

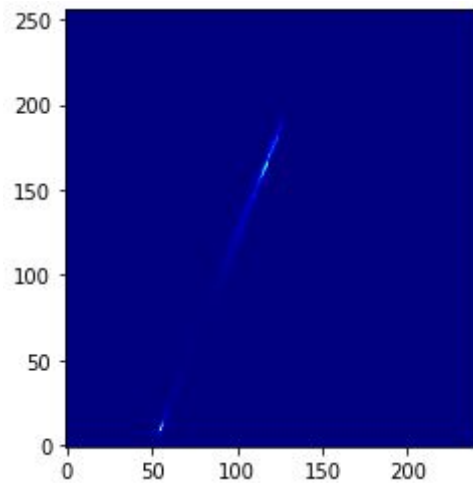
Reconstruction et analyse d'images médicales

TP2 - Recalage

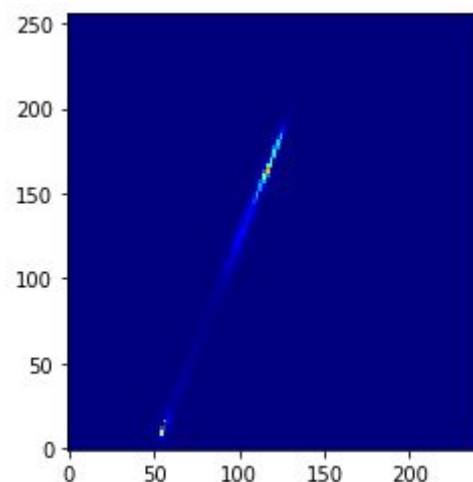
Q1 - Histogramme Conjoint

Q1a,b.

On va s'intéresser ici à l'histogramme conjoint de I2 et J2. On peut ainsi l'afficher sur 255 bins :



Si on réduit le nombre de bins à 100 (ce n'est pas exactement 100 bins puisque nous avons des intervalles entiers) on peut obtenir :



On souhaite confirmer l'équation suivante :

$$\sum_{i,j} H_{I,J}(i,j) = n * p$$

La taille de I1 et J1 est de 507*512 = 259584

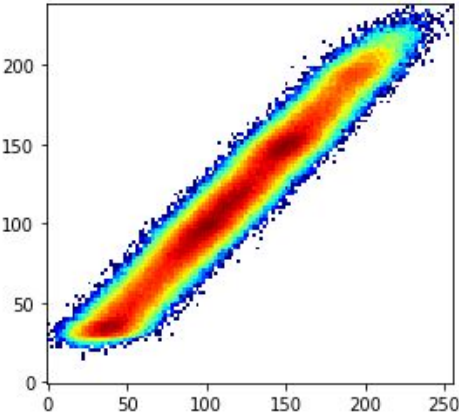
Si on vérifie la somme sur notre image avec :

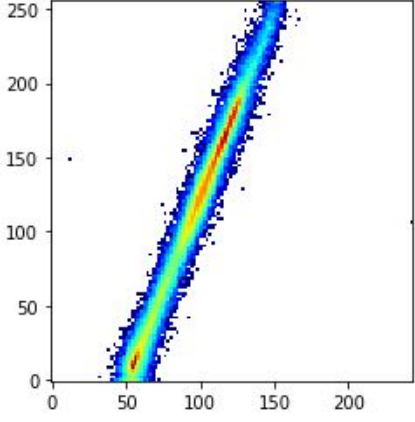
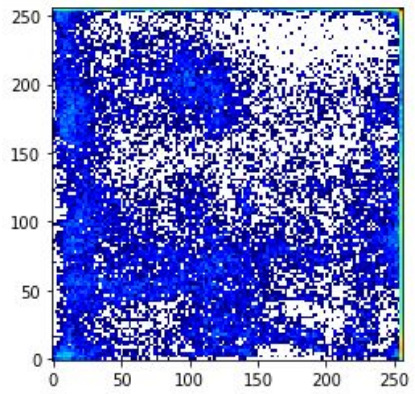
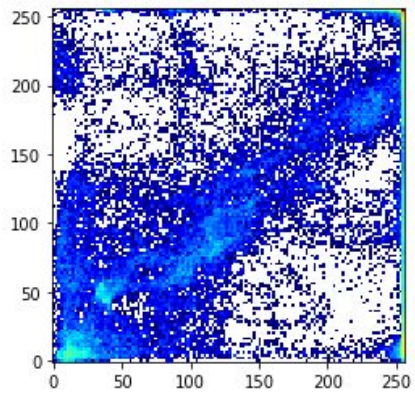
```
for i in range (Conjoint.shape[0]) :  
    for j in range (Conjoint.shape[1]):  
        sum = sum + Conjoint[i,j]  
print(sum)
```

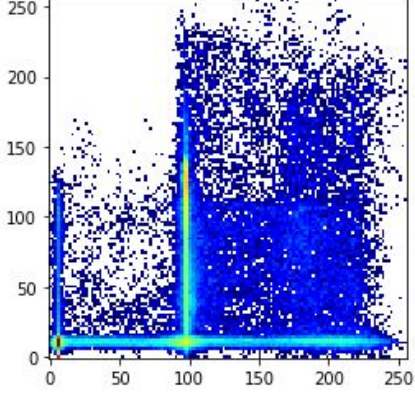
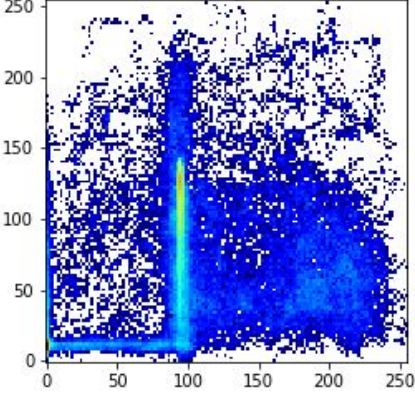
On obtient bien que la somme vérifie $n * p = 259584$.

Q1c.

Une échelle logarithmique a été utilisée pour la visualisation et on affichera les histogrammes avec 100 bins.

Histogramme conjoint	Description
	La présence d'une droite croissante nous indique que la paire d'images est corrélée positivement . L'étalement de la droite est peut être dû au bruit différenciant les images J1 et I1.

 <p>I2/J2</p>	<p>La présence d'une droite croissante très définie nous indique que les images I2/J2 sont très corrélées positivement. Cela est cohérent avec le fait que la différence entre ces deux images est un changement d'intensité uniquement.</p>
 <p>I3/J3</p>	<p>L'histogramme ne semble pas révéler de structures particulières, il est réparti relativement régulièrement. Cela signifie que les images I3/J3 sont trop différentes (visages différents + translation + changement d'échelle), elles ne sont pas corrélées.</p>
 <p>I4/J4</p>	<p>On peut déceler une forme s'approchant d'une droite croissante, il existe une certaine corrélation positive entre la paire d'images. Cela signifie que les deux visages présents sur ces images sont très similaires.</p>

 <p style="text-align: center;">I5/J5</p>	<p>On peut observer la présence d'une droite verticale pour une intensité de 100 de l'image J5. Il y aussi une droite horizontale pour une intensité de 0 de l'image I5. La présence de la droite verticale est causée par un débruitage (cf paire I6/J6). La droite horizontale est quant à elle causée par la symétrie qui a été appliquée sur la paire. En effet, un pixel blanc (boîte crânienne) sur l'image I5, prend une grande variété de valeurs dan l'image J5.</p>
 <p style="text-align: center;">I6/J6</p>	<p>On peut observer une droite verticale pour une intensité de 100 sur l'image J6. Cela est causé par le débruitage qui différencie la paire. En effet, la matière cérébrale de l'image I6 prend une variété de valeurs d'intensité tandis que sur l'image J6, elle est uniformisée à une intensité autour de 100.</p>

Q2 - Critères de similarité

Pour les critères de similarité SSD et CR, nous avons utilisé les images avec leur intensité normalisée tandis que que pour l'information mutuelle nous avons utilisé des images non normalisées (intensité entre 0 et 255).

On compare les 3 critères de similarité :

Images comparées	SSD (somme des différences au carré)	CR (coefficient de corrélation)	IM (information mutuelle)
I1/J1	409	0.978	1.555
I2/J2	7280	0.996	1.528
I3/J3	13391	0.143	0.835
I4/J4	5571	0.564	1.035
I5/J5	10763	0.656	0.517

I6/J6	6544	0.78	0.476
-------	------	------	-------

En observant les résultats de la SSD, on constate que les images I1/J1 sont celles qui ont la valeur minimale. Cela paraît logique puisque la différence entre ces deux images provient d'un ajout de bruit, il existe une différence entre pixels seulement sur un petit ensemble de pixels, ceux qui ont subi le bruit.

On peut remarquer que la SSD des images I2/J2 est relativement élevée. Cela est dû au fait que la différence entre ces deux images porte sur une modification de l'intensité. La différence entre chaque pixel est uniforme ce qui mène à une forte SSD malgré des images très similaires. C'est pour cette raison que la SSD est normalement utilisée sur des images de même intensité.

Lorsqu'on étudie les histogrammes conjoints des paires d'image 1 et 2, on peut observer une relation linéaire qui est confirmée par le coefficient de corrélation : leur CR est proche de 1, il y a donc une corrélation linéaire positive entre les images I1/J1 et I2/J2.

Au contraire, les images I3/J3 ne semblent pas être corrélées linéairement (CR proche de 0), ce qui est aussi cohérent avec leur histogramme conjoint qui ne présente pas de relation de linéarité.

