IFOSUP

Institut de Formation Supérieure
Ville de Wavre

Mathématiques

appliquées à l'informatique

G. Barmarin

2024-2025

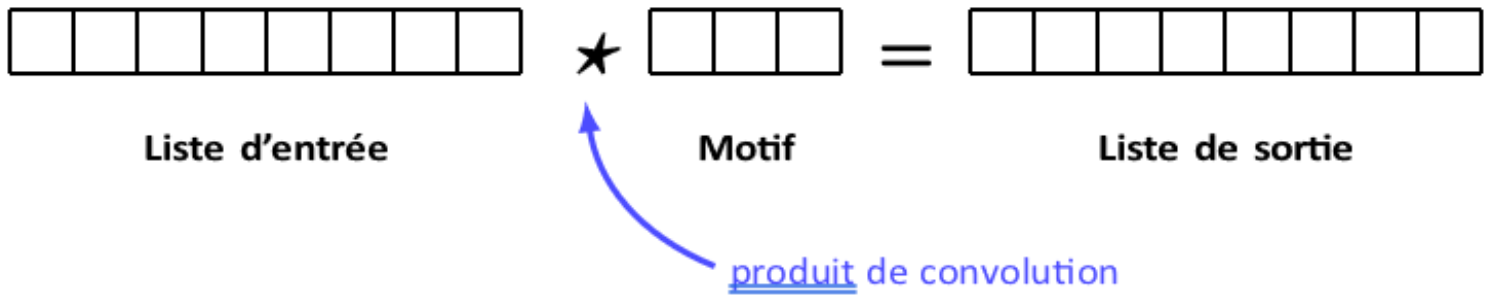
Convolution et traitement d’images avec SciPy

Convolution

Convolution à une dimension

Idée

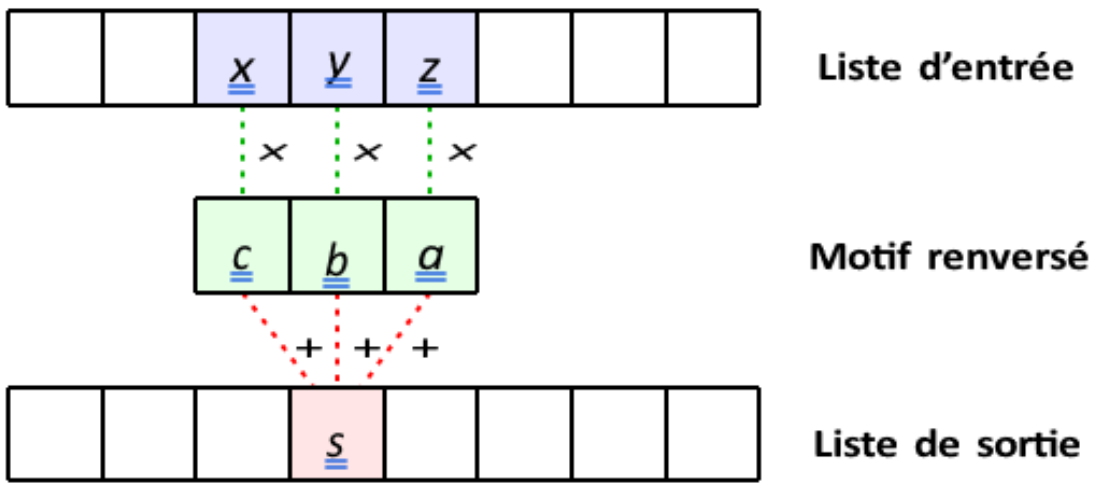
La convolution est une opération qui à partir d'un tableau de nombres et d'un motif produit un nouveau tableau de nombres.



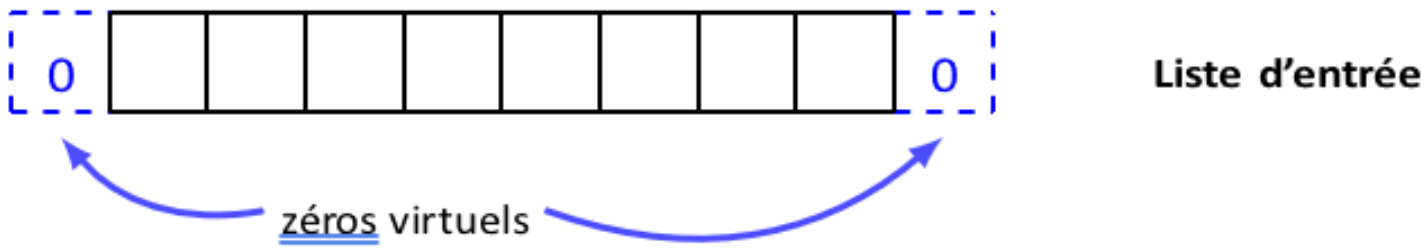
Le calcul de la liste de sortie se fait par des opérations très simples.

- On commence par renverser le motif.

- On calcule la liste de sortie terme par terme :
  - — on centre le motif renversé sous la liste d'entrée, à la position à calculer,
  - — on multiplie terme à terme les éléments de la liste d'entrée et ceux du motif,
  - — la somme de tous ces produits est le terme de la liste de sortie.



Pour le calcul des coefficients sur les bords, on rajoute des zéros virtuels à gauche et à droite de la liste d'entrée.

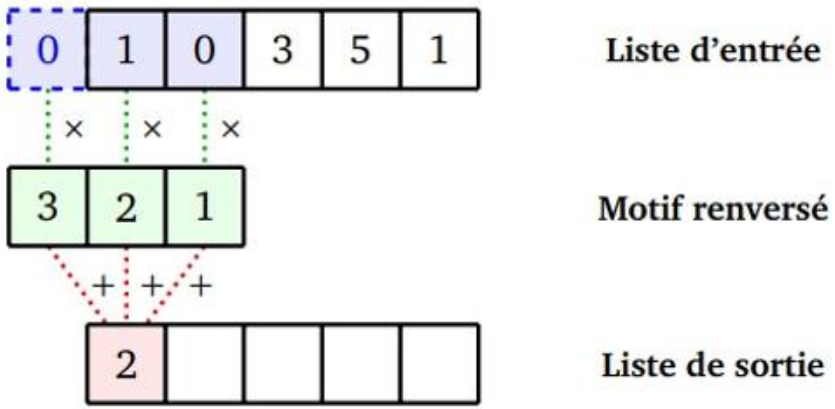


La liste de sortie est de même taille que celle de la liste d'entrée.

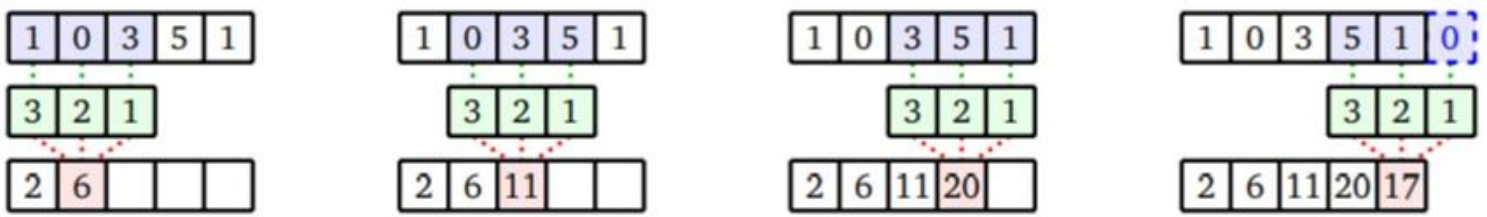
Exemple.

Calculons la convolution [1, 0, 3, 5, 1]⋆[1, 2, 3].

On commence par calculer le coefficient le plus à gauche (après ajout d'un zéro virtuel à la liste d'entrée).



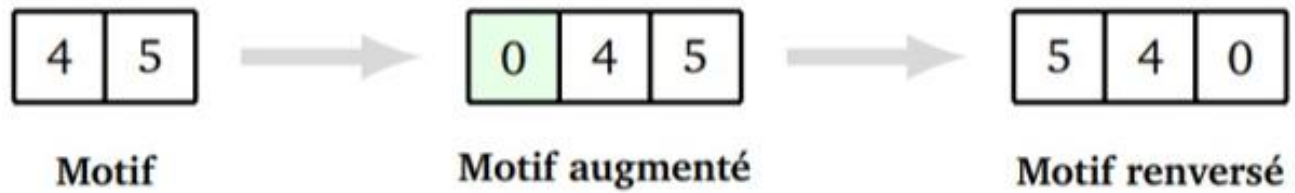
On continue en calculant les coefficients un par un.



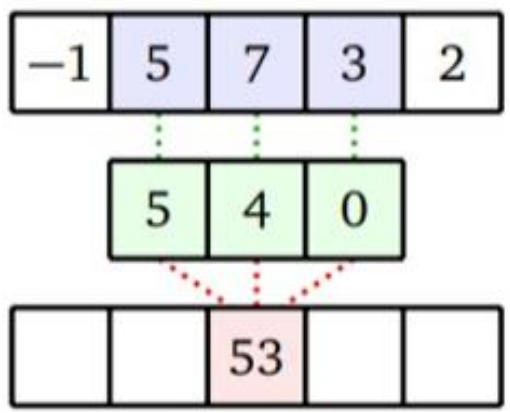
Ainsi [1, 0, 3, 5, 1]⋆[1, 2, 3]=[2, 6, 11, 20, 17].

Si le motif est de longueur paire alors on ajoute un 0 au début du motif pour le calcul.

Ainsi : $[-1, 5, 7, 3, 2] * [4, 5] = [-1, 5, 7, 3, 2] * [0, 4, 5]$



Lorsque l'on renverse le motif, le zéro ajouté se retrouve à droite !



Après calcul on obtient $[-1, 5, 7, 3, 2] * [4, 5] = [-4, 15, 53, 47, 23]$.

Exemples:

Moyenne mobile.
La convolution par le motif [, ,] ou [1,1,1] correspond à effectuer une moyenne mobile sur trois termes.
Plus généralement, le motif de longueur n : 1/n[1, 1, 1, . . . , 1] correspond à une moyenne mobile sur n termes.
Cela permet de « lisser » des données et de supprimer du « bruit ».

