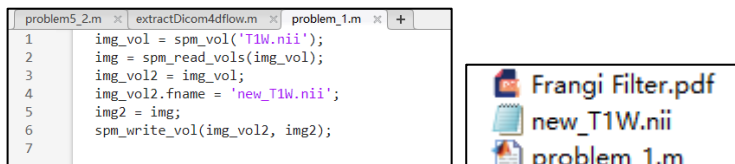


请完成以下练习，并给出文字回答或运行结果截图。在下一次课之前，将带有你的结果的文档上传到 elearning（文档命名规则为：姓名-学号-2022xxxx.docx）。

- 下面练习使用 SPM 生成一个新的 NIFTI 文件：使用以下句子将 3 Exercise 中的“T1W.nii”加载到 Matlab 中，并生成一个新的 NIFTI 文件（仅命名与原文件不一样，数据完全一致）

```
img_vol = spm_vol('T1W.nii');
img = spm_read_vols(img_vol);
img_vol2 = img_vol;
img_vol2.fname = 'new_T1W.nii';
img2 = img;
spm_write_vol(img_vol2, img2);
```



- 对 `img` 进行裁减操作，例如取出部分区域 `img_crop = img(50:150, 100:250, 30:120)`，并创建相应的 NIFTI 文件头，使用 `spm_write_vol` 生成一个新的 NIFTI 文件（假设命名为“cropped_T1W.nii”）。使用 `spm_reslice` 函数，将“cropped_T1W.nii”插值到“T1W.nii”，并显示，以验证所创建的 NIFTI 文件头是否正确。另外，请也生成将图像左右翻转后的 NIFTI 文件，即 `img_flip = img(:, end:-1:1, :)`？（提示：创建文件头时，只需修改原文件头中的 `dim` 和 `mat` 域。在创建文件头的 `mat` 域时，由于已知原图像和裁减后的图像的像素对应关系，可以基于以下公式推导计算出新仿射变换矩阵（即 `mat`）各元素的数值）

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} I_1 \\ J_1 \\ K_1 \\ 1 \end{bmatrix} = \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} I_2 \\ J_2 \\ K_2 \\ 1 \end{bmatrix} = \text{World coordinate}$$

- (1) 裁剪图像，保存为 `cropped_T1W.nii`，旋转图像保存为 `invert_cropped_T1W.nii`

```
1 % 使用 img_vol 和 img 分别读取文件头和像素矩阵
2 img_vol = spm_vol('T1W.nii');
3 img_data = spm_read_vols(img_vol);
4
5 % 裁剪图像
6 img_data_crop = img_data(50:150, 100:250, 30:120);
7
8 % 修改新的文件的文件头
9 img_vol_crop = img_vol;
10 img_vol_crop.fname = 'cropped_T1W.nii';
11 img_vol_crop.dim = size(img_data_crop);
12
13 % 计算新的 mat 矩阵，已知图像没有旋转和缩放，不考虑裁剪，相当于 x y z 方向移动 -49, -99, -29
14 trans_mat = [1, 0, 0, -49;
15              0, 1, 0, -99;
16              0, 0, 1, -29;
17              0, 0, 0, 1];
18 img_vol_crop.mat = img_vol.mat * inv(trans_mat);
19 img_vol_crop.mat
20
21 % 保存文件
22 spm_write_vol(img_vol_crop, img_data_crop);
```

```

24 % 反转图像-----
25 img_data_crop_invert = img_data_crop(:, end:-1:1, :);
26
27 % 修改新的文件的文件头
28 img_vol_crop_invert = img_vol_crop;
29 img_vol_crop_invert.fname = 'invert_cropped_T1W.nii';
30 img_vol_crop_invert.dim = size(img_data_crop_invert);
31
32 % 计算新的mat矩阵，相当于沿着y轴缩放-1倍，然后向y正方向平移152个单位
33 % 至于为啥是152不是151，我暂时也没弄明白。但确实这里写152比151好
34 trans_mat = [1,0,0,0;
35              0,-1,0,152;
36              0,0,1,0;
37              0,0,0,1];
38 img_vol_crop_invert.mat = img_vol_crop.mat * inv(trans_mat);
39 img_vol_crop_invert.mat
40
41 %保存文件
42 spm_write_vol(img_vol_crop_invert, img_data_crop_invert);