En étudiant le critère d'information mutuelle, on constate que les paires d'images les plus liées sont I1/J1 et I2/J2 (IM la plus élevée) ce qui est en accord avec leur coefficient de corrélation (les paires semblent corrélées linéairement).

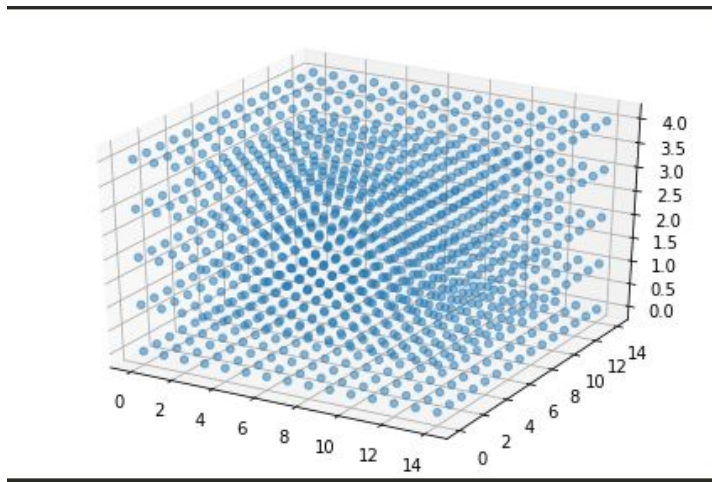
Malgré le fait que les paires I3/J3 et I4/J4 ne soient pas corrélées linéairement (d'après leur CR), elles conservent une IM relativement élevée, elles doivent être donc liées non linéairement.

Les paires ayant l'information mutuelle la plus faible sont les paires I5/J5 et I6/J6, ce qui peut s'expliquer par le fait que ces paires sont liées par un débruitage qui lisse l'image et cause une perte d'information. La perte d'information entre I5 et J5 (et I6 et J6) se retrouve dans une information mutuelle faible.

Q3 - Transformations spatiales

Qa.

A l'aide de la fonction meshgrid de numpy on est capable d'afficher une grille.



Qb.

Les opérations utilisées sont les suivantes :

Translation

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & p \\ 0 & 1 & 0 & q \\ 0 & 0 & 1 & r \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Rotation par rapport à x

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Rotation par rapport à y

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\omega) & 0 & -\sin(\omega) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\omega) & 0 & \cos(\omega) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Rotation par rapport à z

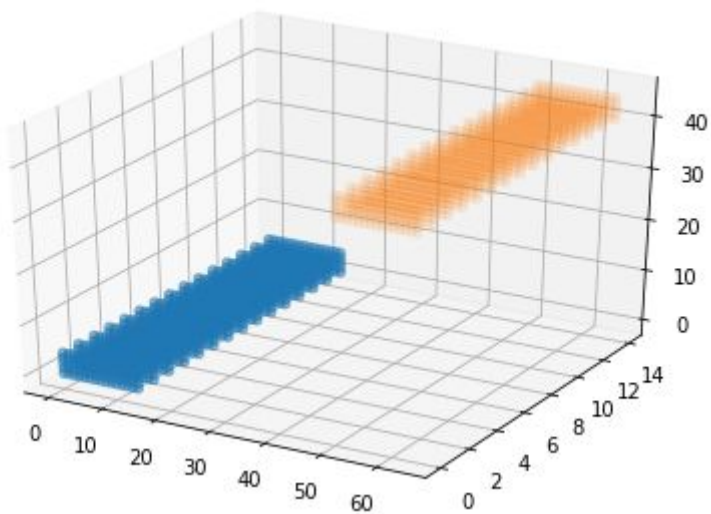
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 & 0 \\ \sin(\phi) & \cos(\phi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$T(p, q, r) * R_z(\phi) * R_y(\omega) * R_x(\theta)$$

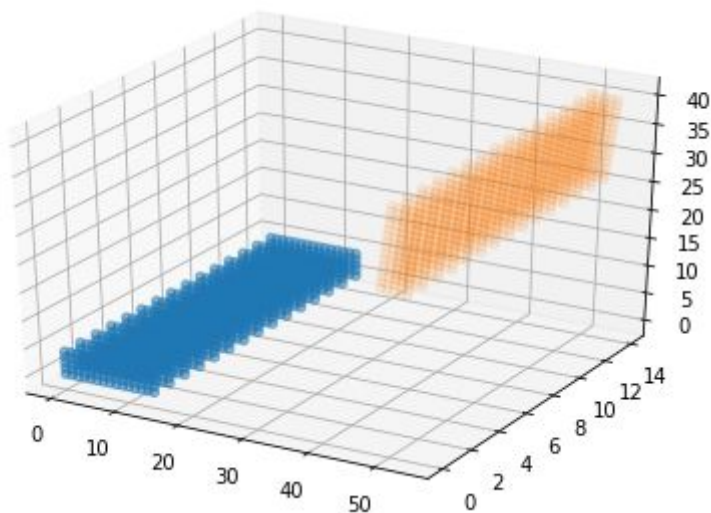
Ici on crée la matrice qui va nous permettre d'effectuer les différentes opérations de translation et de rotation. On va combiner les différentes matrices qui se situent ci dessus

Exemple :

On a ici un translation sur les axes x et z :



On peut ajouter une rotation sur l'axe y :

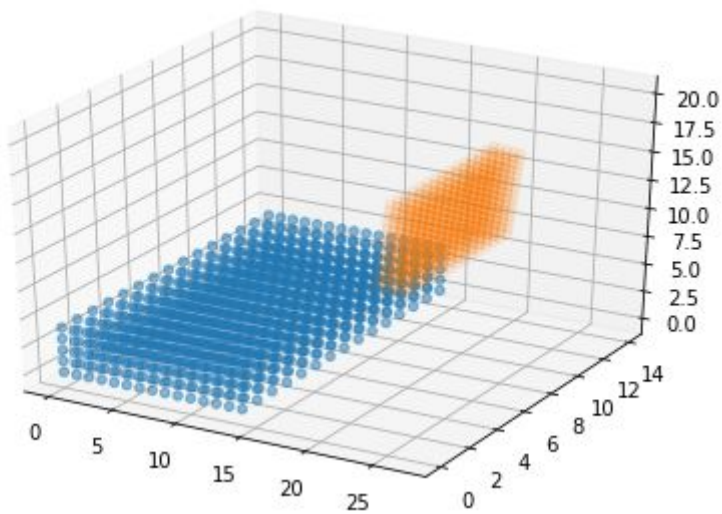


Qc.

$$\begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

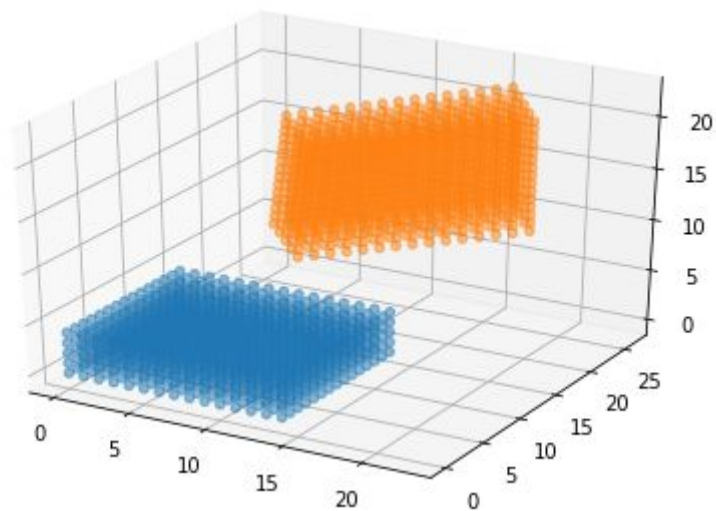
On ajoute cette matrice à l'opération pour pouvoir effectuer l'homothétie

Si on met $s = 0.5$ on obtient ainsi :



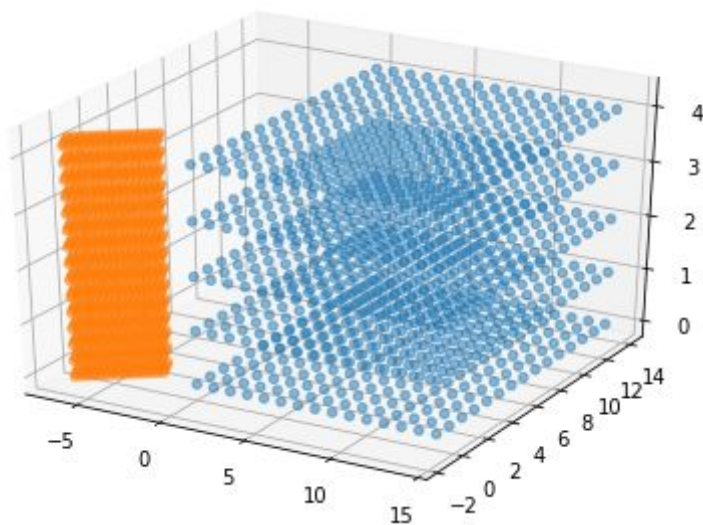
Q3.d.