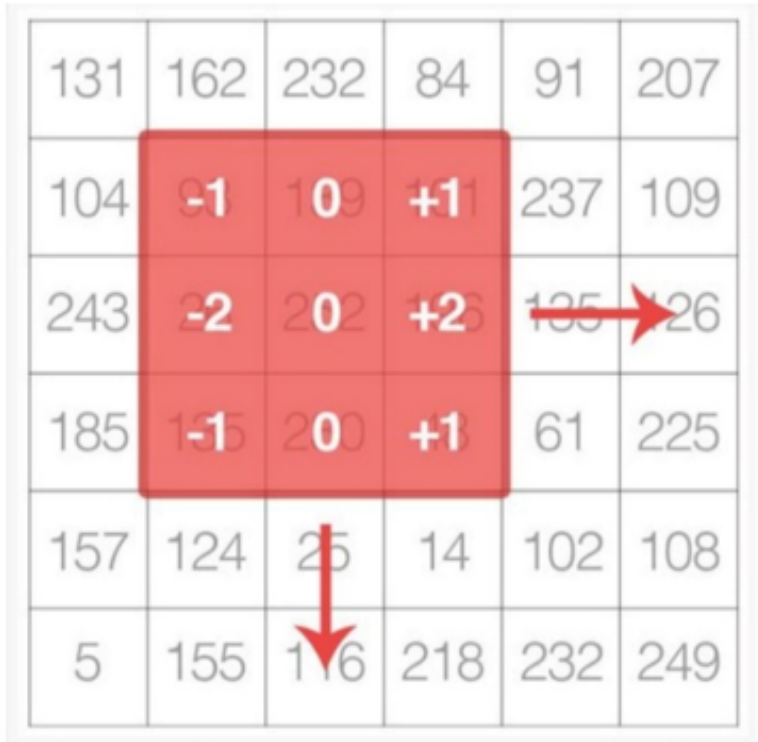
Homothétie.
La convolution par le motif [k] (de longueur 1 et qui pourrait encore s'écrire [0, k, 0]) multiplie
tous les termes de la liste d'entrée par un facteur k.
Exemple : [1, 2, 3, 4, 5, 6, 7, 8, 9] ⋆ [2] = [2, 4, 6, 8, 10, 12, 14, 16, 18].

Translation.
La convolution par le motif [0,0,1] décale la liste d'un cran vers la droite (avec ajout d'un
zéro en début, le dernier terme disparaissant).
Exemple : [1, 2, 3, 4, 5, 6, 7, 8, 9] ⋆ [0, 0, 1] = [0, 1, 2, 3, 4, 5, 6, 7, 8].

Dérivée.
La convolution par le motif [0, 1,−1] (qui s'écrit aussi [1,−1]) correspond au calcul de la dérivée
de la liste d'entrée vu comme une fonction n → f (n).
Exemple : [16, 9, 4, 1, 0, 1, 4, 9, 16] ⋆ [0, 1,−1] = [16,−7,−5,−3,−1, 1, 3, 5, 7].
Le calcul correspond à f (n+1)− f (n), une analogie discrète du taux d'accroissement f (x+h)−f (x)/h avec h = 1.

Convolution : deux dimensions

La convolution en deux dimensions est une opération qui à partir d'une matrice d'entrée notée A (en arrière-plan gris) et d'une matrice d'un motif noté M (en rouge), associe une matrice de sortie A⋆ M.



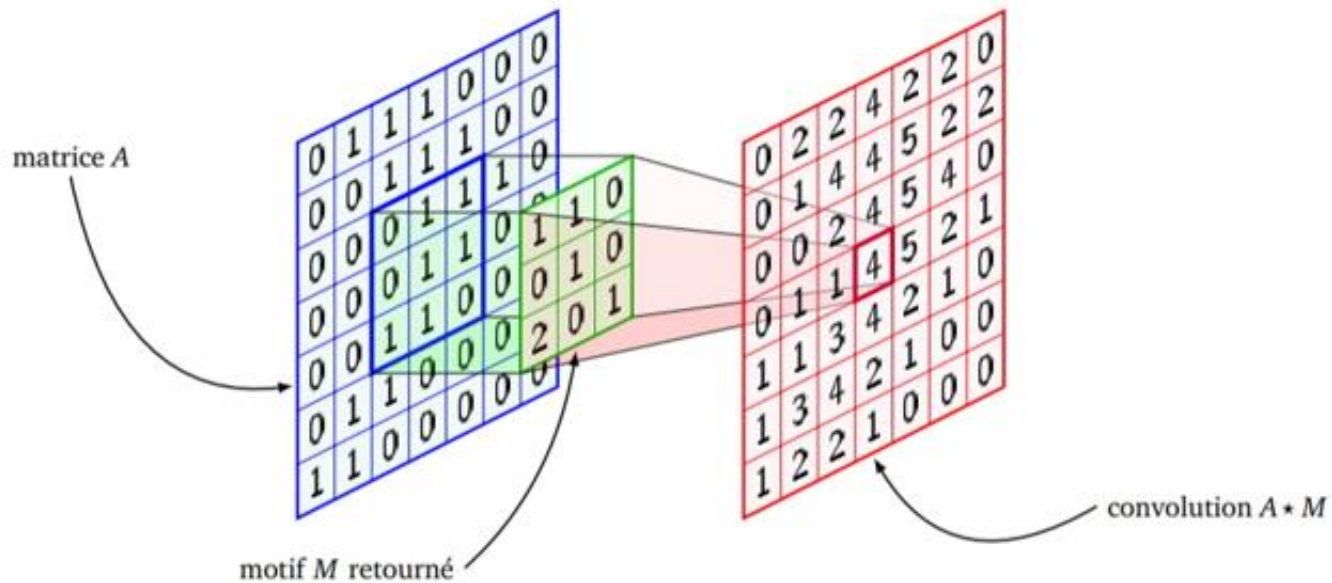
Tout d'abord, il faut retourner la matrice M :

$$\begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix} \quad \begin{pmatrix} m_{33} & m_{32} & m_{31} \\ m_{23} & m_{22} & m_{21} \\ m_{13} & m_{12} & m_{11} \end{pmatrix}$$

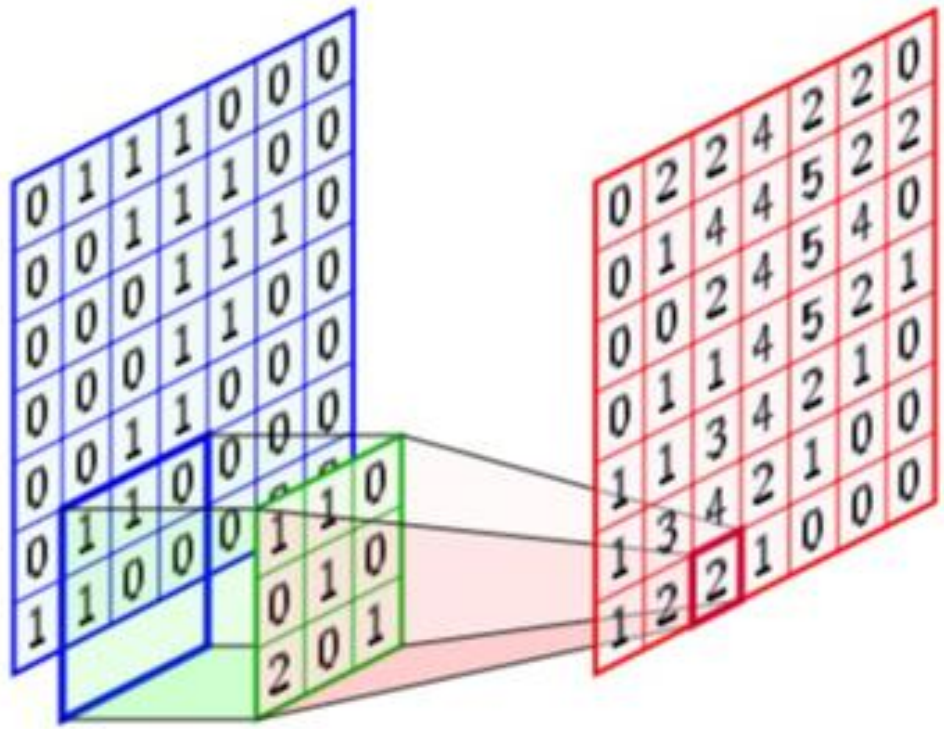
Motif de taille 3×2 **Motif retourné**

Le calcul de A⋆ M s'effectue coefficient par coefficient :

- on centre le motif retourné sur la position du coefficient à calculer,
- on multiplie chaque coefficient de A par le coefficient du motif retourné en face (quitte à ajouter des zéros virtuels sur les bords de A),
- la somme de ces produits donne un coefficient de A⋆ M.



Voici un autre schéma pour le calcul d'un autre coefficient. Pour ce calcul, on rajoute des zéros virtuels autour de la matrice A.



