Maudits rectangles !

2021 - 2022

Elèves : Lise Beaumont-Thomas, Andréa Boucq, Elyna Coquet, Timothé Pollet (1ère générale), Charline Guicheney (Terminale générale)

Etablissement : Lycée de la Mer, Gujan – Mestras (33)

Encadrés par : Mme Delmaire

Chercheurs : Eric Sopena et François Dross (Université de Bordeaux)

Table des matières

I - Introduction

Nous considérons une grille de m × n (avec m lignes et n colonnes) cases. L'objectif est de combler toutes les cases de la grille avec des entiers naturels, sans former de rectangles dont les 4 coins contiennent le même entier.

La question principale est alors : pour une taille de grille m × n donnée, combien d'entiers au minimum nous faudra-t-il pour remplir cette grille ?
Afin d'y parvenir, nous avons rempli la totalité d'une grille avec le même entier. Par la suite, nous avons retiré un par un tous les entiers qui permettaient de former un rectangle interdit dans le but de déterminer le nombre de rectangles possibles dans une grille donnée.
En poursuivant nos recherches, nous avons conjecturé plusieurs formules, et enfin nous avons écrit un algorithme qui à partir de la longueur et de la largeur d'une grille retourne cette grille remplie de X ne formant aucun rectangle.

Par exemple, la grille ci-contre montre un rectangle interdit dans une grille 3 x 4.

2			2
2			2

Contrairement à la grille ci-dessus, voici une grille 4 x 6 correctement remplie avec 3 entiers :

1	1	3	1	3	3
2	3	1	3	1	3
3	2	2	3	3	1
3	3	3	2	2	2

II - Recherches

II.1 - Compter le nombre de rectangles en fonction d'une grille donnée

Pour commencer on va compter le nombre de rectangles qui se forment dans une grille quand on la remplit intégralement, le but étant de connaître le nombre de cases remplies qu'il nous resterait à retirer afin de ne plus avoir de rectangles.

Afin de compter le nombre total de rectangles dans une grille $m \times n$, nous commençons par multiplier par $m - 1$ afin de ne pas compter deux fois la case de départ puis $n - 1$.(1) Ainsi le nombre de rectangles réalisés à partir de cette case serait de :

$$(m - 1)(n - 1)$$

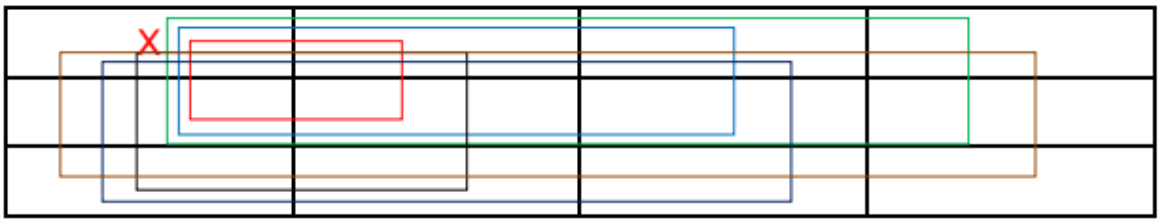
Ensuite, nous multiplions la formule précédente par $m \times n$ afin d'obtenir le nombre total de rectangles à partir de chaque case. (2) Nous obtenons alors :

$$mn\,(m - 1)(n - 1)$$

Mais chaque rectangle a déjà été compté quatre fois. Nous devons alors diviser la formule par quatre, ce qui nous donne :

$$mn\,(m - 1)(n - 1)/4 \text{ rectangles distincts}$$

Prenons pour exemple une grille 3x4 :
Première étape : On peut former (3-1)(4-1) = 6 rectangles à partir de la X .

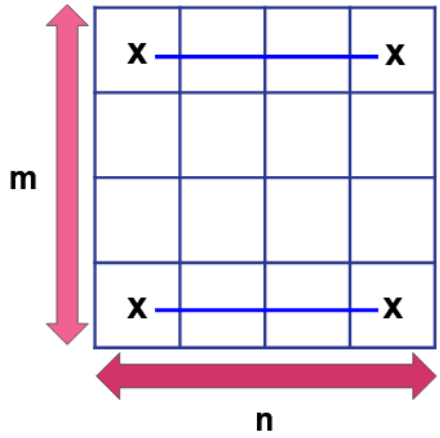


Deuxième étape, comme nous avons $3 \times 4 = 12$ cases de départ de « rectangles » dans cette grille, on peut former $12 \times 6 = 72$ rectangles.