Voici les résultats après application de la matrice M1 au nuage de points :



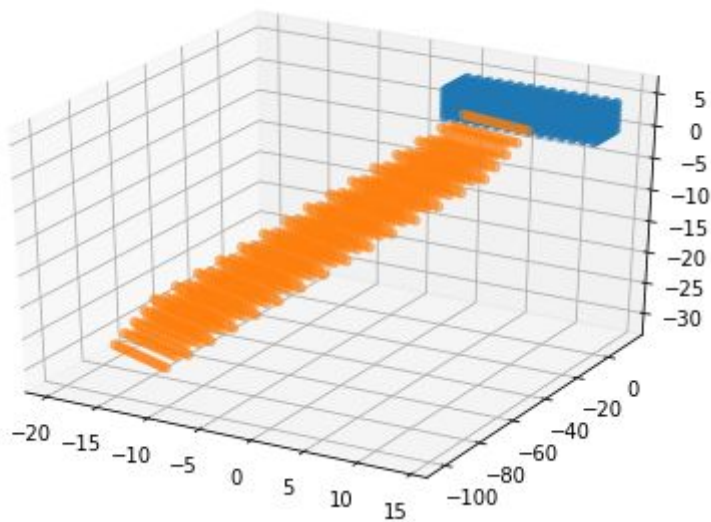
Ainsi la matrice M1 permet de réaliser une **translation sur les axes x y et z** ainsi que des **rotations par rapport aux axes y et z**.

Voici les résultats après application de la matrice M2 au nuage de points :



On peut d'abord observer une opération de **scaling** diminuant la taille de la matrice. Il y également eu une **translation** sur l'axe x ainsi que des **rotations** par rapport aux axes x et y.

Voici les résultats après application de la matrice M3 au nuage de points :



Un **scaling** est réalisé pour augmenter la taille de la matrice. Il y également une **translation** et une **rotation** sur l'axe y.

Q4 - Recalage iconique 2D simple

Qa. Fonction translation

Nous avons d'abord implémenté "à la main" une fonction de translation ainsi deux interpolations différentes pour gérer des translations par des valeurs décimales :

- interpolation des plus proches voisins
- interpolation bi-linéaire

Pour la suite de ce TP, pour des besoins de rapidité de calcul, nous avons utilisé la méthode *ndimage.interpolation.shift* de la librairie Scipy. Cette méthode permet de réaliser plusieurs interpolations.

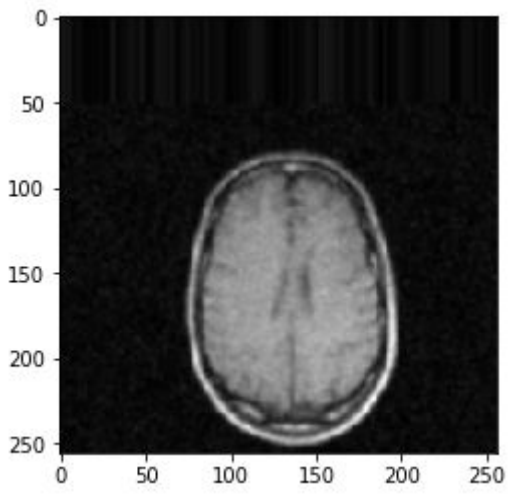
Qb. Recalage translation

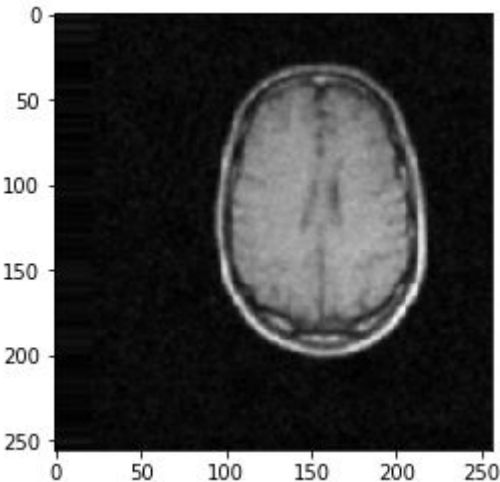
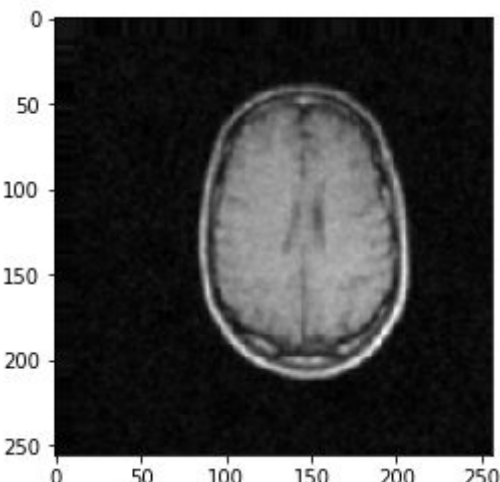
Avant d'essayer de recalibrer les images de cerveau, on réalise un **débruitage gaussien** ($\sigma=1$) pour atténuer le bruit présent sur ces IRM.

On réalise le recalage à l'aide de la méthode **Lucas-Kanade itérative** et la translation se fait avec une interpolation des plus proches voisins. Nous avons également implémenté un recalage par descente de gradient qui sera nécessaire pour les questions suivantes.

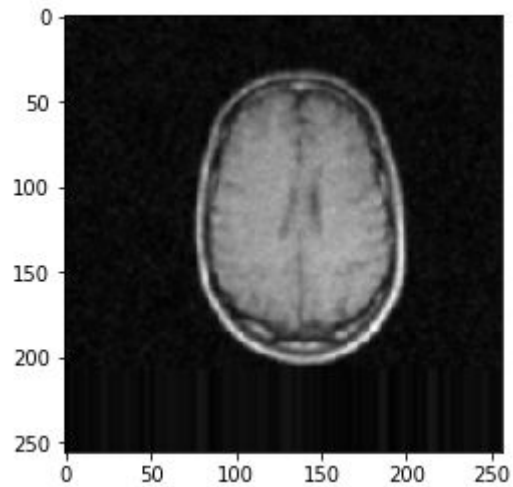
Pour minimiser la SSD, on réalise le recalage sur un grand nombre d'itération (800) et on conserve le recalage dont l'énergie est minimale.

On réalise un recalage depuis les translations suivantes :

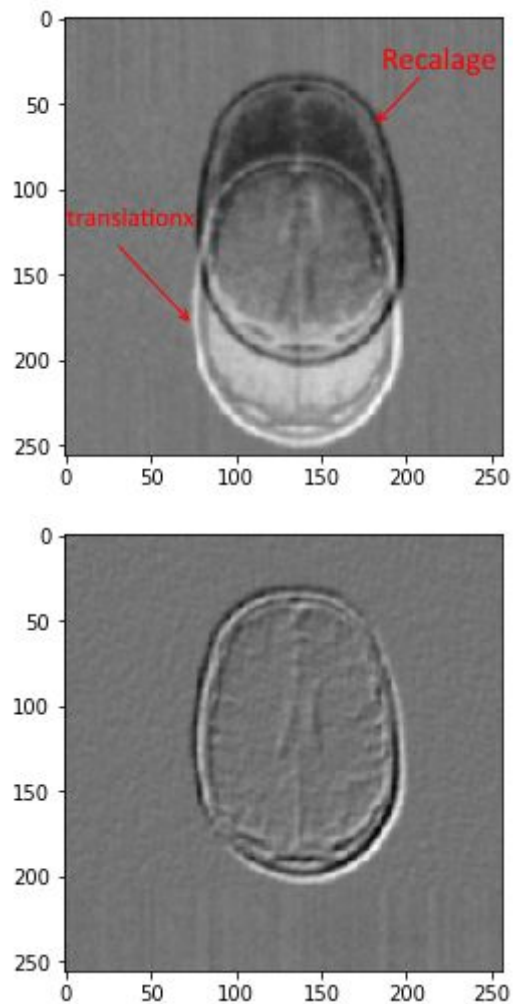
u	translation
[50,0]	

[0,20]	 <p>translationxy</p>
[10,10]	 <p>translationxy</p>

Recalage vers BrainMRI_1 de :	Image recalée
translationx	L'image recalée avec la SSD la plus faible(304) est :



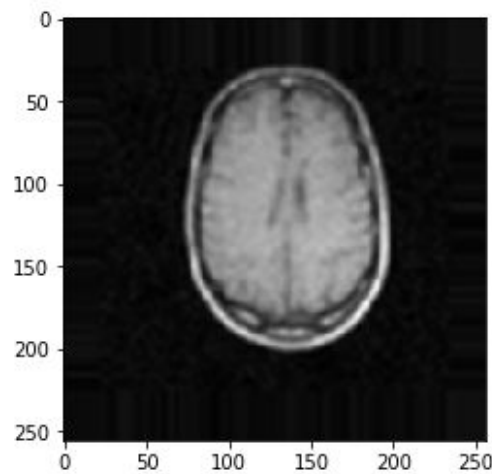
En la comparant à l'image translationx et également à l'image objectif BrainMRI_1, on obtient :



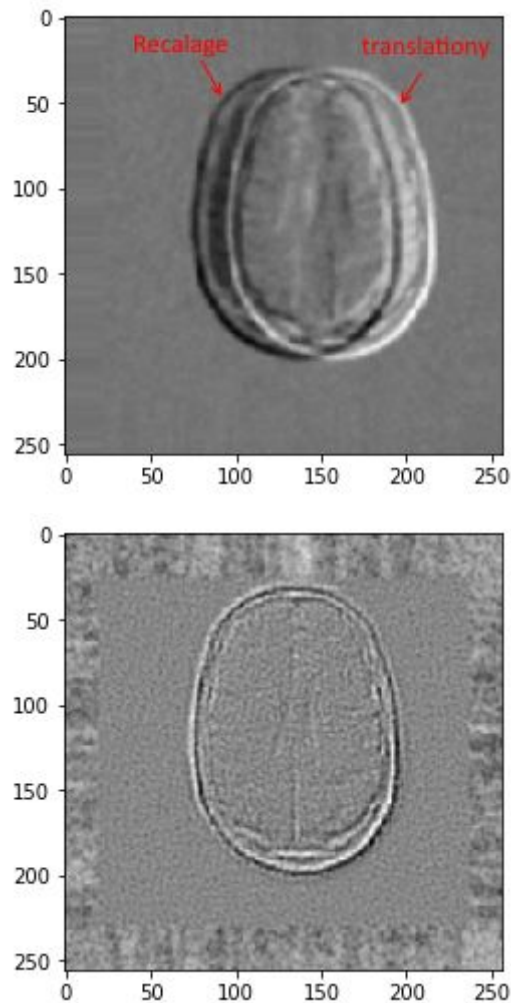
On peut constater, en regardant la dernière image, que le résultat est un recalage très satisfaisant.

