cos = -0.7805450860162132
le sin = 0.6250994870386172
du coup, partie reelle = -2.5887751822714606
et partie imaginaire = 2.0732204551507216

12 --> (-2.9455793927390714-1.8230639157942106j)
lesh = 3.4641016151377544
le cos = -0.8503155276586587
le sin = -0.5262732212668404
du coup, partie reelle = -2.9455793927390714
et partie imaginaire = -1.8230639157942106

13 --> (0.4116436689178993-3.5819756405983294j)
lesh = 3.6055512754639896
le cos = 0.114169411961816
le sin = -0.993461295356942
du coup, partie reelle = 0.41164366891789933
et partie imaginaire = -3.58197564059833

14 --> (3.432696664132815-1.4888228947901916j)
lesh = 3.741657386773941
le cos = 0.9174267735647724
le sin = -0.39790465584927737
du coup, partie reelle = 3.4326966641328145
et partie imaginaire = -1.4888228947901914

15 --> (3.269754966309667+2.0757414242369565j)
lesh = 3.8729833462074166
le cos = 0.8442471020464223
le sin = 0.5359541311401731
du coup, partie reelle = 3.2697549663096668
et partie imaginaire = 2.075741424236956

16 --> (0.37237811509675045+3.9826290988989914j)
lesh = 4.0
le cos = 0.09309452877418761
le sin = 0.9956572747247479
du coup, partie reelle = 0.37237811509675045
et partie imaginaire = 3.9826290988989914

17 --> (-2.8506298260430993+2.9789108068006147j)
lesh = 4.123105625617661
le cos = -0.6913792866065762
le sin = 0.722491994454874
du coup, partie reelle = -2.8506298260430993
et partie imaginaire = 2.9789108068006147

```
18 --> (-4.242640556183696-0.0010540518471141614j)
lesh = 4.242640687119285
le cos = -0.9999999691381858
le sin = -0.0002484424029388766
du coup, partie reelle = -4.242640556183696
et partie imaginaire = -0.0010540518471141614

19 --> (-3.1467916150920603-3.0162398000136372j)
lesh = 4.358898943540673
le cos = -0.7219235077140753
le sin = -0.691972867249725
du coup, partie reelle = -3.14679161509206
et partie imaginaire = -3.0162398000136363

20 --> (-0.36542982089876647-4.457180840620885j)
lesh = 4.47213595499958
le cos = -0.08171259205352151
le sin = -0.9966559347637954
du coup, partie reelle = -0.36542982089876647
et partie imaginaire = -4.457180840620885

21 --> (2.6161909053591517-3.7623855659296885j)
lesh = 4.582575694955841
le cos = 0.5708996598220649
le sin = -0.8210198404515269
du coup, partie reelle = 2.616190905359152
et partie imaginaire = -3.7623855659296894

22 --> (4.4727155265233-1.4123794882386964j)
lesh = 4.690415759823431
le cos = 0.9535861543096288
le sin = -0.3011203186584605
du coup, partie reelle = 4.4727155265233005
et partie imaginaire = -1.4123794882386966

23 --> (4.548988332160371+1.5187841037714367j)
lesh = 4.795831523312719
le cos = 0.9485296366328896
le sin = 0.3166883774771006
du coup, partie reelle = 4.548988332160371
et partie imaginaire = 1.5187841037714367

24 --> (2.953836256212608+3.9083054347739865j)
lesh = 4.898979485566357
le cos = 0.6029493009544873
le sin = 0.7977795061785525
du coup, partie reelle = 2.9538362562126084
et partie imaginaire = 3.9083054347739874
```