Troisième étape, on remarque que le rectangle rouge peut être formé en partant des quatre cases qui le composent d'où le nombre total de rectangles formés sera de 72/4 = 18 dans cette grille 3x4.

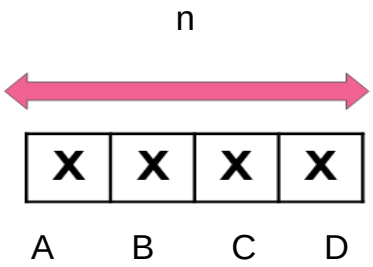
II.2 Nombre de duos autorisés

Un duo est défini lorsqu'un entier naturel (représenté par une X) apparaît deux fois sur la même ligne ou la même colonne, et si 2 duos se répètent un rectangle est formé.



Par la suite, nous avons cherché à calculer le nombre de duos que l'on peut faire dans une grille de taille donnée sans former de rectangle.

Nous avons donc cherché le nombre de duos que possède une ligne ou une colonne (cela revient au même de travailler sur une ligne ou sur une colonne). Nous avons d'abord attribué une lettre à chaque case d'une colonne.



Puis nous avons constitué toutes les combinaisons de duos possibles pour chaque lettre :

On remarque que chaque duo est représenté deux fois donc nous obtenons la formule suivante : $\frac{n(n-1)}{2} = \frac{n^2-n}{2}$

On en conclut que si le nombre de duos créés sur les lignes en remplissant la grille dépasse $\frac{n(n-1)}{2}$, on aura nécessairement formé un rectangle. (3)

Dans la suite de cet article, on retiendra que $\frac{n(n-1)}{2}$ est le nombre de duos autorisés.

II.3 Implication stratégique

On veut remplir un maximum de cases en créant un minimum de duos.

Par exemple, sur cette grille 3×5, on a disposé le chiffre 1, deux fois sur la première ligne, une fois sur la deuxième et deux fois sur la troisième :

A	B	C	D	E
1	**1**			
	1			
	1	**1**		

On se demande sur quelle ligne notre joueur a intérêt à jouer…

A	B	C	D	E
1	**1**		**1**	
	1			**1**
	1	**1**	**1**	

S'il ajoute un 1 sur la première ligne, il forme 2 nouveaux duos : AD et BD.
S'il ajoute un 1 sur la deuxième ligne, il forme 1 nouveau duo : BE.
S'il ajoute un 1 sur la troisième ligne, il forme 2 nouveaux duos : BD et CD.
On en conclut que les joueurs auront intérêt à répartir leurs coups en fonction du nombre de cases remplies sur les lignes.

II.4 Nombre de duos en pratique

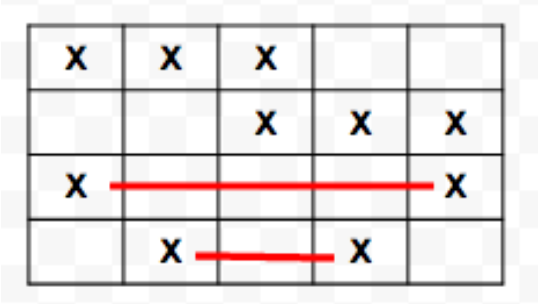
Maintenant on va tenter de calculer le nombre de duos que formera le joueur en emplissant ses cases et en suivant notre stratégie.

Imaginons que nous voulions remplir 10 cases d'une grille de dimension 4×5. Le nombre de duos autorisés sur les lignes est donc de $\frac{n(n-1)}{2}=\frac{5(5-1)}{2}$ soit 10 duos.

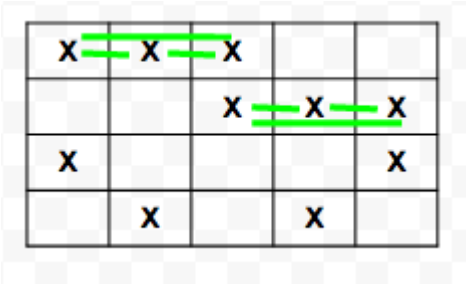
On va compter le nombre de duos formés sur les lignes en remplissant 10 cases de cette grille et en répartissant nos coups en fonction des colonnes.
Donc, naturellement comme on répartit nos coups sur 4 lignes, on commence par remplir 2 cases par ligne (= quotient de 10 par 4 noté $10\vdash4$), mais à la fin il nous en reste 2 (= reste de la division euclidienne de 10 par 4 noté $10\mathrm{mod}4$), donc on remplit une case de plus sur 2 lignes.