translationy

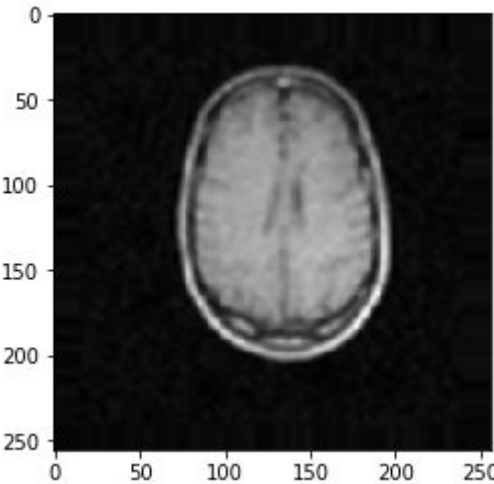
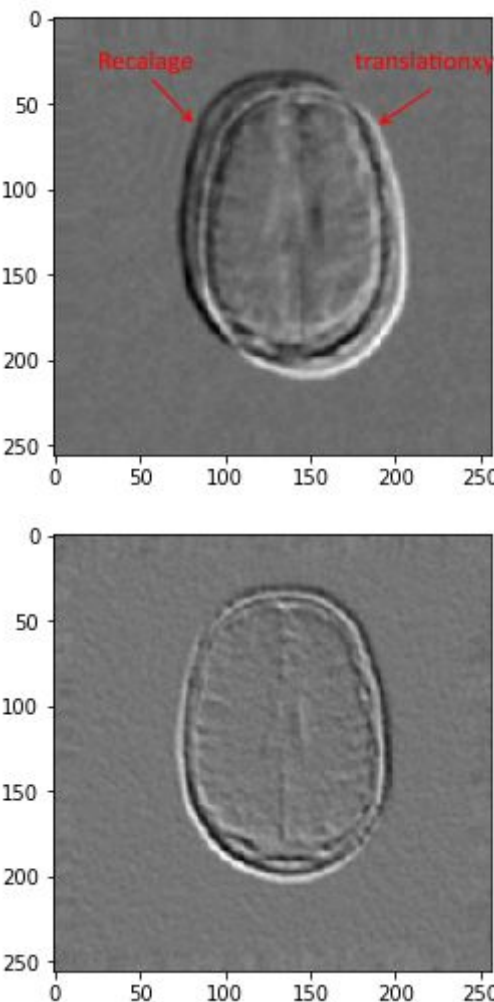
L'image recalée avec la SSD la plus faible(**14.0**) est :



En la comparant à l'image translationy et également à l'image objectif BrainMRI_1, on obtient :



Le recalage est également satisfaisant. Sa SSD est plus faible que le recalage précédent car c'est un recalage d'une translation plus faible (plus l'image est translaté, plus le nombre de pixels impactés par l'interpolation des plus proches

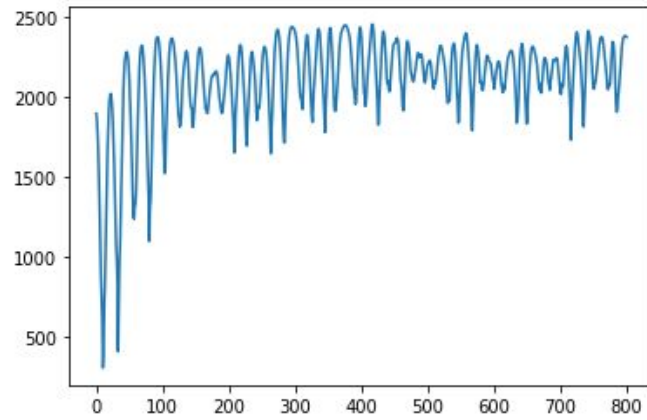
	<p>voisins augmente).</p> <p>Les artefacts présents sur les contours de l'images sont aussi causés par l'interpolation des plus proches voisins.</p>
translationxy	<p>L'image recalée avec la SSD la plus faible(96.0) est :</p>  <p>En la comparant à l'image translationxy et également à l'image objectif BrainMRI_1, on obtient :</p> 

De même, le recalage est satisfaisant.

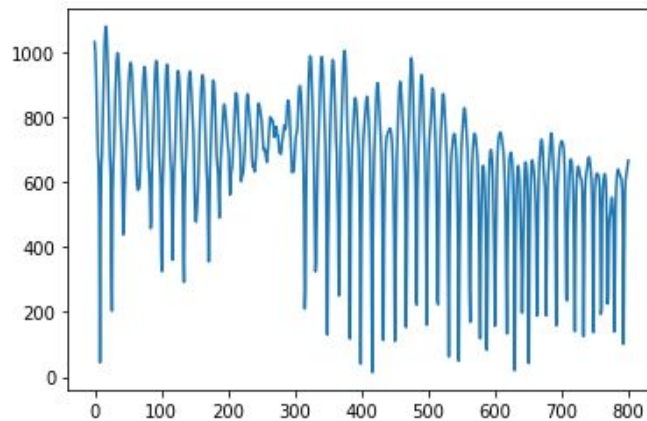
Recalage vers
BrainMRI_1 de :

Evolution de la SSD au cours des itérations

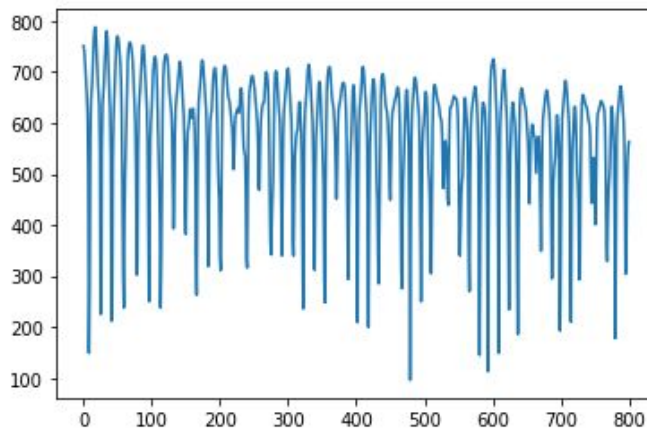
translationx



translationy



translationxy



On peut observer que pour toutes ces images, la courbe d'énergie n'est pas strictement décroissante, elle a un aspect **oscillatoire**. C'est pour cela que nous ne sélectionnons pas le dernier recalage comme résultat mais plutôt le recalage possédant la SSD minimale. Cet aspect oscillatoire provient peut être du fait que le recalage n'est jamais parfait : ainsi la méthode va probablement surcorriger (réaliser une translation trop grande dans la bonne direction) ce qui va ensuite nécessiter une translation dans le sens inverse. Cela est peut être la source de cet effet de yoyo.

Pour remédier à cela, il serait peut être intéressant d'affecter un coefficient variable au vecteur de translation u , qui diminuera lorsque la SSD diminue et vice-versa.

Qc. Fonction rotation

Pour réaliser la rotation, on utilise la méthode `image.rotate()` de la librairie PIL.

Cette méthode réalise une rotation centrale. Pour effectuer une rotation par rapport au coin en haut à gauche, on place l'image dans une matrice vide 4 fois plus grande, de sorte que son coin soit placé au centre de la matrice, on effectue la rotation, puis on tronque la matrice pour obtenir l'image tournée.

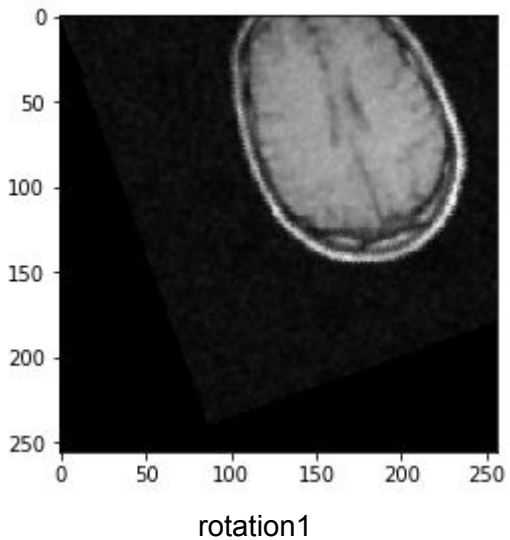
Qd. Recalage rotation

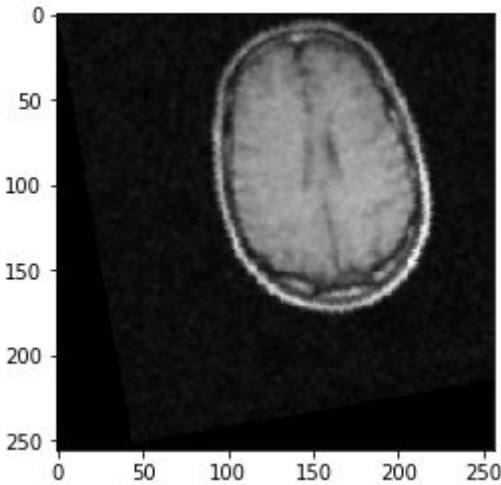
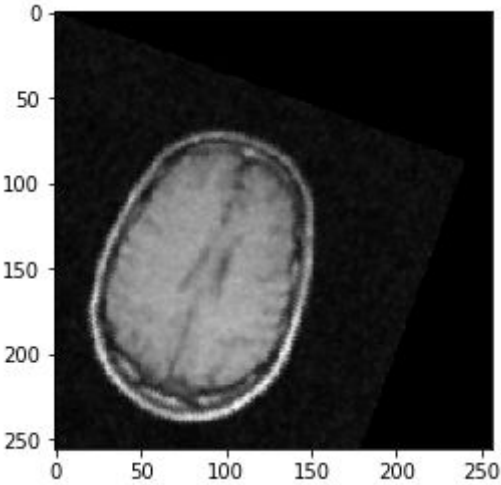
On continue de réaliser un recalage avec des images débruitées.