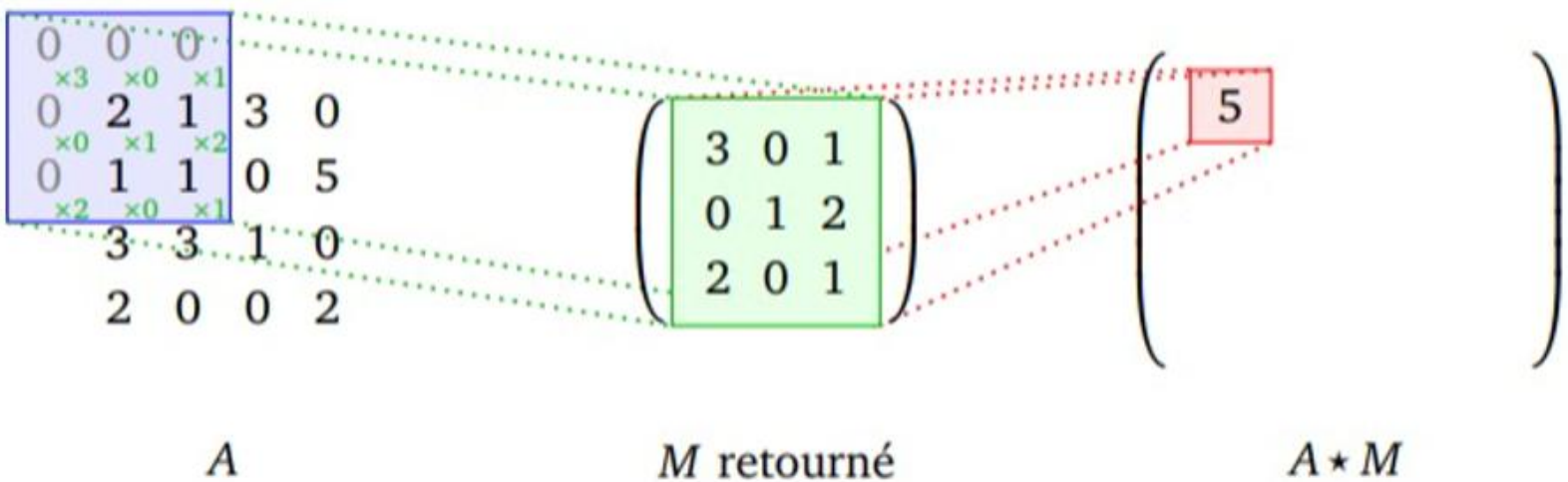
Exemple. Calculons la convolution $A \star M$ définie par :

$$\begin{pmatrix} 2 & 1 & 3 & 0 \\ 1 & 1 & 0 & 5 \\ 3 & 3 & 1 & 0 \\ 2 & 0 & 0 & 2 \end{pmatrix} \star \begin{pmatrix} 1 & 0 & 2 \\ 2 & 1 & 0 \\ 1 & 0 & 3 \end{pmatrix}.$$

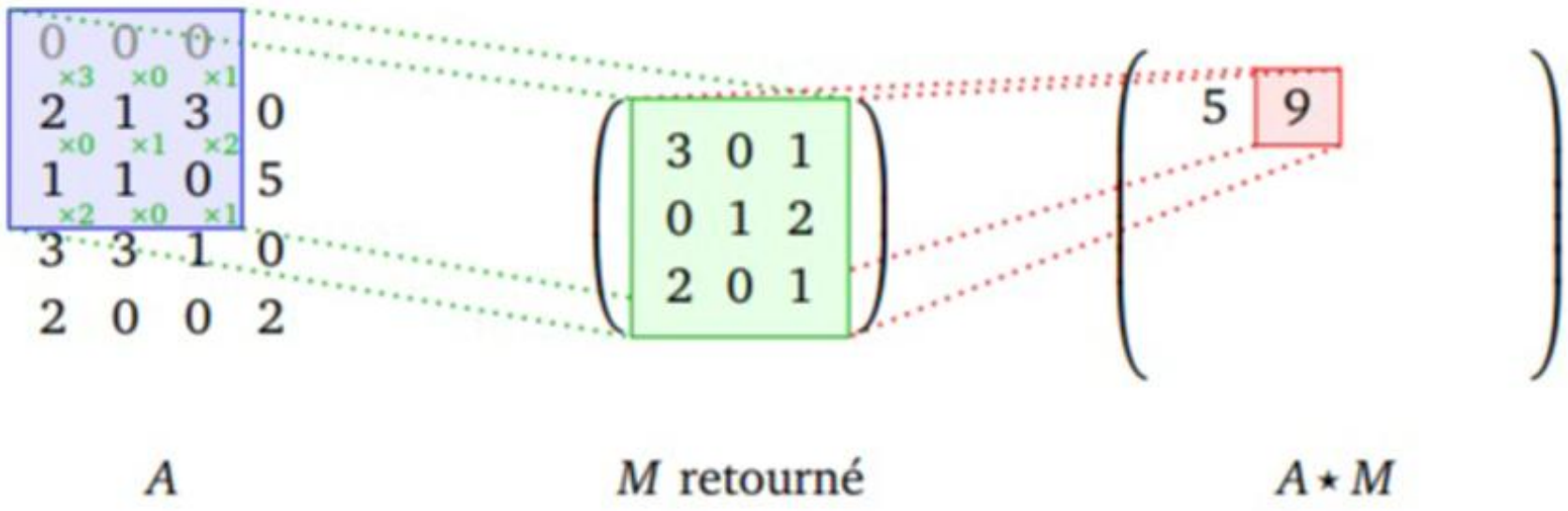
On commence par retourner M. Pour calculer le premier coefficient de $A \star M$, on centre le motif sur le premier coefficient de A, puis on rajoute des zéros virtuels à gauche et en haut. Ensuite on calcule les produits des coefficients de la matrice M retournée avec les coefficients de A correspondants, et on les additionne :

$0 \times 3 + 0 \times 0 + 0 \times 1 + 0 \times 0 + 2 \times 1 + 1 \times 2 + 0 \times 2 + 1 \times 0 + 1 \times 1.$

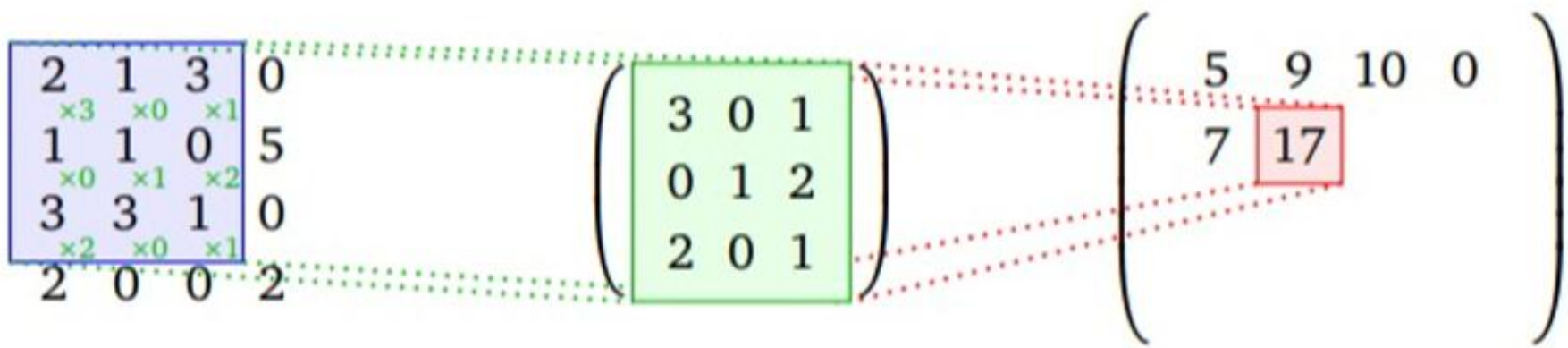
Cette somme vaut 5, c'est le premier coefficient de $A \star M$.



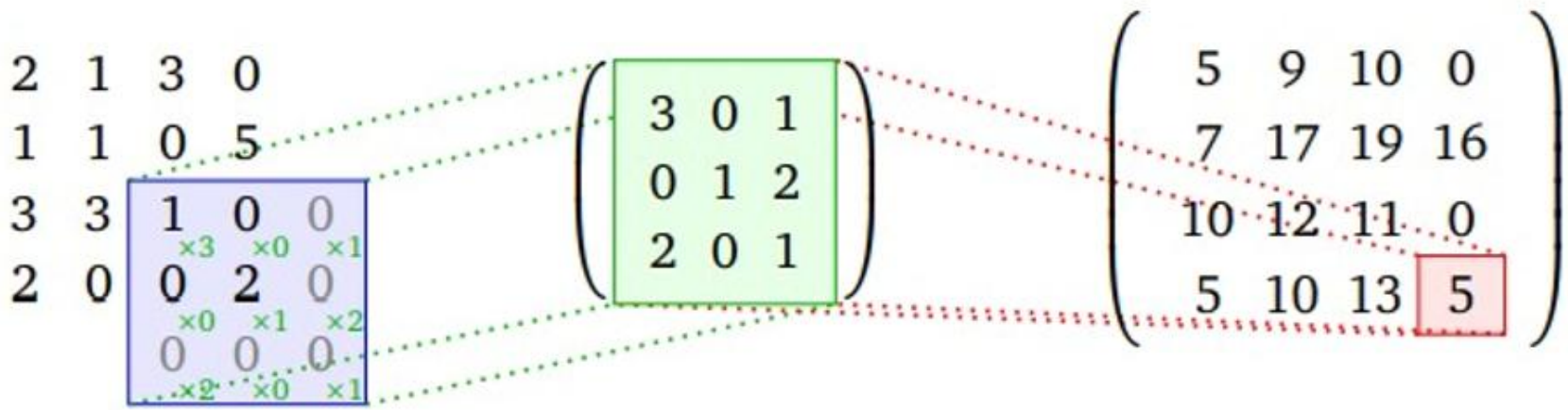
On continue avec le second coefficient.



Et ainsi de suite :



Jusqu'à calculer entièrement la matrice A⋆ M.

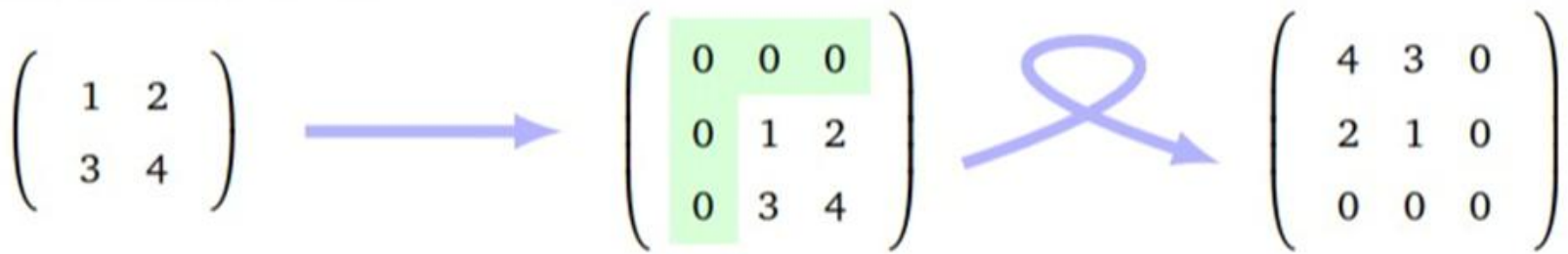


Remarque.