X	X	X		
		X	X	X
X				X
	X		X	

Pour compter les duos on commence par les 2 lignes qui contiennent 2 cases remplies, soit 1 duo. Soit un total sur ces deux lignes de 2 duos.



Ensuite on y ajoute le nombre de duos sur les 2 lignes qui contiennent 3 cases remplies, soit 3 duos. Donc un total sur ces deux autres lignes de 6 duos.



Sur cette grille on a donc 8 duos formés pour 10 duos autorisés, il était donc possible de ne pas former de rectangles, ce que l'on a fait.

Maintenant, décomposons les calculs que nous avons faits pour revenir au cas général, on appelle d le nombre de duos formés :

d = 8

Pour le trouver on avait additionné le nombre de duos sur les lignes.

d = 6+2

Que l'on avait obtenu en multipliant le nombre de ces lignes par le nombre de duos qu'elles contenaient :
d = 2x3+2x1

Le nombre de duos de chaque ligne avait été trouvé à partir du nombre de cases remplies sur celles-ci et grâce à la formule démontrée plus haut (cf. Nombre de duos autorisés) :
d = 2x3+2x1
d = 2(3(3-1)/2)+2(2(2-1)/2)

Le nombre de lignes plus ou moins remplies avait été trouvé grâce aux divisions euclidiennes :
d = 2(3(3-1)/2)+2(2(2-1)/2)
d = 10mod4(3(3-1)/2)+(4-10mod4)(2(2-1)/2)

Et le nombre de cases qu'elle contenait, avait aussi été obtenu grâce à la division euclidienne :
d = 10mod4(3(3-1)/2)+(4-10mod4)(2(2-1)/2)
d = 10mod4((10⊦4+1)((10⊦4+1)-1)/2)+(4-10mod4)((10⊦4)((10⊦4)-1)/2)

Soit, dans le cas général d'une grille m×n :
$$d = c \bmod m \times \frac{(c⊦m+1)((c⊦m+1)-1)}{2} + (m - c \bmod m) \times \frac{(c⊦m)(c⊦m-1)}{2}$$

II.5 - Le nombre d'entiers dans la grille

Donc, pour tenter de remplir la grille au maximum, on doit faire en sorte que le nombre de duos formés en remplissant un certain nombre de cases c, donné par la formule :

$$c \bmod m \times \frac{(c ⊦ m + 1)((c ⊦ m + 1) - 1)}{2} + (m - c \bmod m) \times \frac{(c ⊦ m)(c ⊦ m - 1)}{2}$$

reste inférieur (4) au nombre de duos autorisés, donné par la formule :

$$\frac{n(n-1)}{2}$$

Exemples : Prenons une grille 4x5.

1) Peut-on remplir 9 cases dans cette grille sans former de rectangle ?
D'après la formule : d = 1(2+1)(3-1)/2 + (4 - 1)(2x1/2) = 3 + 3 = 6 duos possibles et le nombre de duos maximum pour ne pas former de rectangle, est de 5×4/2 = 10 donc comme 6 < 10, c'est possible

Vérifions-le : On place deux **0** sur chaque ligne car le quotient de 9 par 4 est 2.
Puis on ajoute un **0** sur une des 4 lignes car le reste de cette division est 1.

0		**0**		
	0		**0**	
	0	**0**		**0**
0			**0**	

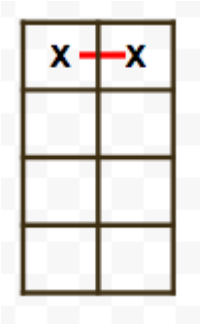
2) Peut-on remplir 11 cases dans cette grille sans former de rectangle ? (5)
Dans ce cas, d = 3x3 + 1x1 = 10 or 10 n'est pas strictement inférieur à 10 donc dans ce cas, si on place onze 0, on formera forcément un rectangle.
Toujours en appliquant la même méthode : on commence par placer deux **0** par ligne puis on complète par un **0** sur 3 des 4 lignes.