On réalise le recalage à l'aide de la méthode de descente de gradient.

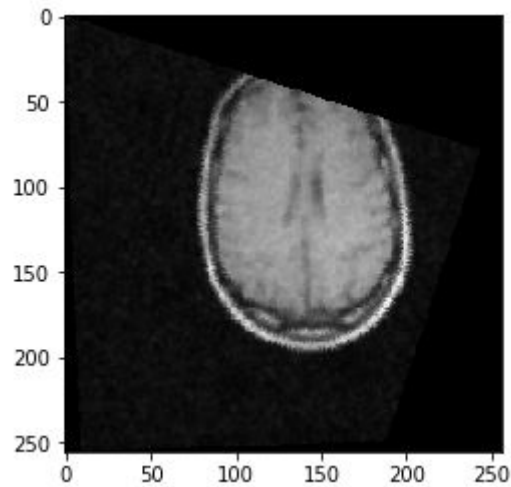
Pour minimiser la SSD, on réalise le recalage sur un grand nombre d'itérations (300).

On réalise un recalage depuis les rotations suivantes :

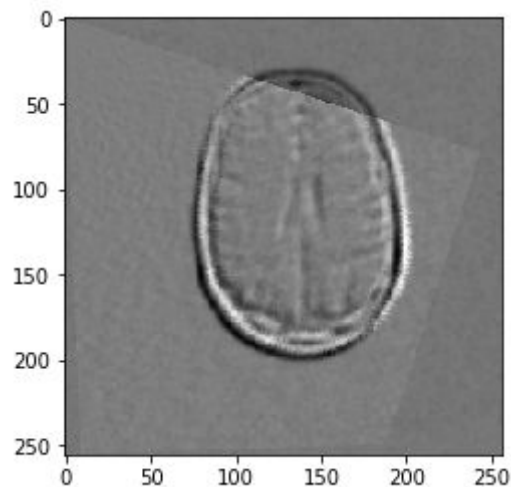
angle de rotation	rotation
20°	

10°	 <p>rotation2</p>
-20°	 <p>rotation3</p>

Recalage vers BrainMRI_1 de :	Image recalée
rotation1 $\epsilon=0.00001$	L'image recalée qui pour SSD 598.0 est :



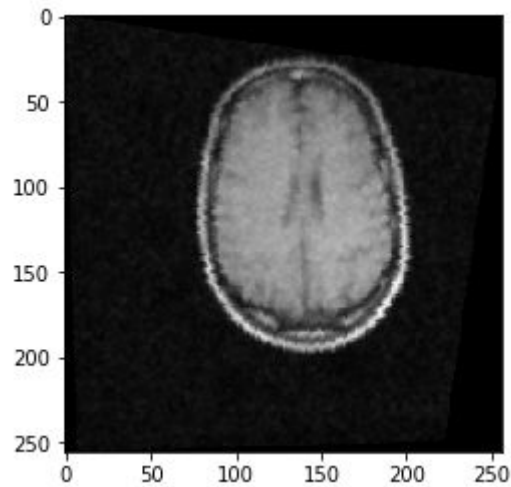
En la comparant à l'image à l'image objectif BrainMRI_1, on obtient :



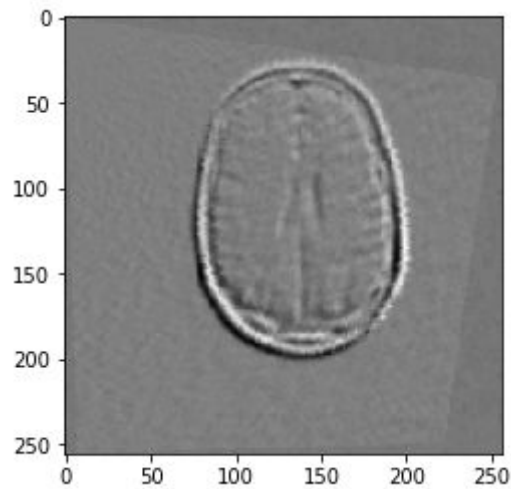
On peut constater, en regardant la dernière image, que le résultat est un **recalage très satisfaisant**. La source de SSD majeure entre ces deux images provient du fait que l'image rotation1 ne contient pas la totalité du cerveau (une partie a été tronquée).

rotation2
 $\epsilon=0.000003$

L'image recalée qui a pour SSD 494.0 est :



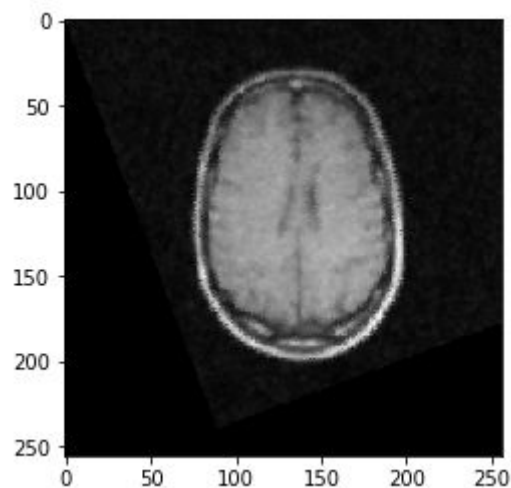
En la comparant à l'image objectif BrainMRI_1, on obtient :



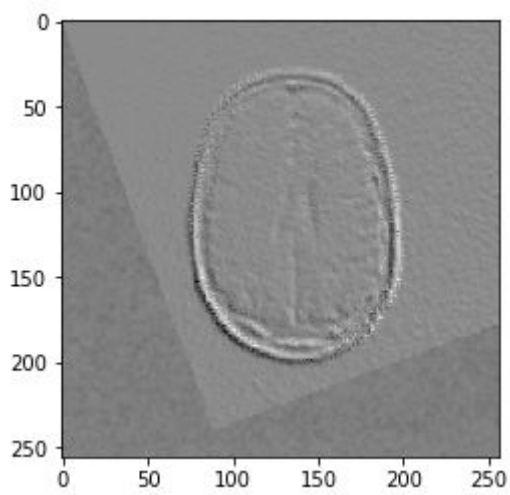
Le recalage est également satisfaisant. Sa SSD est plus faible que le recalage précédent car c'est un recalage d'une rotation plus faible (l'image initiale a été moins tronquée).

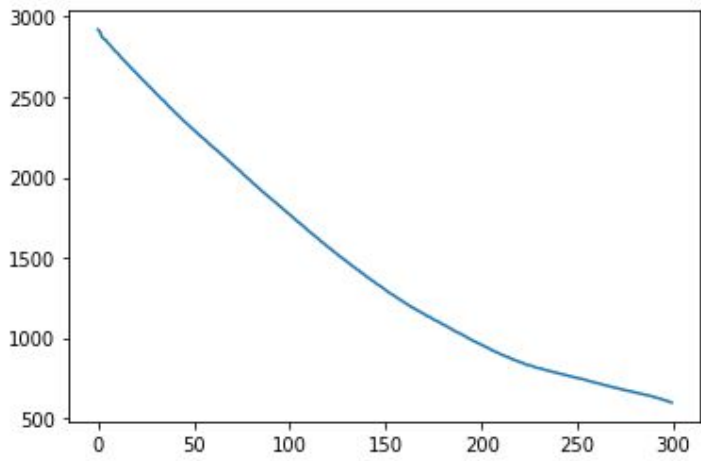
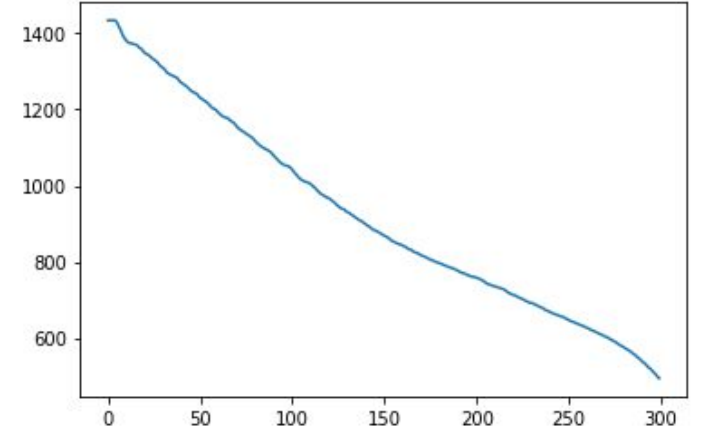
rotation3
 $\epsilon=0.000014$