• Il ne faut pas confondre le fait de retourner la matrice avec une opération différente qui s'appelle la transposition.

• Dans la plupart de nos situations, le motif sera une matrice de taille 3 × 3. Par contre la matrice A pourra être de très grande taille (par exemple une matrice 600 × 400, codant une image).

• Cependant, si la matrice du motif possède une dimension paire on rajoute des zéros virtuels à gauche ou en haut (avant de la retourner).



Filtrage des images et convolution

La notion de filtre est la même que celle utilisée en physique pour le traitement du signal.
Un signal (électrique, électromagnétique, image,...) peut être représenté par son spectre fréquentiel dont il est possible, grâce à un filtre, de supprimer certaines fréquences. Par exemple, un filtre passe-bas atténue les composantes du spectre dont les valeurs sont supérieures à une valeur seuil. La fonction de transfert associée au filtre correspond à une opération mathématique réalisée sur le signal. De la même façon, filtrer une image consiste à lui appliquer une transformation mathématique qui modifie la valeur des pixels.

Comment filtrer une image ?

De nombreuses transformations d'image se font, non pas pixel par pixel, mais en prenant en compte les pixels voisins du pixel à modifier.
La convolution discrète du tableau T représentant une image (ou de chaque matrice représentant un canal de couleur) en général très grand avec un tableau M (on parle parfois de masque de convolution ou de filtre), en général assez petit, carré d'ordre impair va donner un résultat correspondant à l'image de départ filtrée.

La convolution est donc un outil puissant dans le domaine du traitement des images.

Il faut gérer les « bords » de T (image de départ), là où le masque « déborde ».

En général on souhaite conserver la taille de T pour le résultat, il faut donc adapter le calcul, soit en définissant des masques partiels, soit en bordant T en lui ajoutant des valeurs tout autour.

Dans les applications pratiques, cela n'a pas une grande importance car M est beaucoup plus petit que T et donc les anomalies sur les bords sont peu visibles.

Filtrage linéaire et filtrage non linéaire

On peut distinguer filtrage linéaire et filtrage non linéaire :

- Le filtre local est dit linéaire si la valeur du nouveau pixel est une combinaison linéaire des valeurs des pixels du voisinage. D'une manière générale, l'image résultante est floutée, voire altérée puisque les contours sont dégradés.

- Si le filtre ne peut pas être exprimé par une combinaison linéaire, il est appelé « non-linéaire ». Les filtres non-linéaires sont plus complexes à mettre en œuvre que les filtres linéaires. Cependant les résultats obtenus avec les filtres non-linéaires sont très souvent de meilleure qualité que ceux obtenus par les filtres linéaires.

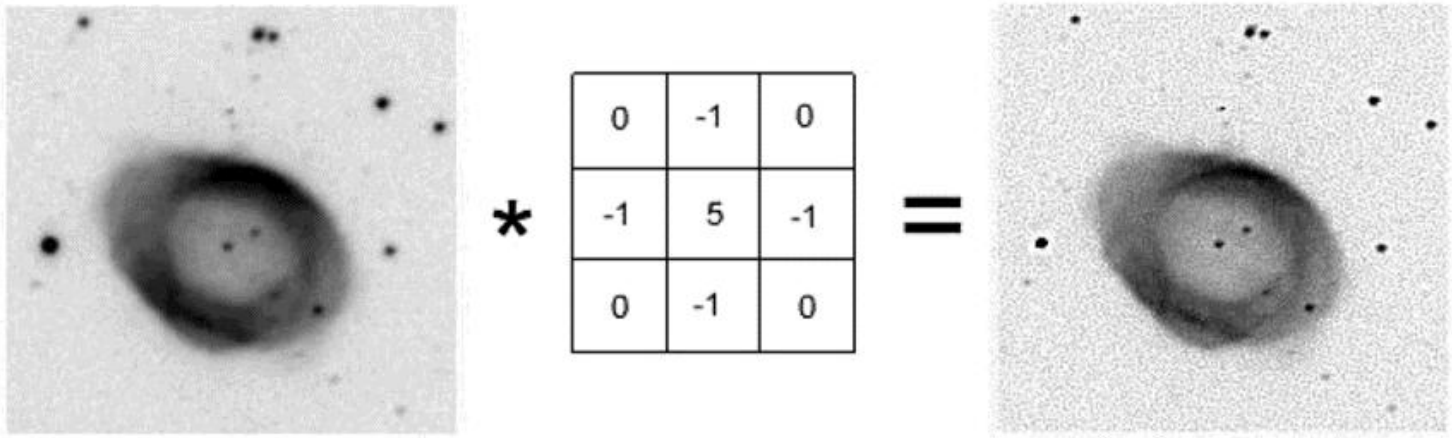
Fréquence spatiale

Dans une image les détails se répètent fréquemment sur un petit nombre de pixels, on dit qu'ils ont une fréquence spatiale élevée : c'est le cas pour les bords et les contours dans une image.

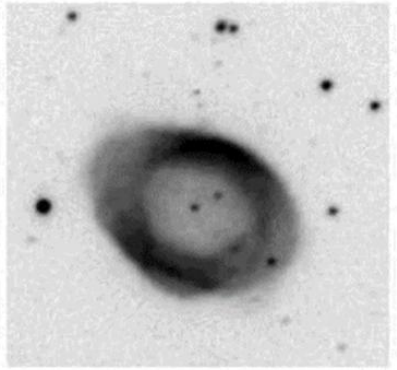
Au contraire, les fréquences basses correspondent à des variations qui se répètent peu car, diluées sur de grandes parties de l'image, par exemple des variations de fond de ciel.

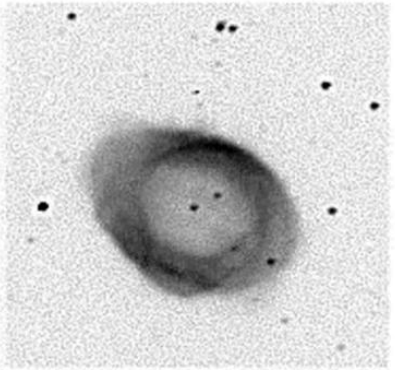
La plupart des filtres agissent sélectivement sur ces fréquences pour les sélectionner, en vue de les amplifier ou de les réduire, on parle alors de filtrage passe-bas ou passe-haut

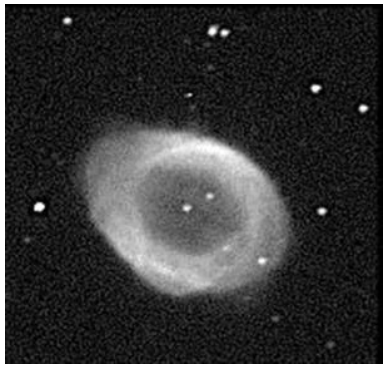
Un filtre « passe haut » favorise les hautes fréquences spatiales, comme les détails, et de ce fait, il améliore le contraste.
Un filtre « passe haut » est caractérisé par un noyau comportant des valeurs négatives autour du pixel central, comme dans l'exemple ci-dessous :



Dans cette image de ciel profond, le filtre passe-haut a été utilisé pour améliorer les détails de la nébuleuse
planétaire M 57, mais un examen attentif montre des effets secondaires au traitement :

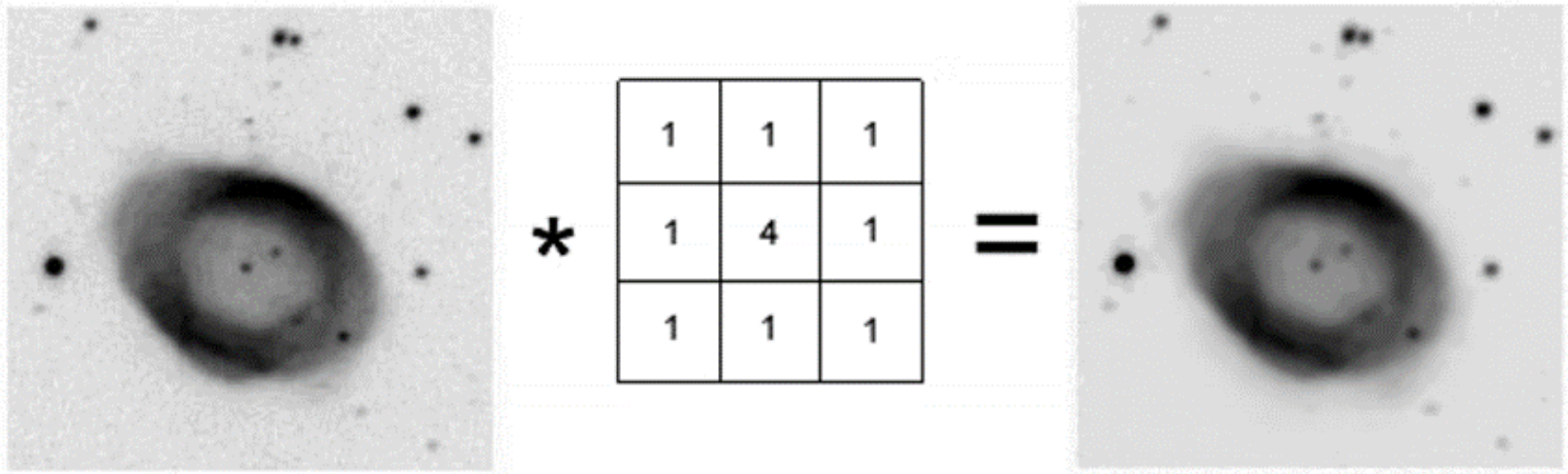






- Augmentation du bruit : particulièrement dans les images avec un rapport Signal/Bruit faible, le filtre augmente le bruit granuleux dans l'image.
- Effet couronne : L'effet du filtre sur les objets stellaires est bien sûr positif, et augmente le contraste et la différence par rapport au fond de ciel mais aussi des effets secondaires comme l'apparition de couronnes noires autour des étoiles. La cause de cet anneau noir est localisée dans le noyau du « passe haut » : les valeurs négatives créent une sorte de dépression autour de l'étoile qui peut dégénérer en un anneau noir plus sombre que le fond de ciel.
- Effet de bord : il est possible que sur les bords de l'image apparaisse un cadre. Mais cet effet est souvent négligeable et peut s'éliminer en tronquant les bords de l'image.