0	**0**	**0**		
	0		**0**	**0**
	0	**0**		**0**
0			**0**	

Peu importe la case vide sur la première ligne ou la dernière ligne, si on place le dernier **0**, on obtiendra un rectangle.

On a observé aussi que si un entier unique parvenait à remplir la moitié ou plus de la grille, un seul autre entier pourrait la compléter, soit 2 entiers.
Si on remplit entre le tiers et la moitié de la grille, deux autres entiers suffisent, soit 3 entiers.
Si on remplit entre le quart et le tiers de la grille, il faut trois autres entiers minimum, soit 4 entiers.

Par exemple : Prenons cette grille :

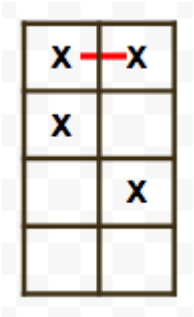


Si on veut la remplir sans former de rectangles avec un premier entier, cela signifie que sur les lignes suivantes, on a : (6)

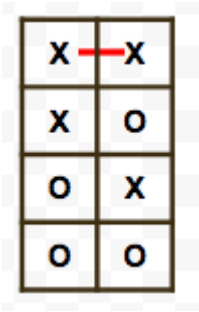
- une case remplie sur la première colonne et pas sur la deuxième;
- une case remplie sur la deuxième colonne et pas sur la première;
- aucune des cases de la première ni de la deuxième colonne remplies;

On peut bien sûr commuter ces trois lignes !

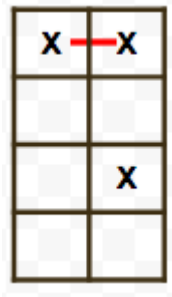
Cela donnerait une grille comme celle-ci par exemple :



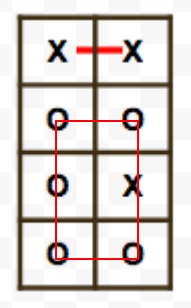
Et si on remplit les cases restantes avec un deuxième entier, on obtient une grille sans rectangle.



Par contre, si on ne remplit qu'une deuxième ligne avec le premier entier :



Il nous resterait plus de cases à remplir avec un deuxième entier et on obtiendrait forcément un rectangle



Il est donc nécessaire que le nombre total de cases divisé par le nombre d'entiers à placer reste inférieur ou égal au nombre maximal de cases pouvant être occupées par un seul entier sans former de rectangle

II.6 Formules finales

On peut en conclure qu'avec : m et n les dimensions de notre grille, k le nombre d'entiers que l'on y place et c le nombre maximal de cases pouvant être occupées par un entier sans former de rectangle, on devra respecter les deux inégalités suivantes : (7)

$$\frac{mn}{k} \leq c$$

et

$$c \; mod \; m \times \frac{(\mathrm{c} \vdash \mathrm{m}+1)((\mathrm{c} \vdash \mathrm{m}+1)-1)}{2} + (m - \mathrm{c} \bmod \mathrm{m}) \times \frac{(\mathrm{c} \vdash \mathrm{m})(\mathrm{c} \vdash \mathrm{m}-1)}{2} < \frac{n(n-1)}{2}$$

II.7 - Python

On a ici un code qui à partir de la longueur et de la largeur d'une grille retourne cette grille remplie sans rectangle. (8)

l=int(input("Rentrez le nombre de lignes:")) L=int(input("Rentrez le nombre de colonnes: ")) c=0	*Dans ce programme, on considère que l'utilisateur doit fournir la longueur et la largeur de la grille qu'il compte remplir pour se voir retourner le nombre de cases maximal qu'il peut occuper sans former de rectangles. Donc par défaut on met le nombre de cases remplies à zéro pour le faire augmenter avec des tests successifs ensuite.*
def duos(l,c): duos=(l-c%l)*(c//l)*(c//l-1)/2+(c%l)*(c//l)*(c//l+1)/2 return duos	*Pour un nombre de cases remplies fourni et une largeur donnés la fonction duos retourne le nombre de duos qui ont été formés.*

```
while duos(l,c)<L*(L-1)/2:
        duos(l,c)
        c=c+1

c=c-1
```

Donc dans cette boucle on incrémente le nombre de cases que l'on remplit jusqu'à ce qu'il n'y ait plus de duos formés (obtenus grâce à l'appel à duos) que de duos autorisés (que l'on obtient grâce à la formule du L(L-1)/ 2)

