Matrices booléennes triangulaires, espace non commutatif de Goldbach
Denise Vella-Chemla, juin 2025.

Le contexte est la conjecture de Goldbach : on cherche à décomposer n un nombre pair $\geqslant 4$ en somme de 2 nombres premiers.

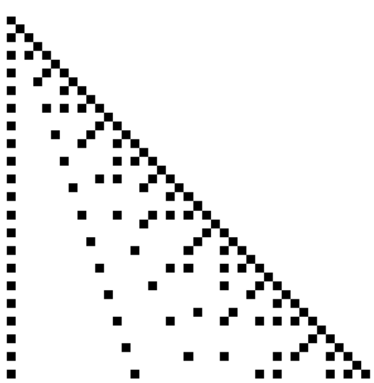
Cette note est destinée à fournir le programme de calcul des matrices contenant les décomposants de Goldbach et à signaler que l'ajout des éléments matriciels, de M et de sa transformée M′, plutôt que le $\vee$ (ou) booléen, permet de voir les nombres qui ont un pgcd différent de 1 avec n ainsi que ce qu'on avait appelé les séquences fractales d'entiers [1], par les nuances de couleurs offertes par la bibliothèque matplotlib de python. On fournit aussi le calcul du nombre d'éléments des matrices utilisées.

On a échangé l'étape qu'on avait appelée symétrisation, et qui consistait à permuter les coordonnées, par une symétrisation effectuée par calcul matriciel : cette étape est constituée successivement d'une rotation de 90° (`rot90` en python) dans le sens horaire de la matrice, suivie d'un "flip droite-gauche" (`fliplr` en python).

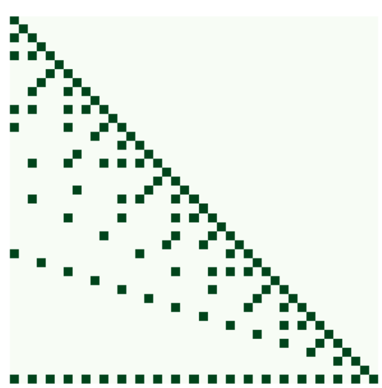
Le calcul des sommes par diagonales est maintenu dans la matrice somme.

L'ensemble des matrices sous forme d'un petit livre est téléchargeable ici :
https://denisevellachemla.eu/matricestropjolies.pdf.