Les filtres « passe bas » agissent en sens inverse des filtres « passe haut » et le résultat est, un adoucissement des détails, ainsi qu'une réduction du bruit granuleux.



Filtre de flou

Il s'agit ici d'un filtre de moyenne (filtre linéaire).
C'est un cas particulier de filtre de convolution « passe-bas », qui remplace chaque pixel par la moyenne des valeurs des pixels adjacents et du pixel central. Ce lissage va réduire les fortes variations de niveaux, il atténue donc les hautes fréquences en agissant comme un filtre passe-bas. Il doit être normalisé à un.
D'une manière générale, si on a un filtre de taille d, tous les coefficients du filtre ont comme valeur $1/d^2$; Plus d est grand, plus le lissage sera important, et plus l'image filtrée perdra les détails de l'image originale.
On obtient donc une image floue par application de ce motif :

$$M = \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}.$$

En effet, la convolution par cette matrice M remplace la couleur d'un pixel par la moyenne de la couleur de ses 9 pixels voisins. Outre l'aspect esthétique, il y a un intérêt fondamental : le flou réduit le bruit. Par exemple, si un pixel est mauvais (par exemple à cause d'une erreur de l'appareil photo), le flou élimine cette erreur.

Une variante est le flou gaussien défini par la matrice :

$$M = \frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}.$$

Il existe aussi des flous définis par des matrices M de taille 5 × 5, qui tiennent compte des 25 pixels voisins.

Filtre médian (non-linéaire)

La technique du filtre médian est largement utilisée en traitement d'images numériques, car il permet de réduire le bruit tout en conservant les contours de l'image assez nets. L'idée principale du filtre médian est de remplacer chaque pixel par la valeur médiane des pixels de son voisinage. Le résultat obtenu avec ce filtre est l'élimination des pixels isolés, en évitant de créer un flou trop important dans l'image.

Filtre de Piqué

C’est en quelque sorte le contraire du flou !

Le motif est :

$$M = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{pmatrix}.$$

Détection de contours

Filtre gradient

On va faire l'approximation qu'une image numérique correspond à l'image d'une fonction $f : R^2 \dashrightarrow R$ de sorte que f (x, y) représente la valeur du pixel au point de coordonnées (x, y).
La notion de contour est intimement liée à celle des variations entre un pixel et ses voisins.
Un contour provoque donc une augmentation locale du gradient.
Les méthodes dérivatives reposent sur le constat que les contours d'image se traduisent par de fortes transitions au sein de l'image, alors que les parties homogènes de l'image suivent une faible variation qui peut être annulée par dérivation.
On va donc chercher les points (x, y) où le gradient n'est pas « trop petit » en calculant sa norme.

- Détection d'un contour vertical : le gradient de luminance des pixels est alors horizontal.
- Détection d'un contour horizontal : le gradient de luminance des pixels est alors vertical.

On cherche ici à traduire par une matrice l'expression mathématique du gradient d'une fonction I(x,y), fonction de luminance du pixel en niveaux de gris.
Sa différentielle totale s'écrit :

$$dI = \frac{\partial I}{\partial x}dx + \frac{\partial I}{\partial y}dy$$

$\frac{\partial I}{\partial x}$ est le gradient de I selon x, $\frac{\partial I}{\partial y}$ est le gradient de I selon y.

L'estimation des deux dérivées partielles de la différentielle précédente peut se faire par une approche aux différences finies à l'ordre 1 :

Taylor :

$$f(x + \Delta x) = f(x) + \frac{df(x)}{dx}\Delta x + \frac{1}{2!}\frac{d^2 f(x)}{dx^2}(\Delta x)^2.$$

soit à l'ordre 1 :

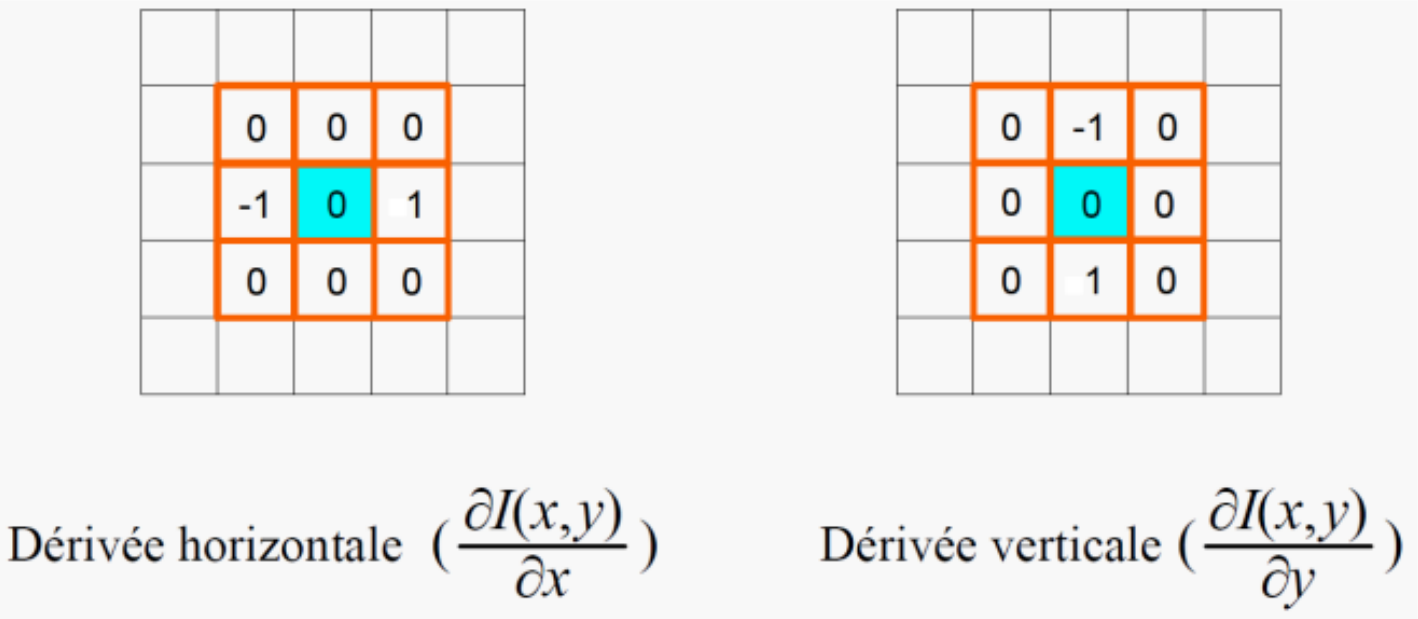
$$\frac{df(x)}{dx} = \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

que l'on nommera différence finie AVANT d'ordre 1.

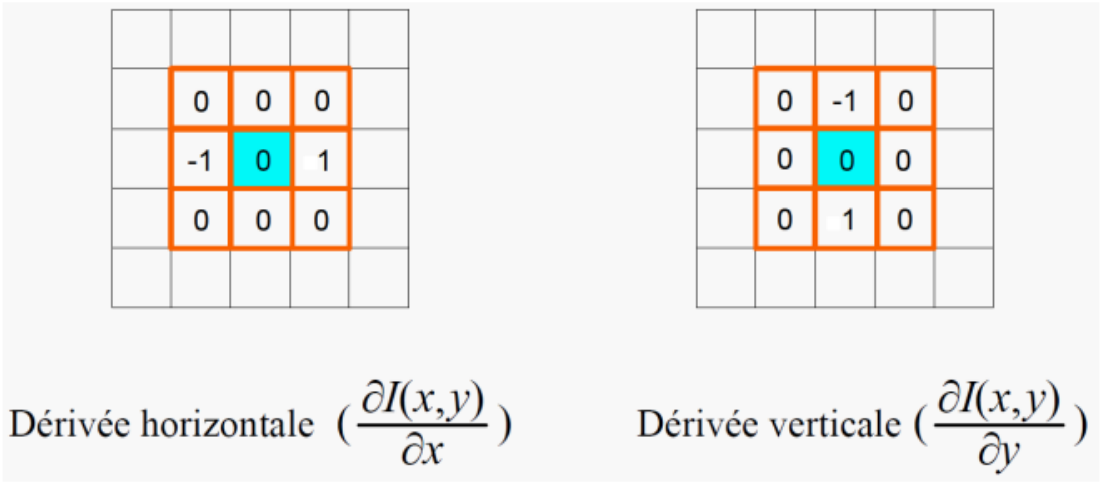
Il est aisé d'établir les deux autres relations :

- Différence finie arrière à l'ordre 1 : $df(x)/dx = (f(x) - f(x + \Delta x)) / \Delta x$
- Différence finie centrée à l'ordre 1 : $df(x)/dx = (f(x + \Delta x) - f(x - \Delta x)) / \Delta x$

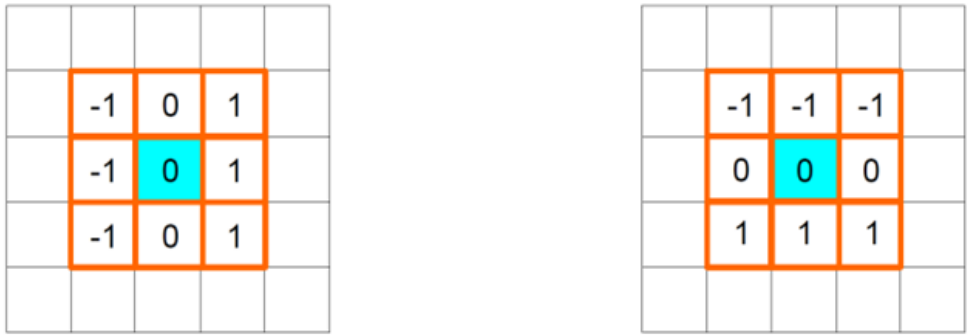
Soit au final la matrice M suivante :



De ces masques de départ



On peut en déduire les Filtre de Prewitt :



Le masque de Prewitt horizontal et vertical est calculé sur 9 points, ce filtre effectue une moyenne locale sur 3 points en même temps que la dérivation. Il doit être normalisé par un facteur de 1/3 à placer devant le filtre
Notons bien que le masque de Prewitt horizontal détecte les contours verticaux !