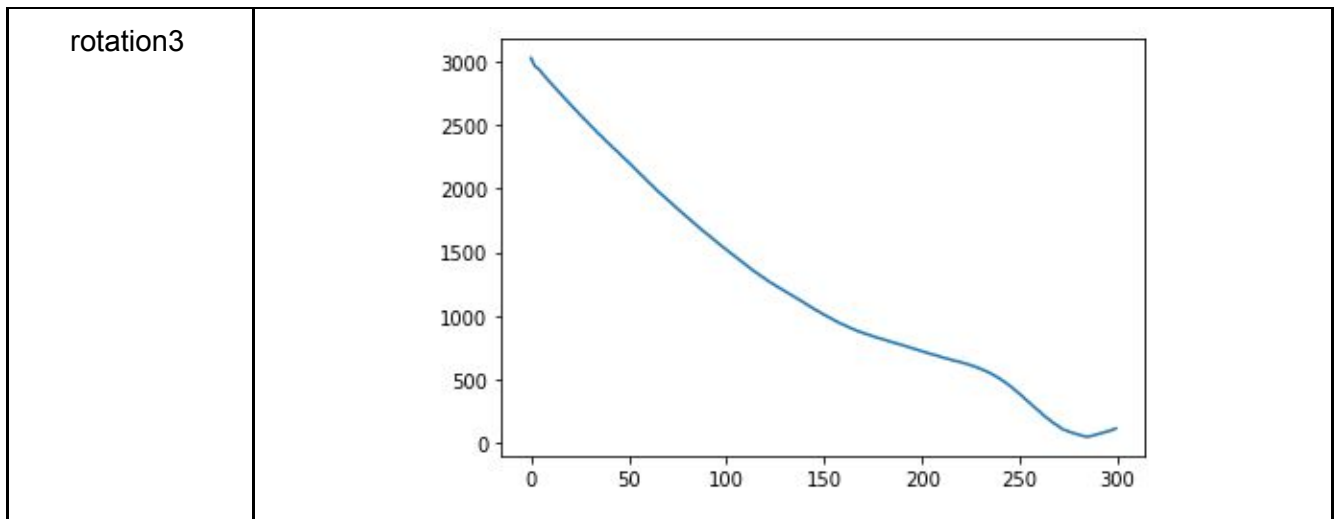
L'image recalée qui a pour SSD 111.0 est :



En la comparant à l'image objectif BrainMRI_1, on obtient :

	 <p>De même, ce recalage est satisfaisant le plus satisfaisant des 3. Il serait sûrement possible d'améliorer les autres en modifiant ϵ.</p>
--	--

Recalage vers BrainMRI_1 de :	Evolution de la SSD au cours des itérations
rotation1	
rotation2	

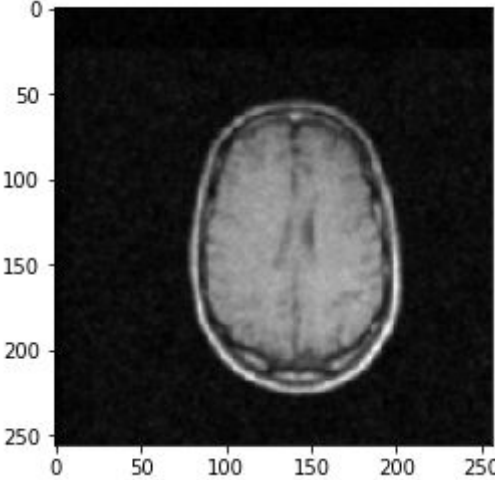


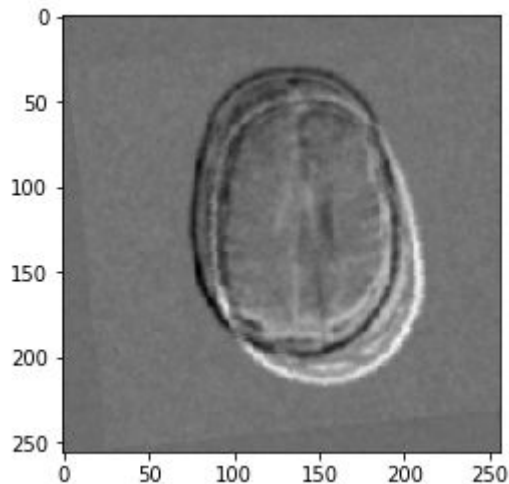
Qe. Recalage transformation rigide

Ce recalage regroupe les deux méthodes implémentées précédemment.

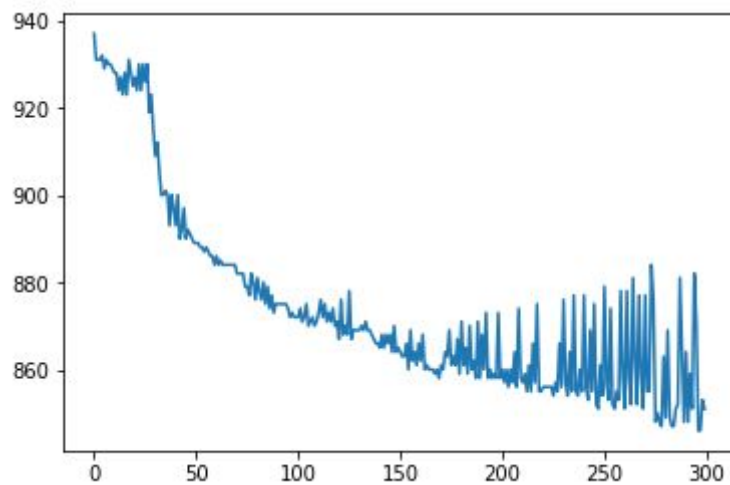
Nous continuons d'utiliser les IRMs débruitées lors du recalage.

Nous obtenons les recalages suivants :

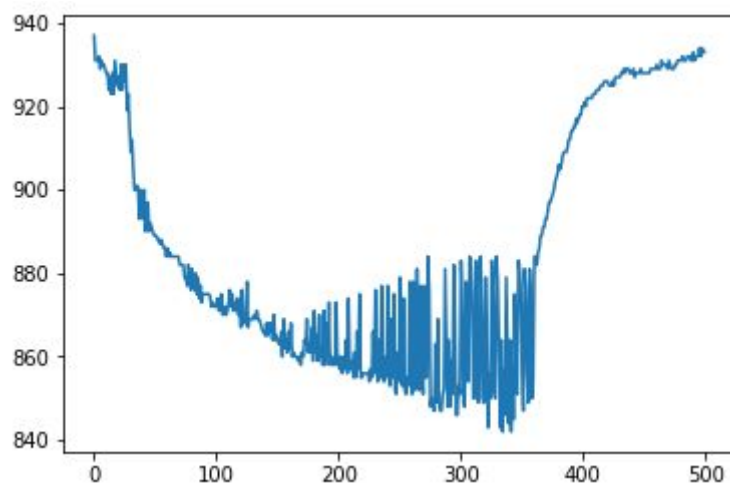
Recalage vers BrainMRI_1 de :	Résultats du recalage
BrainMRI_2 $\epsilon=0.0008$	<p data-bbox="660 1122 1222 1160">L'image recalée qui a pour SSD 851.0 est :</p>  <p data-bbox="555 1671 1326 1709">En la comparant à l'image objectif BrainMRI_1, on obtient :</p>



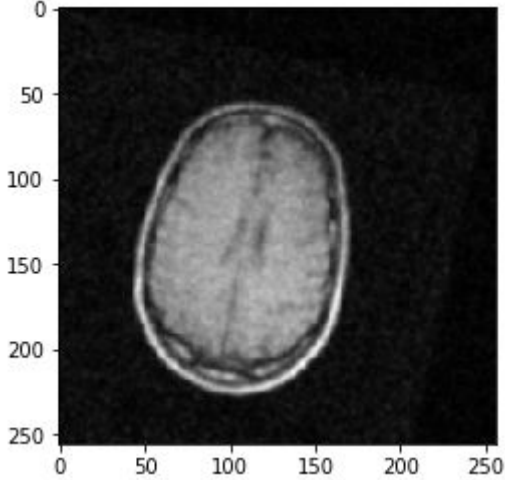
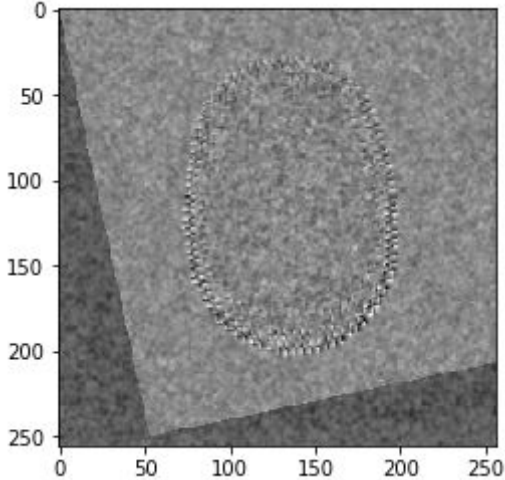
D'après l'image ci-dessus ainsi que la SSD obtenue, ce recalage n'est pas du tout satisfaisant.
Sa courbe d'énergie est :

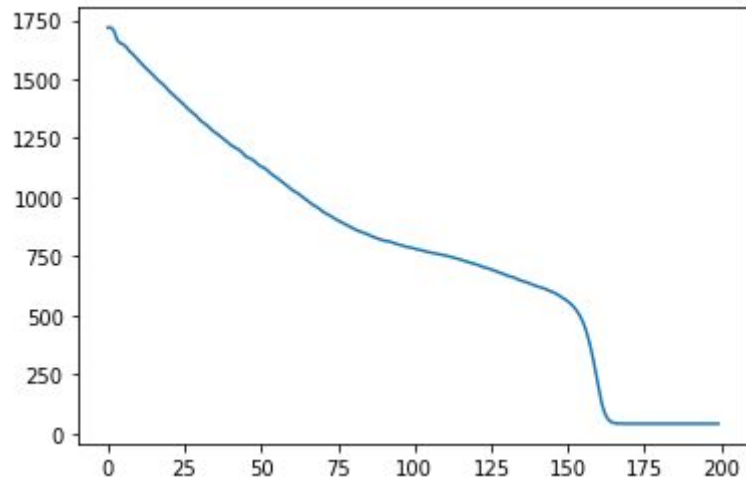


Cette dernière ne paraît pas convergente. Si on augmente le nombre d'itérations, on obtient :



On peut confirmer avec cette courbe énergétique que ce recalage **ne converge pas**.