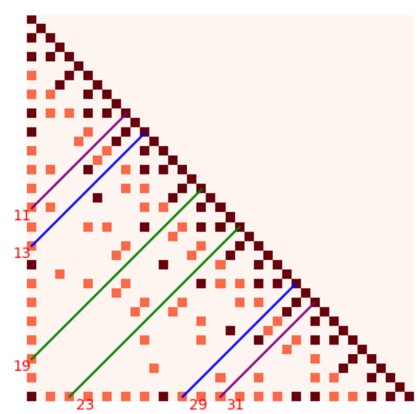
Les matrices pour $n = 42, 44, 46$



M

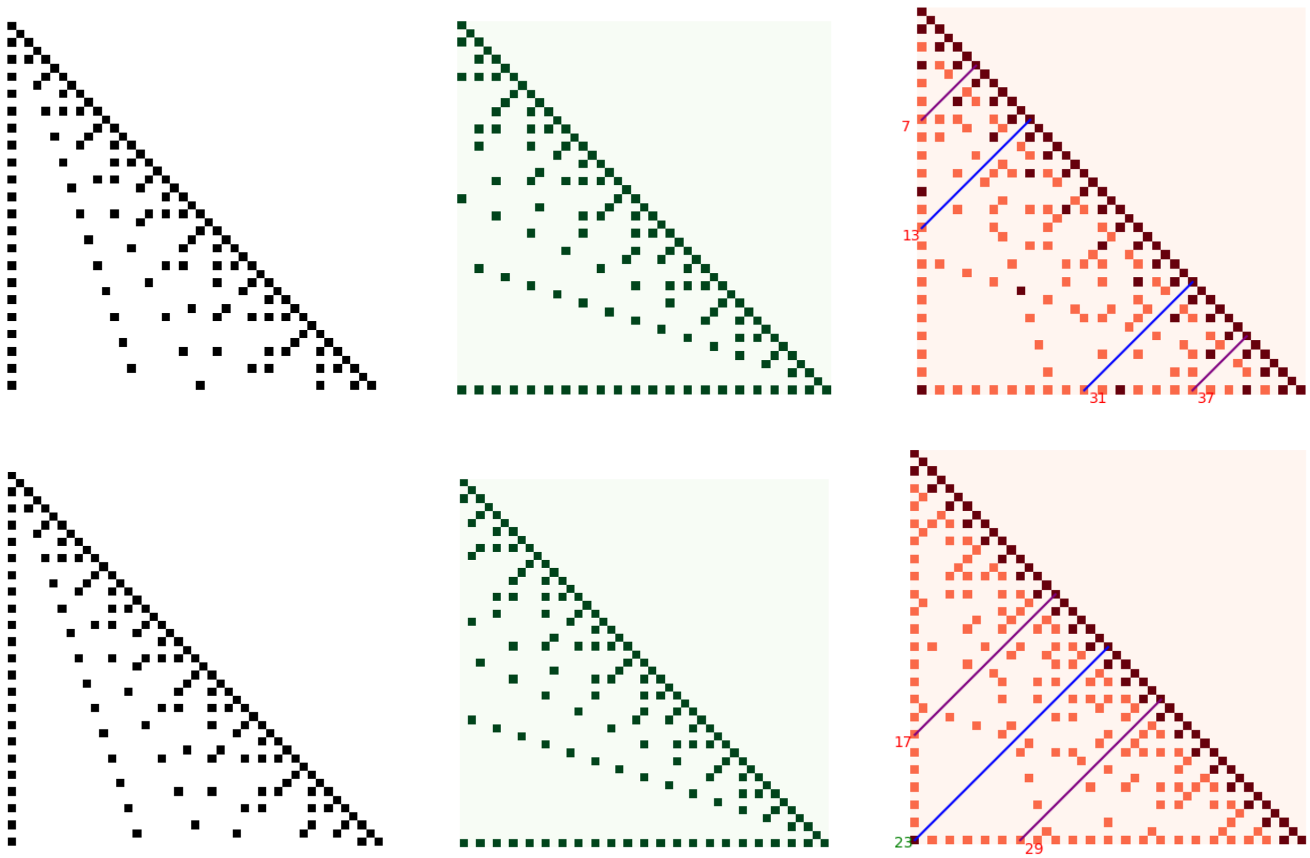


$\mathrm{M}' =$ `fliplr`(`rot90`(M))



$\mathrm{M} + \mathrm{M}'$

[1] ou ce qu'on avait appelé les "Diracs".



M $\qquad$ $\mathrm{M}' = \texttt{fliplr}(\texttt{rot90}(\mathrm{M}))$ $\qquad$ $\mathrm{M} + \mathrm{M}'$

Programme de calcul des matrices et de recherche des décomposants de Goldbach des nombres pairs de 4 à 102.

```
import numpy as np
import matplotlib.pyplot as plt
import math
from math import sqrt,floor

mescouleurs = ['purple','blue','green','yellow','orange','red']

def ecrimat (m,couleur) :
    plt.grid (False)
    plt.axis ('off')
    plt.imshow(m, cmap=couleur, aspect='equal', origin='upper')
    if couleur =='Grays':
        nomfic ='fig'+str(n) +'-1' ; plt.savefig (nomfic) ; plt.close()
    else :
        if couleur =='Greens':
            nomfic ='fig'+ str(n)+'-2' ; plt.savefig (nomfic) ; plt.close()
```

```
        elif couleur =='Reds':
            indice = 0
            for k in range (0 ,n-1, 2) :
                somme = 0
                x = 0
                y = k - x
                while y != 0:
                    somme = somme + m[y , x]
                    x = x+1
                    y = y-1
                print((k+2)//2,' --> ',int(somme))
                if somme == 3 and ((k+2)//2 >= floor(sqrt(n))):
                    plt.plot([(k+2)//2-1,0],[(k+2)//2-1,k],color=mescouleurs[indice])
                    plt.plot([n-1-(k+2)//2,n-2-k],[n-1-(k+2)//2,n-2],
                              color=mescouleurs[indice])
                    plt.annotate((k+2)//2, xy=(1,k+1),xytext =(-20,
                                          -2),textcoords ='offset points',
                                          color='red')
                    plt.annotate(n-(k+2)//2, xy=(n-k+1,n-3),xytext =(-15,
                                          -15),textcoords ='offset points',
                                          color='red')
                    indice=(indice+1) % len(mescouleurs)
                if somme == 4 and (k+2)//2 == n//2:
                    plt.plot([(k+2)//2-1,0],[(k+2)//2-1,k],color=mescouleurs[indice])
                    plt.annotate((k+2)//2, xy=(1,k+0.5),xytext =(-20,
                                          -2),textcoords ='offset points',
                                          color='green')
            nomfic ='fig'+ str ( n ) +'-3' ; plt.savefig ( nomfic ) ; plt.close ()
    plt.show()

for n in range(4,104,2):
    print('= ',n,' :::::::::')
    mn = np.zeros((n-1,n-1))
    for x in range(n-1):
        mn[x,x] = 1
    for ligne in range(2,n-1):
        x = 2*ligne-2
        y = 0
        while x < n-1 and y < n-1:
            mn[x,y] = 1
            x = x+ligne ; y = y+ligne
    miroir = np.rot90(mn)
    miroir2 = np.fliplr(miroir)
    ecrimat(mn,'Grays')
    ecrimat(miroir2,'Greens')
    matou = miroir2 + mn
    ecrimat(matou,'Reds')
```

On sait exactement combien les matrices M et M′ contiennent de booléens égaux à 1 (elles en contiennent le même nombre étant symétriques l'une de l'autre) ; on sait aussi exactement combien leur somme $M + M'$ contient d'éléments non nuls.

Le nombre de booléens à vrai dans les matrices M et M′ est (# désigne le cardinal ensembliste) :

$$\#\left\{k \text{ divise } x \text{ avec } 1 \leq k < n-1 \text{ et } k \leq x \leq n-k\right\}.$$

Le nombre d'éléments non nuls dans la somme des deux matrices $\mathrm{M}+\mathrm{M}'$ est :

$$\begin{aligned}&\#\{k \text{ divise } x \text{ avec } 1 \leq k < n-1 \text{ et } k \leq x \leq n-k \text{ et } k \text{ divise } n\}\\ &\qquad + 2 \times \#\{k \text{ ne divise pas } x \text{ avec } 1 \leq k < n-1 \text{ et } k \leq x \leq n-k \text{ et } k \text{ ne divise pas } n\}\end{aligned}$$