```

至于为什么是 152，猜想可能是 1 变-1，需要移动 2 个单位。151 变-151，-151 需要移动 152 才能到 1 的位置。

(2) 验证仿射变换矩阵是否正确

```

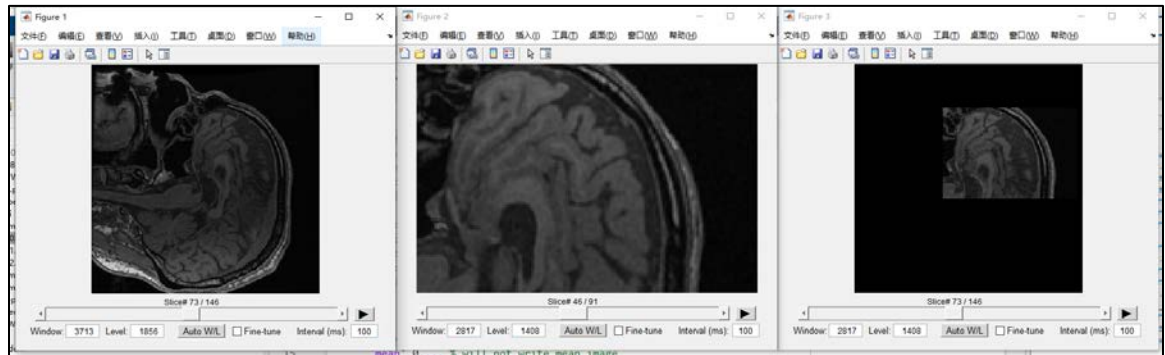
1  ref_vol = spm_vol('T1W.nii');
2  ref_vol.mat
3  ref_vol_data = spm_read_vols(ref_vol);
4  figure, imshow3D(ref_vol_data);
5
6  to_reslice_file = 'cropped_T1W.nii';
7  to_reslice_vol = spm_vol(to_reslice_file);
8  to_reslice_vol.mat
9  to_reslice_vol_data = spm_read_vols(to_reslice_vol);
10 figure, imshow3D(to_reslice_vol_data);
11
12 prefix = 'resliced_';
13 resflags = struct(...
14     'mask',0,... % will not mask anything
15     'mean',0,... % will not write mean image
16     'which',1,... % write only the coregistered file
17     'interp',1,...
18     'prefix', prefix);
19 spm_reslice([ref_vol; to_reslice_vol], resflags);
20
21 reslice_vol = spm_vol(['resliced_',to_reslice_file]);
22 reslice_vol.mat
23 reslice_vol_data = spm_read_vols(reslice_vol);
24 figure, imshow3D(reslice_vol_data);