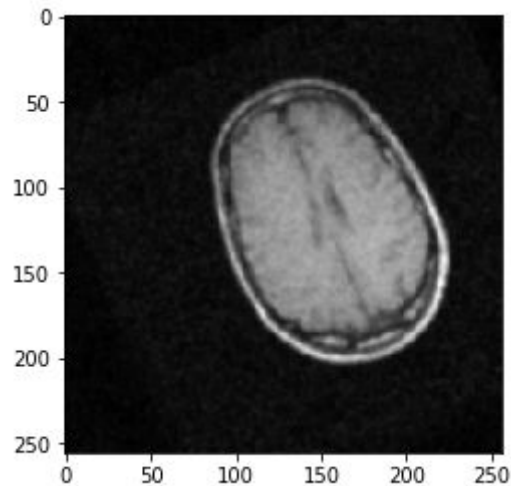
	<p>Une raison possible pour cette divergence est que la descente de gradient a rencontré un minimum local qu'elle n'a pas pu dépasser à cause de son pas ϵ fixe.</p>
<p>BrainMRI_3 $\epsilon=0.00001$</p>	<p>L'image recalée qui a pour SSD 41.0 est :</p>  <p>En la comparant à l'image objectif BrainMRI_1, on obtient :</p>  <p>D'après l'image ci-dessus ainsi que la SSD obtenue, ce recalage est très satisfaisant. Sa courbe d'énergie est :</p>



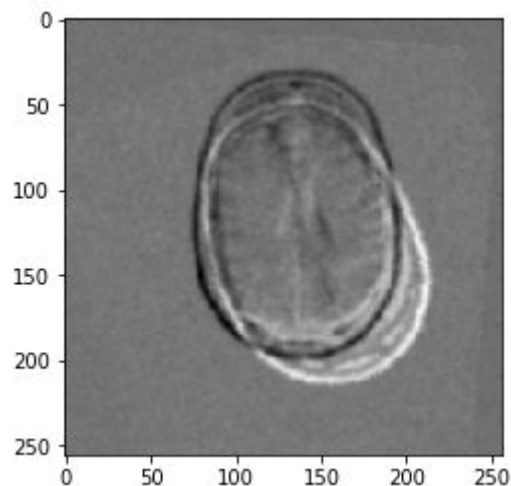
On peut confirmer avec la courbe énergétique que ce recalage **converge**.

BrainMRI_4
 $\epsilon=0.000014$

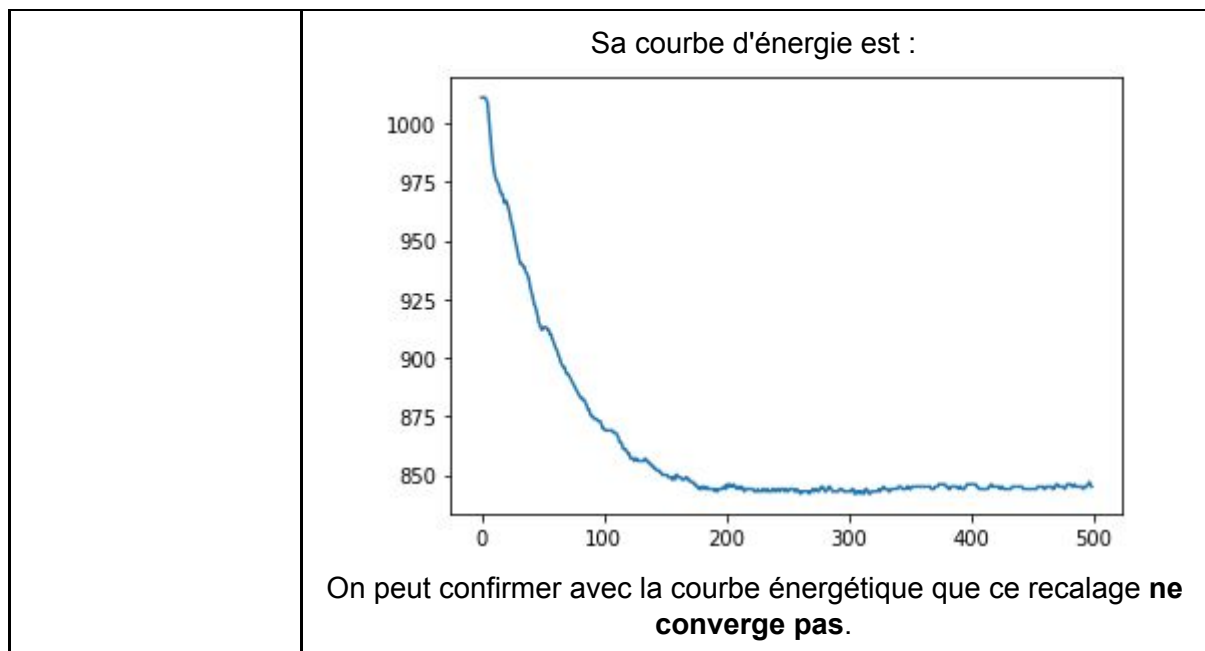
L'image recalée qui a pour SSD 845.0 est :



En la comparant à l'image objectif BrainMRI_1, on obtient :



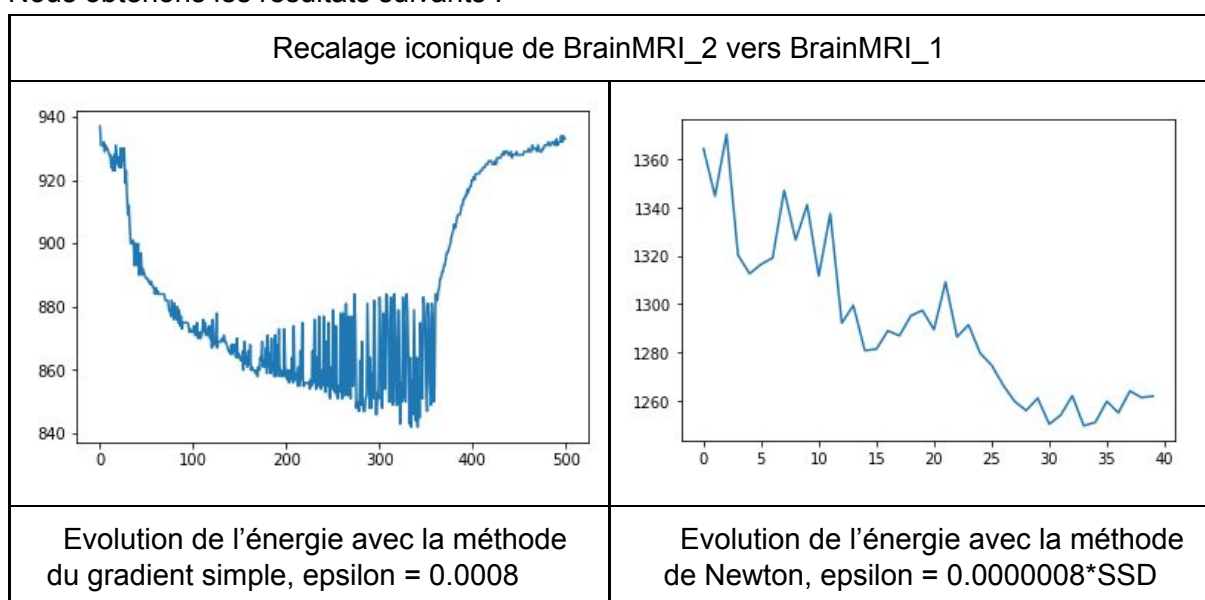
D'après l'image ci-dessus ainsi que la SSD obtenue, ce recalage n'est pas satisfaisant.