Comme dans la boucle d'avant, on ajoutait 1 avant de tester que c'était toujours bon au niveau des duos, il faut retirer 1 ensuite, sinon le nombre de cases à remplir est trop grand.

```
dui=[]
for i in range(L):
        for j in range(L):
                if j>i:
                        dui.append((i,j))
```

On va attribuer un numéro à chaque ligne et donc écrire tous les duos qui pourront être formés avec ces lignes, comme 01, 02, 12, 31, etc… Pour cela on utilise deux variables, l'une pour le premier numéro de ligne et l'autre pour le second.
Pour éviter d'avoir des doubles, comme 01 et 10, qui désignent le même duo, on va demander au programme de ne conserver que les doubles ayant une deuxième valeur supérieure à la première. On aurait pu faire l'inverse, et au lieu de ne garder que 01, on aurait conservé 10.
Et ensuite on demande au code d'enregistrer les duos valides dans une liste que l'on va appeler dui.

```
tableau=[[False for i in range(L)] for j in range(l)]
```

Ici le code forme une liste de listes, c'est-à-dire que c'est un peu comme un tableau ou une grille, et c'est cette fameuse grille que l'on souhaite remplir. Ici on va considérer que False signifie que la case est vide et True qu'elle est remplie. On démarre donc avec toutes les cases vides.

```
j=0
for i in range(c%l):
        for j in range(j,j+c//l+1):
                if j>L-1:
                        j=0
                tableau[i][j]=True

for i in range(c%l,l):
        for j in range(j,j+c//l):
           if j>L-1:
              j=0

        tableau[i][j]=True          (9)
```

On a remarqué que l'on avait intérêt à bien répartir les cases en fonction des lignes. On va donc respecter ceci pendant le premier remplissage : c'est-à-dire qu'il y a c MOD l lignes qui sont plus remplies que les autres, elles contiennent c//l+1 cases remplies, donc on s'occupe dans un premier temps de remplir ces lignes. Dans un deuxième temps on s'occupe des lignes restantes l-c MOD l

En effet, i correspond à l'indice de la ligne que l'on est en train de remplir

Cependant, j correspond à l'indice de la colonne que l'on est en train de remplir, donc comme on remplit colonne par colonne puis on change de ligne, le programme recommence en remplissant la colonne.

Mais comme à chaque fois on reprend de la colonne où l'on s'est arrêté au niveau de la ligne précédente au bout d'un moment on arrive à la fin du tableau : et à ce moment-là il faut remettre j à zéro.

```
for i in range(l):
  for xa in range(L-1):
    for xb in range(xa+1,L):
      if tableau[i][xb]==True and tableau[i][xa]==True:
            if ((xa,xb) in dui)==True:
              dui.remove((xa,xb))
            else:
                tableau[i][xb]=False
                xb=xb+1
                while xb in range(xb,L) and
tableau[i][xb]==True:
                    xb=xb+1
                tableau[i][xb]=True      (10)
print(tableau)
for i in range(l):
      print("")
      for j in range(L):
        if tableau[i][j]==True:
          print("X",end='')
        else:
          print("O",end='')
```

Ici c'est la partie qui permet de faire passer la case de ligne i et de colonne j de vide à remplir.

On commence par chercher des duos. Afin de les constituer, on reprend la liste de duos autorisés que l'on a créée précédemment, puis on prend deux entiers : xa et xb.

Si les deux cases regardées sont remplies et forment un duo :

Si le duo qu'elles forment se situe dans la liste des duos autorisés:
On considère que ce duo est fait et on le retire de la liste des duos autorisés.
Par contre, si le duo a déjà été retiré de la liste, c'est-à-dire qu'il a déjà été réalisé,
alors on supprime l'une des cases qui forment ce duo dans le but de l'enlever.

En revanche on décale la case d'un emplacement jusqu'à trouver un endroit où il y aurait une case vide.

Puis on le retranscrit en remplaçant les cases vides, avec des False par des O et celles avec des True par des X.