On peut imaginer ainsi différents filtres comme ceux qui suivent:

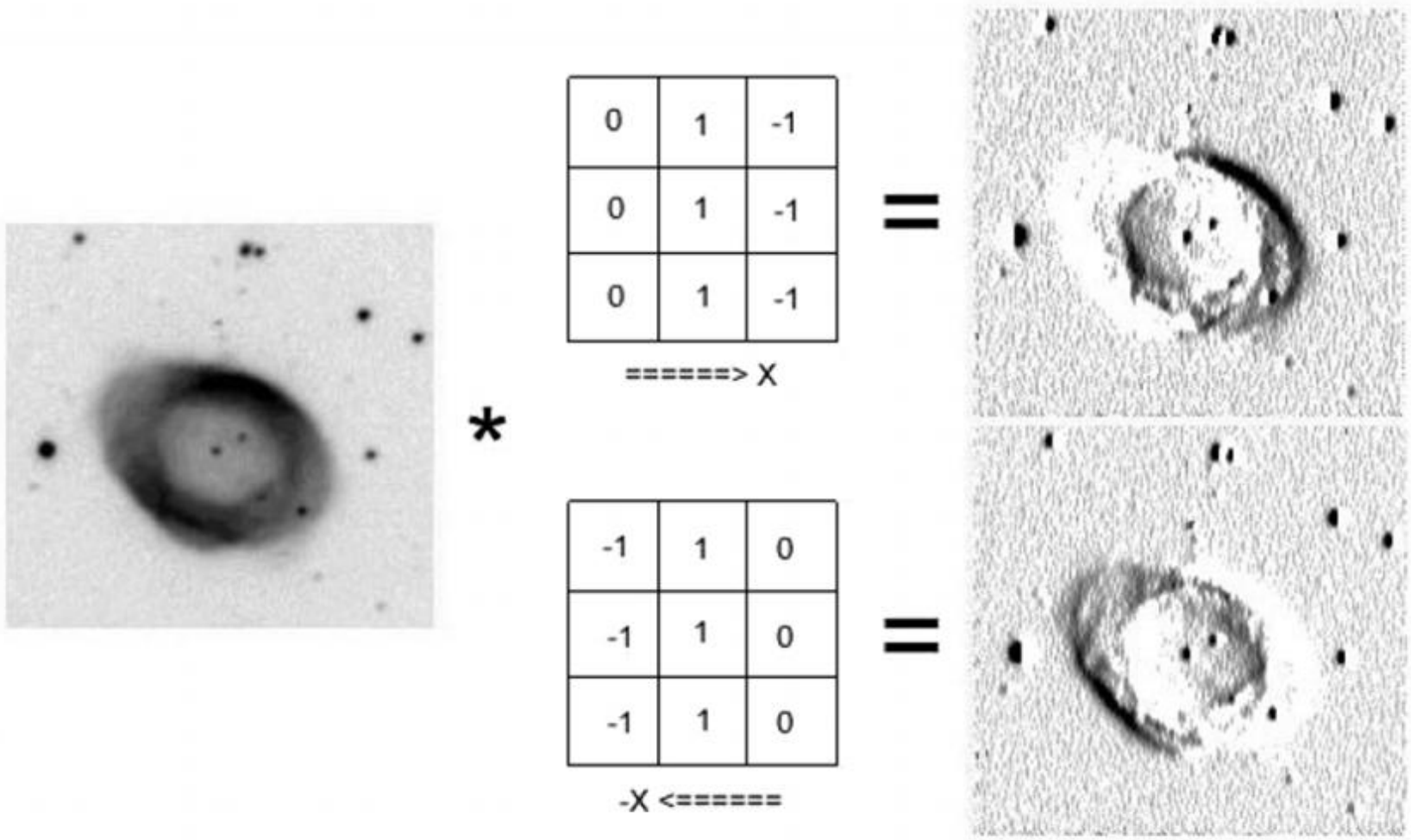


FIGURE 13 – Gradient selon la direction X ou (-X)

L’effet visible est un effet de relief qui permet de visualiser de faibles variations de luminosité.
Les contours obtenus à l’aide des filtres de gradient sont toutefois d’une qualité médiocre et ne peuvent en règle générale être utilisés tels quel car ils sont bruités, épais et non fermés. Ces défauts peuvent être compensés par des filtrages ultérieurs relativement simples (un seuillages par exemple) mais dont l’enchaînement est souvent délicat. Ces méthodes se caractérisent par ailleurs par une grande rapidité et un faible coût, du fait de l’aspect local de ces filtres.

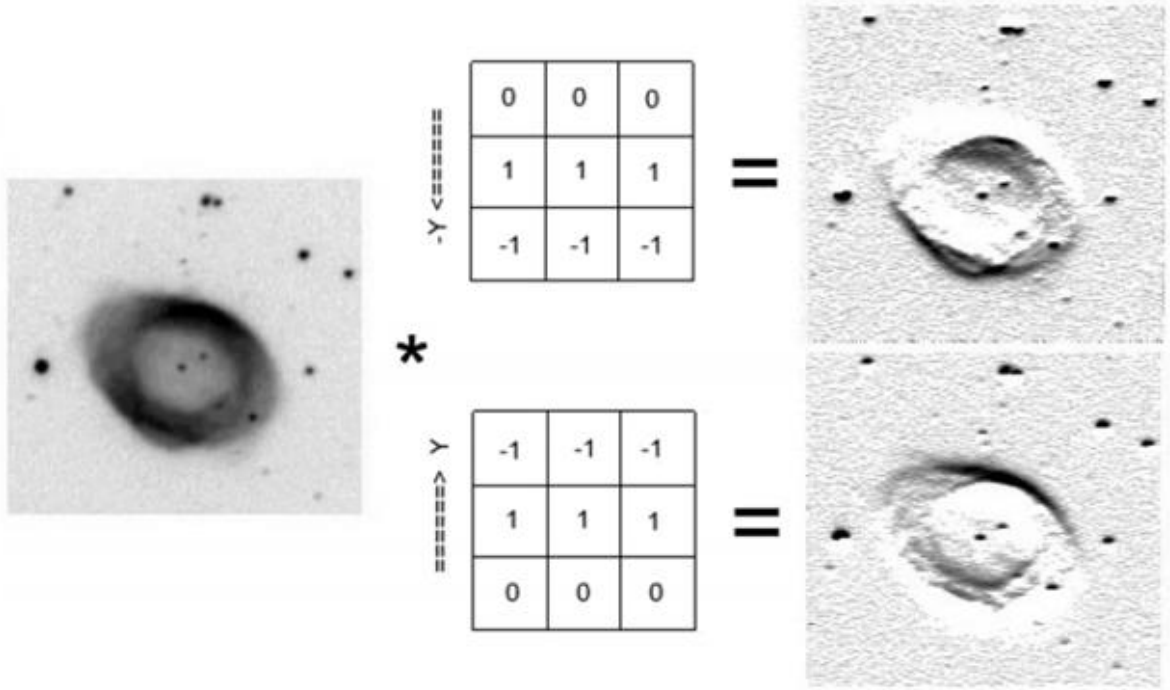


FIGURE 14 – Gradient selon la direction Y ou (-Y)

Voici un autre exemple de filtre gradient : le filtre de Sobel :

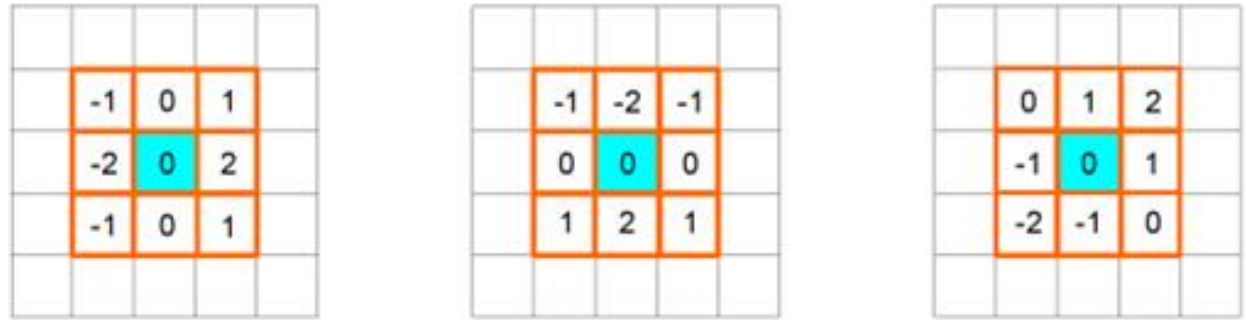


FIGURE 15 – Également calculé sur 9 points, le filtre de Sobel est une variante du filtre de Prewitt mais permet de privilégier le calcul suivant certaines directions (horizontale, verticale, obliques). On utilisera un facteur de normalisation de 1/4 avec ce filtre.

Filtre Laplacien

Le filtre Laplacien est un filtre de convolution particulier utilisé pour mettre en valeur les détails qui ont une variation rapide de luminosité. Le Laplacien est donc idéal pour rendre visible les contours des objets, d'où son utilisation dans la reconnaissance de formes.

Une frontière est un lieu de variation. On peut donc la localiser par la recherche du maximum de la dérivée première (approche basée sur le calcul du gradient). On peut aussi rechercher le passage par zéro de la dérivée seconde (également appelé zéro-crossing).