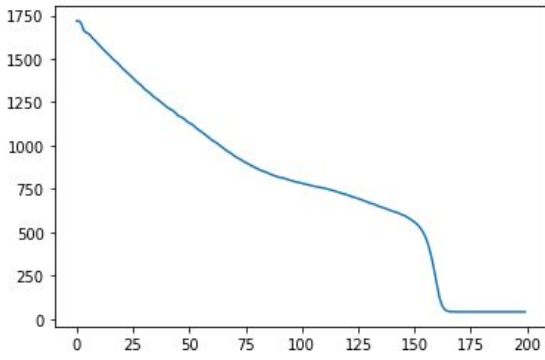
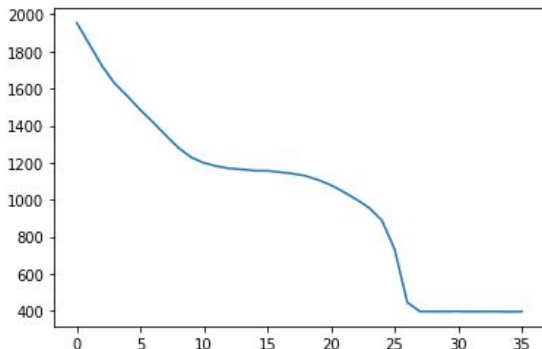


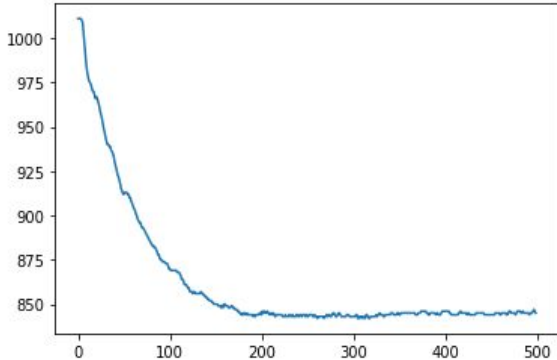
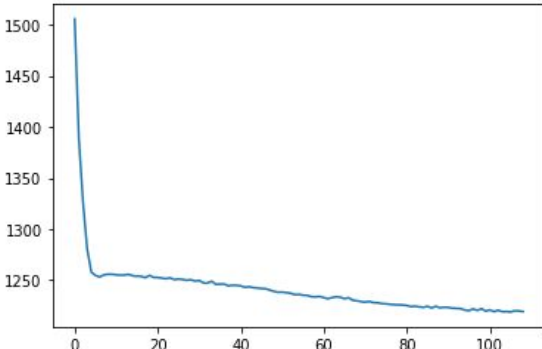
Qi. Optimisation

Nous avons choisi d'utiliser la méthode de Newton pour optimiser la descente de gradient. Concrètement nous rendons epsilon proportionnel à la SSD. Ainsi notre epsilon évolue désormais à chaque itérations, et devient plus petit lorsque la qualité du recalage augmente. Notre SSD variant entre l'ordre 10^3 et 10^2 il a fallu abaisser la précédente valeur de epsilon (plus exactement, le coefficient qui multiplié par la SSD donne epsilon) pour éviter que celui-ci soit trop important et que notre fonction ne parviennent pas à recalier.

Nous obtenons les résultats suivants :



Recalage iconique de BrainMRI_3 vers BrainMRI_1	
	
Evolution de l'énergie avec la méthode du gradient simple, $\epsilon = 0.00001$	Evolution de l'énergie avec la méthode de Newton, $\epsilon = 0.00000005 \cdot \text{SSD}$
<p>On remarque que la convergence s'effectue plus rapidement. On a également ajouté un critère d'arrêt qui se déclenche quand la différence entre l'énergie actuelle et celle calculée 10 itérations avant est inférieure à 1.</p>	

Recalage iconique de BrainMRI_4 vers BrainMRI_1	
	
Evolution de l'énergie avec la méthode du gradient simple, $\epsilon = 0.000014$	Evolution de l'énergie avec la méthode de Newton, $\epsilon = 0.00000014 \cdot \text{SSD}$
<p>Dans ce cas, nous avons testé plusieurs valeurs de epsilon mais la méthode de Newton ne permet pas d'améliorer significativement les résultats. Nous pensons que cela est dû au fait que la SSD varie peu. Rendre l'epsilon proportionnel à la SSD change alors peu de chose. Cette technique pourrait être efficace sur les images dont la SSD évolue fortement, comme des images translatées.</p>	

Q5 - Préparation TP3

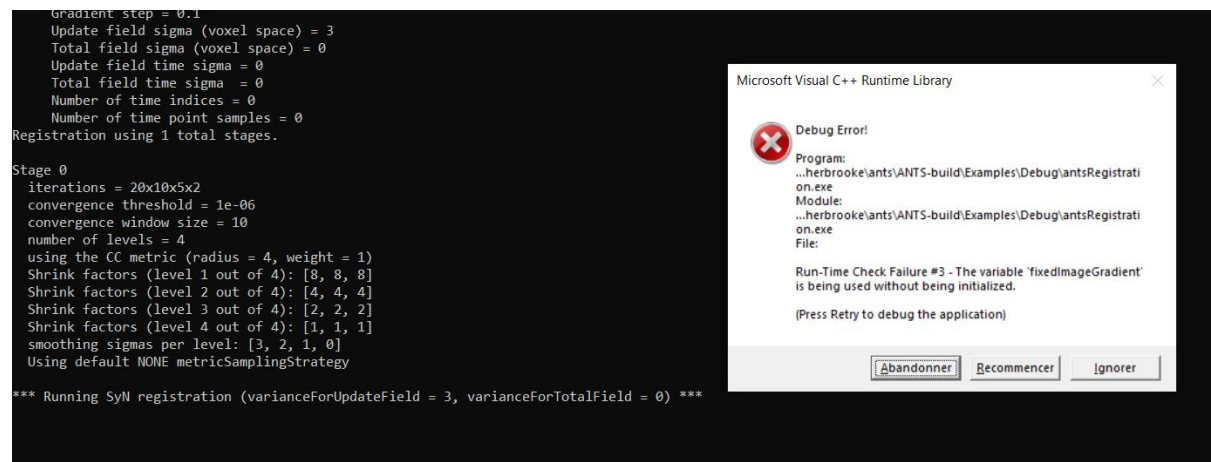
ANTs est un logiciel open source qui va nous permettre d'effectuer un recalage linéaire ou bien non linéaire. Il y a également d'autres fonctionnalités tels que des transformations, des seuillages etc...

Pour effectuer un recalage on utilise le script antsRegistration à l'aide de cette commande

```
.\Sherbrooke\ants\ANTS-build\Examples\Debug\antsRegistration -d 3 --float 0 --interpolation Linear -t SyN[0.1, 3, 0] -m CC[D:\Sherbrooke\ImagerieMedicale-TP1\Data_MiseEnForme\IRM\Brain\t1.nii, D:\Sherbrooke\ImagerieMedicale-TP1\Data_MiseEnForme\IRM\Brain\flair.nii, 1, 4] -o T1toFlairSyN -f 8x4x2x1 -c [20x10x5x2, 1e-6,10] --verbose 1 -smoothing-sigmas 3x2x1x0vox
```

On peut voir qu'on peut alors changer l'interpolation, ici elle est linéaire. On a aussi accès à différents recalage, linéaire avec des transformations Affine et Rigide mais aussi non linéaire avec SyN qu'on a voulu utiliser ici. Il y aussi différents métriques pour pouvoir optimiser le recalage. On peut voir qu'ici le métrique utilisé est CC correspondant au coefficient de corrélation.

Malheureusement, tout ne s'est pas passé comme prévu :



Nous avons eu un problème lors de l'exécution du fichier et n'avons pas pu aller plus loin.

En installant le sous-système linux et en rebuildant ANTs avec nous n'avons pas pu aller jusqu'à cette étape, le sous-système ubuntu ne semble pas se comporter exactement comme prévu. Nous n'avons donc pas pu résoudre le problème.

Ayant eu ces problèmes avec nous avons demandé à une autre équipe de fournir des images correspondants à ces paramètres et ainsi les analyser. Pour les obtenir il faut donc aussi utiliser antsApplyTransforms.

Ici à gauche nous avons un recalage linéaire avec : interpolation linéaire, information mutuelle et recalage rigide. A droite un recalage non linéaire qui correspond à la commande que nous avons tenté.

Nous n'allons donc pas pouvoir répondre à la question des meilleurs paramètres vu que nous n'avons que cet échantillon, mais nous allons voir si le non linéaire améliore le recalage.

Nous pouvons voir que sur le résultat du linéaire les formes semblent bien retranscrites. Il y a des angles en trop mais les traits semblent dans la bonne direction, c'est un résultat assez satisfaisant. **Le non linéaire ne semble pas être plus efficace** que le linéaire. En effet les formes sont plus floues et l'algorithme a tendance à trop arrondir. Certaines formes sont plus épaisses qu'elles ne devraient l'être. Il y a aussi des angles assez distincts comme le linéaire. Certaines formes ont aussi tendance à être trop arrondies. Néanmoins le linéaire et le non linéaire ont tous deux du mal à bien former les espaces noirs qui se retrouvent trop souvent resserrés.



Image t1.nii à la 70ème tranche

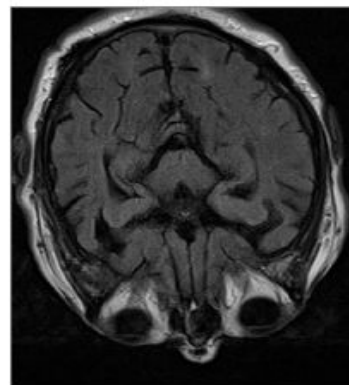


Image flair.nii à la 9ème tranche

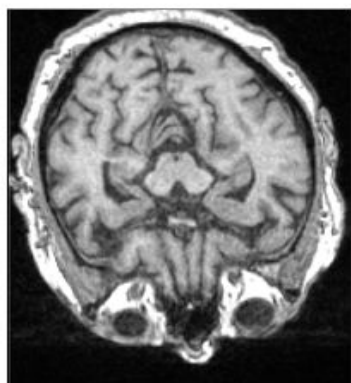


Image nii recalée sur flair (transformation rigide 9ème tranche)

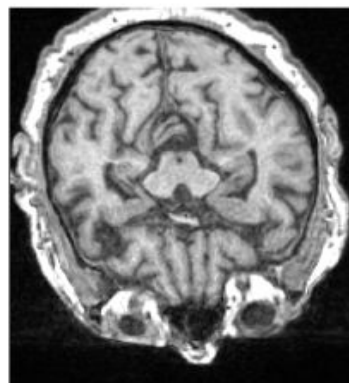


Image nii recalée sur flair (transformation non linéaire) à la 9ème tranche