Ce programme permet donc de placer un maximum de croix dans une grille de dimensions données sans former de rectangles avec ces croix, comme ici :

```
Rentrez le nombre de lignes: 7
Rentrez le nombre de colonnes: 3

XXO
OXX
XOX
XOO
XOO
XOO
XOO
```

```
Rentrez le nombre de lignes: 4
Rentrez le nombre de colonnes: 5

XXXOO
OOXXX
XOOOX
XOOXO
```

III - Conclusion

On a trouvé une formule (11) qui à partir des dimensions m et n d'une grille parvient à donner le nombre k d'entiers minimal permettant de la remplir sans former de rectangles. (12)

On a aussi créé un code Python (13) capable de remplir cette grille avec un maximum de croix sans former de rectangle sauf que ce remplissage n'est pas optimum car il ne permet pas de minimiser le nombre d'entiers k permettant de la remplir sans former de rectangles.

Notes d'édition

(1) La formulation n'est pas claire : Une fois qu'une case est fixée comme coin d'origine, on compte ici le nombre de rectangles que l'on peut former dont cette case est un coin.
(2) Cette partie sert à compter le nombre de cases que l'on peut choisir comme coin de rectangle d'origine.
(3) L'apparition de cette formule est un peu rapide. Cette dernière aurait pu être justifiée comme les auteurs l'ont déjà fait auparavant pour dénombrer les rectangles. De plus, c'est une condition nécessaire mais pas suffisante.
(4) Il s'agit d'une inégalité large : inférieur ou égal. Comme la précédente, cette inégalité ne donne qu'une condition nécessaire mais présentée comme suffisante.
Pour obtenir cette inégalité, les auteurs raisonnent sur les lignes. Il est intéressant de remarquer qu'une inégalité du même type peut être obtenue en raisonnant sur les colonnes : on remplace les 'm' par des 'n' et les 'n' par des 'm' dans cette inégalité.
(5) Dans ce deuxième cas, les auteurs remarquent qu'ils ne parviennent pas à mettre onze "0". En utilisant la formule pour les colonnes mentionnée précédemment, on obtient d=7 pour un maximum de duos autorisés de 6. On démontre donc que la méthode décrite par les auteurs ne permet pas de remplir cette grille. Il reste à montrer que cette méthode est bien optimale, ce qui peut être fait en considérant plusieurs cas sur cet exemple précis.
(6) Précisément, chaque ligne peut être construite d'une des trois façons décrites ici mais rien n'empêche que deux lignes soient construites de la même façon. Cependant, ceci ne change rien à la véracité du raisonnement des auteurs dans ce contexte un peu plus général.
(7) Comme précisé auparavant, on peut rajouter aussi l'inégalité pour les colonnes.
(8) Ce code Python souffre de quelques défauts. Le principal étant que pour certaines données valides pour l et L (par exemple l=8 et L=8), le programme va planter. Lorsqu'il ne plante pas, il semble cependant fonctionner correctement et faire ce que l'on attend de lui. À souligner : les relecteurs apprécient beaucoup les explications et commentaires qui accompagnent le programme.
(9) Cette partie du programme sert à mettre le bon nombre de "X" dans chaque ligne du tableau. Les relecteurs déplorent que les auteurs n'aient pas expliqué l'intérêt de remplir ainsi le tableau et pas d'une façon naïve au départ avec des "X" en début de ligne par exemple.
(10) L'erreur qui peut se produire avec ce programme est ici. En effet, la variable xb peut tout à fait être plus grande que la taille L du tableau. Dans ce cas, la case ciblée du tableau n'existe pas.
(11) Les auteurs résolvent complètement le problème dans un cas plus simple que le problème de départ, celui où l'on n'utilise qu'un entier quitte à laisser des cases vides. Cette approche est utilisée ensuite pour déduire une borne inférieure correcte sur le nombre d'entiers nécessaires. Les auteurs sont ensuite un peu confus sur leurs conclusions et semblent penser que leur stratégie est optimale alors que ce n'est pas démontré et pensent donc que leur borne est optimale alors que ce n'est pas clairement démontré.
(12) Précisément, les auteurs donnent deux inégalités qu'il faut encore résoudre pour chaque cas. La première permet de déterminer le nombre de cases maximal pouvant contenir le même entier et la seconde permet de trouver un nombre k tel que si l'on utilise strictement moins de k entiers pour remplir la grille, il y aura forcément un rectangle dont les quatre coins contiendront le même entier. Il n'est cependant pas clair que l'on puisse effectivement remplir ce tableau avec exactement k entiers. Une amélioration a été évoquée par les relecteurs en rajoutant une autre inégalité. Cependant, il reste encore des obstructions géométriques à lever pour avoir l'optimalité.
(13) Ce code ne fonctionne malheureusement pas toujours.