Dans le cas d'une image, il n'existe pas une dérivée seconde unique mais 4 dérivées partielles (selon x^2, y^2, xy et yx).

En pratique, on lève cette ambiguïté en ayant recours à l'opérateur Laplacien qui fait la somme des deux dérivées partielles principales. Le filtre Laplacien recherche donc les passages par zéro de la dérivée seconde du niveau de gris. Ils sont précédés par des filtres passe-bas à très large support (de taille 10x10 ou 30x30pixels) car ils sont très sensibles au bruit.

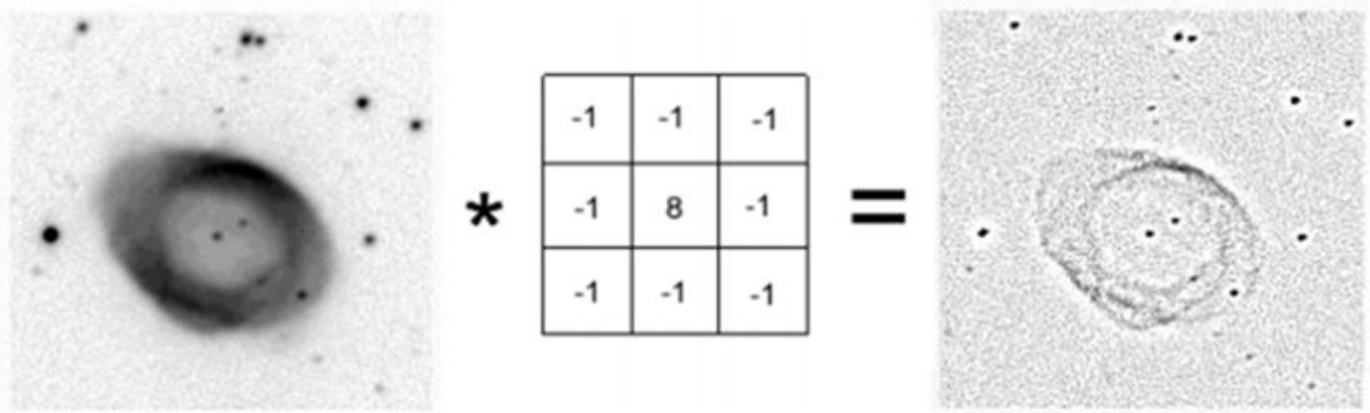
D'un point de vue mathématique, le Laplacien est une dérivée d'ordre 2, à deux dimensions:

$$L(x,y) = \frac{\partial^2 I}{\partial x^2}(x,y) + \frac{\partial^2 I}{\partial y^2}(x,y)$$

Dans le cas du traitement d'image, l'image de départ I (x, y) n'est pas une fonction continue, mais une fonction discrète à cause de la numérisation effectuée. Mais on peut tout de même obtenir la dérivée seconde en bonne approximation.

Il existe trois noyaux typiques de taille 3 × 3 qui peuvent remplir ce rôle :

À titre d'exemple on a appliqué à l'image de M 57 le deuxième noyau :



Un détail est à noter : la somme de tous éléments du noyau d'un filtre Laplacien est toujours nulle, ce qui implique que ce filtre n'est PAS un filtre linéaire.

0	-1	0
-1	4	-1
0	-1	0

-1	-1	-1
-1	8	-1
-1	-1	-1

1	-2	1
-2	4	-2
1	-2	1

FIGURE 16 – Exemples de noyaux de filtre laplacien

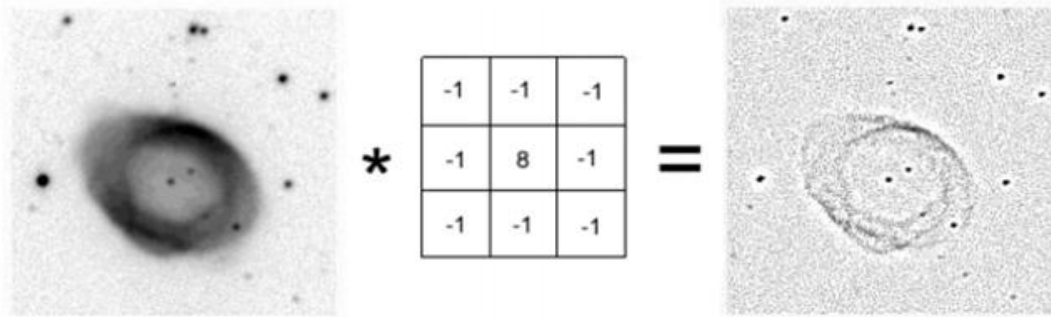


FIGURE 17 – Filtre laplacien

Voici pour finir un exemple de comparaison de l'efficacité d'une recherche de contour sur un visage, entre les filtres gradient et laplacien :

FIGURE 18 – Image originale, filtrage de Sobel et filtrage Laplacien

Convolution avec scipy

Numpy.convolve ne s'applique qu'à des tableaux à une seule dimension (vecteur). Pour pouvoir faire une convolution sur une matrice, on va utiliser la fonction convolve de Scipy :

```
from scipy.ndimage import convolve
convolve(data, kernel)
```

Doc:https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.convolve.html

On peut aussi utiliser la fonction

```
scipy.signal.convolve2d
```

qui fera le calcul plus rapidement (car elle est implémentée en C compilé) et possède d'autres options pour les pixels des bords.)

Doc: https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.convolve2d.html

Traitement d’images 2

Séance d’exercices

du cours de

Mathématique pour Informaticien

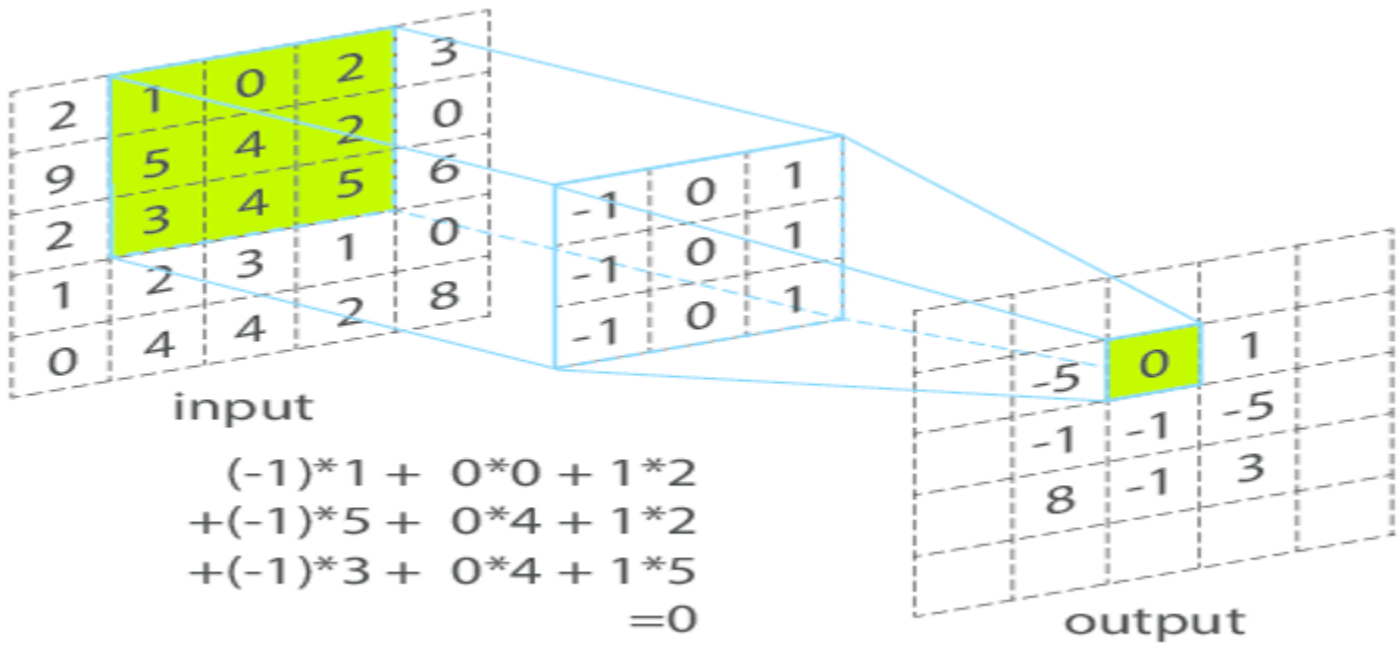
Exercice 1

1. Ecrire votre propre fonction/ programme qui fait la convolution entre une matrice de grande taille et un masque dont les noms sont passés en paramètre.

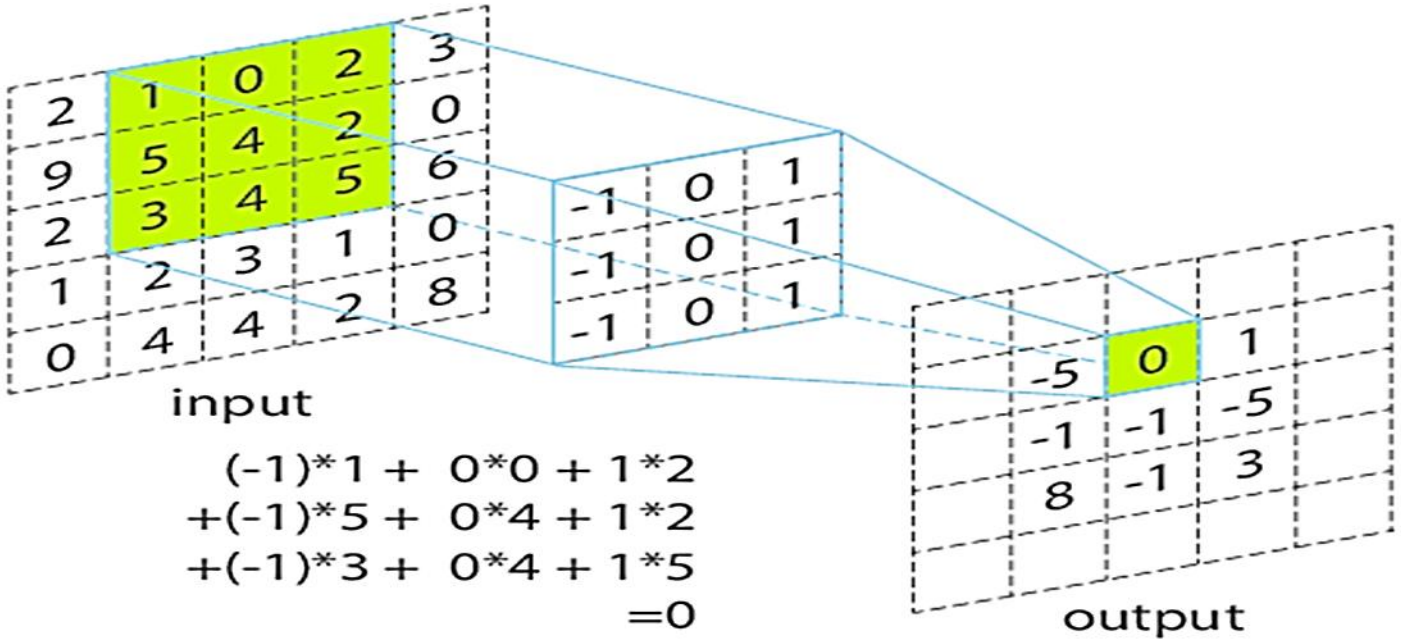
2. Tester le bon fonctionnement en l'utilisant sur la matrice et le masque de l'exemple page 10 et 11 du syllabus sur les tenseurs.

Filtres

Les filtres sont des modifications un peu plus complexes de notre image que celle que nous avons vues. Ils demandent un traitement mathématique un peu plus conséquent mais qui est au final toujours le même. Ils vont nous permettre entre autre de flouter, réhausser les contours, ou détecter les contours. Comme cette partie est un peu plus complexe mais intéressante, j'ai préféré donner déjà des scripts qui fonctionnent pour ensuite les modifier légèrement si besoin.



Pour modifier notre image, nous allons faire ce qu'on appelle une **convolution** de notre image par un noyau de dimension 3x3.



Autre exemple : le calcul qui donne -5 comme premier pixel de l'image modifiée sur la figure ci-contre est : (-1)x2 + 0x1 + 1x0 + (-1)x9 + 0x5 + 1x4 + (-1)x2 + 0x3 + 1x4 = -5

Le tableau de nombres sur la gauche représente notre image. Le tableau central (contenant des -1, 0 et 1) est notre noyau, fixé en fonction du filtre que l'on souhaite appliquer. Pour obtenir la valeur d'un pixel de notre image transformée, on "applique" notre noyau à la zone (en vert sur l'image) autour du pixel choisi de notre image d'origine. Pour l'appliquer, on multiplie simplement terme à terme puis on fait la somme. L'image est surement plus parlante que mes explications.
On applique ainsi notre filtre sur chaque pixel pour lequel c'est possible (on voit bien qu'au bord il y a un problème).

En pratique, si notre noyau est sous la forme d'une matrice 3x3, on aura pour le pixel dont on connait la ligne « ligne » et la colonne « col » :

noyau[0,0]*image[ligne-1,col-1] + noyau[0,1]*image[ligne-1,col] + noyau[0,2]*image[ligne-1,col+1] + noyau[1,0]*image[ligne,col-1]+...+noyau[2,2]*image[ligne+1,col+1].

On va, bien sûr, faire deux boucles de longueur 3 pour calculer cette somme.

Les pixels de bord sont simplement recopiés pour garder une image de même dimension.

Exercice 2

Idem mais avec la fonction convolve de Scipy.
Le résultat est-il identique ? Pourquoi ?

Exercice 3 Lisser une image

Pour lisser une image, il suffit de prendre comme noyau 1/9 partout pour que le calcul corresponde, pour chaque pixel, à faire la moyenne des 9 valeurs autour autrement dit :

$$noyau = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Implanter ce filtre de flou linéaire avec Scipy ou votre programme par convolution avec ce masque de 3x3 puis avec un masque de 10x10 et appliquez-le à une image 512x512

Exercice 5 Lisser une image filtre Gaussien

Il y a d'autres façons de lisser une image. On peut par exemple prendre un filtre gaussien qui prend un peu plus en compte le pixel central et un peu moins ceux qui l'entourent.

$$noyau = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Implanter un filtre de flou Gaussien avec Scipy ou votre programme et appliquez-le à une image 512x512

Exercice 5 filtre flou directionnel

Imaginez le masque d’un filtre de flou directionnel de gauche à droite (comme si la personne ne bougeait que dans cette direction)
et testez-le

Exercice 6 filtre flou médian

Créez un filtre de flou basé sur la médiane (filtre médian non linéaire)
Et testez-le

Exercice 7 Réhausser les contours

Un contour est un endroit de l'image où les pixels changent totalement de couleurs.
Pour amplifier les contours, on peut donc utiliser un filtre de la forme :

$$noyau = \begin{bmatrix} 0 & -0.5 & 0 \\ -0.5 & 3 & -0.5 \\ 0 & -0.5 & 0 \end{bmatrix}$$

Modifier le 3 en 2 dans le noyau précédent (ce qui fait une somme nulle des coefficients du noyau) pour voir apparaitre les contours.

En plus de transformer le 3 en 2, inverser les couleurs de l'image pour voir apparaitre seulement les contours. Cela donne une impression de dessin au crayon.

Remarque : L'utilisation de numpy peut obliger ici à modifier les typages pour que les calculs soient justes.

Exercice 8 Détection des contours

Grâce au filtre précédent, on a pu mettre en valeur les contours.
Il existe des filtres encore plus efficaces pour détecter les contours. Nous allons donner l'exemple du filtre de Sobel.
On applique en réalité 2 filtres qui sont :

$$noyau\_v = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \; et \; noyau\_h = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

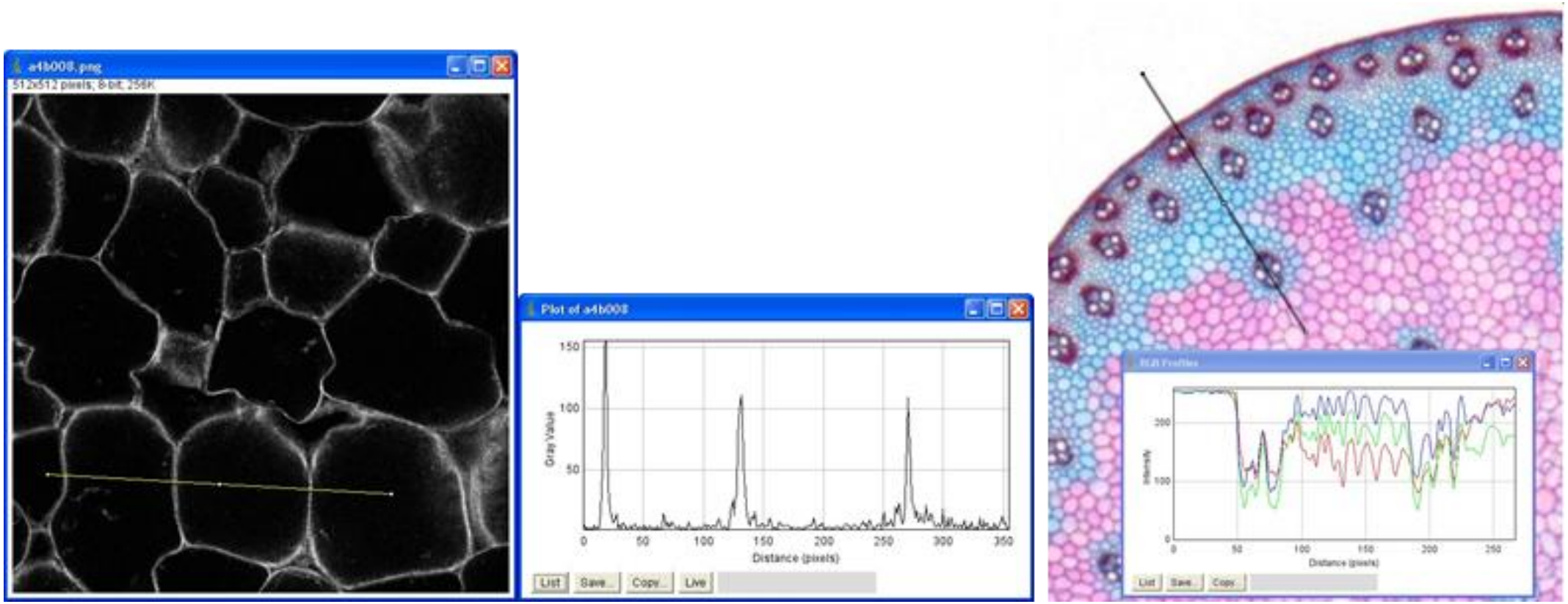
Le premier détecte les contours "verticaux" et le second les "horizontaux".
Pour le voir, vous pouvez mettre un puis l'autre comme noyau dans l'exemple précédent pour voir les contours mis en valeur.

L'idée est donc de faire les mêmes calculs que dans le cas d'un seul filtre. On obtient donc ici deux résultats somme_v et somme_h.
Pour combiner ces deux résultats en un seul, on fait ici une moyenne quadratique $\sqrt{somme\_v^2 + somme\_h^2}$ (car cela correspond en réalité à la norme d'un vecteur).
La détection des contours rend beaucoup mieux si on modifie avant l'image en niveau de gris et au final on inverse les couleurs.

Exercice 9 : Profil d'intensité

Un outil interactif très intuitif pour visualiser les variations d'intensité dans une image est le profil des intensités.

Il faut tout d'abord choisi un outil de sélection linéaire (segment, droite polygonale...), et tracer la région d'intérêt désirée. Le profil d'intensité permet de visualiser les variations d'intensité le long de la sélection.



Sources:

Le traitement d'images pour les nuls, D. Legland, juin 2019

https://www.codingame.com/playgrounds/17176/recueil-dexercices-pour-apprendre-python-au-lycee/manipulations-dimages-i