```

左：原始图像

中：裁剪的图像

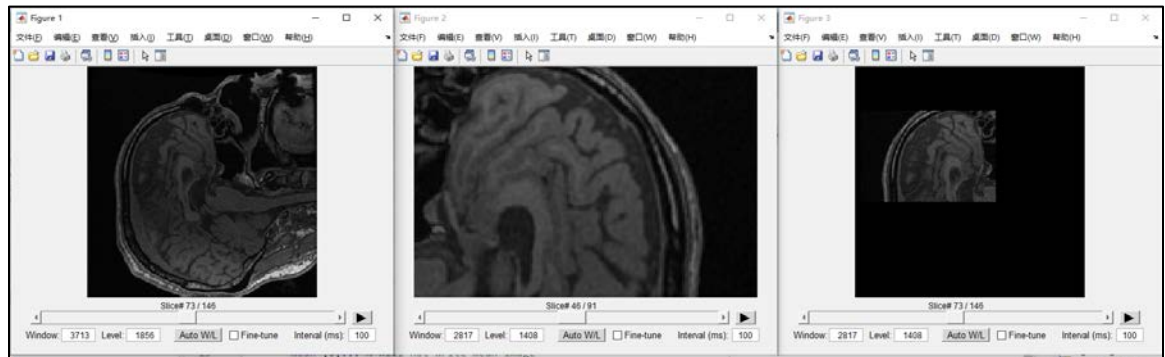
右：插值后的图像



左：原始翻转图像

中：裁剪的图像

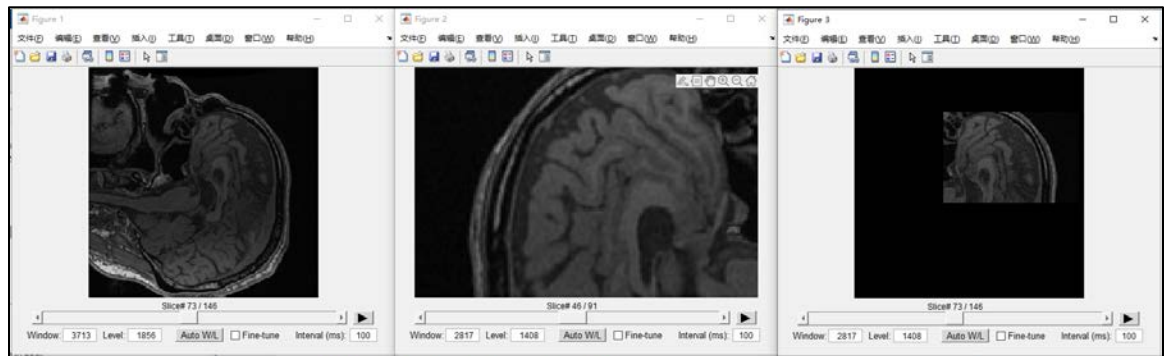
右：插值后的图像


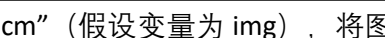



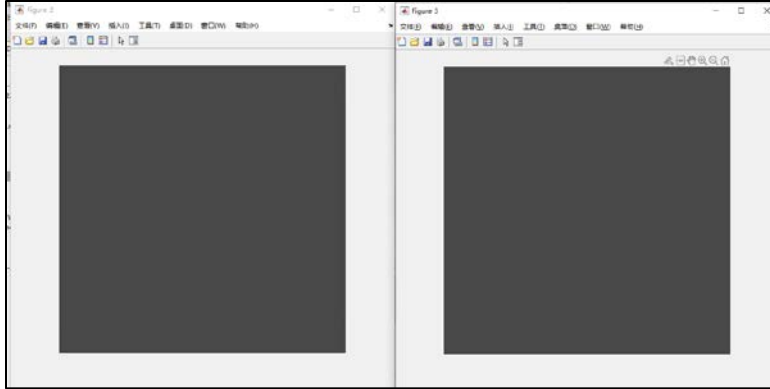
左：原始图像

中：翻转裁剪的图像

右：插值后的图像

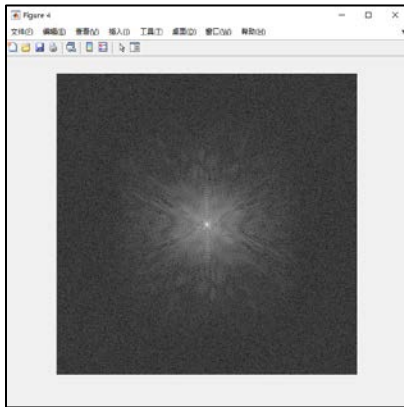


3. 读取“T2W_Head.dcm”（假设变量为），将图像数值归一化到[0, 1]。使用语句添加图像噪声。
 - a. 使用 `fft2` 函数计算的傅立叶变换，并显示（注：为了让图像的频谱图的低频处于图像中心，需要使用 `fftshift` 对 `fft2` 的结果进行位置调整）

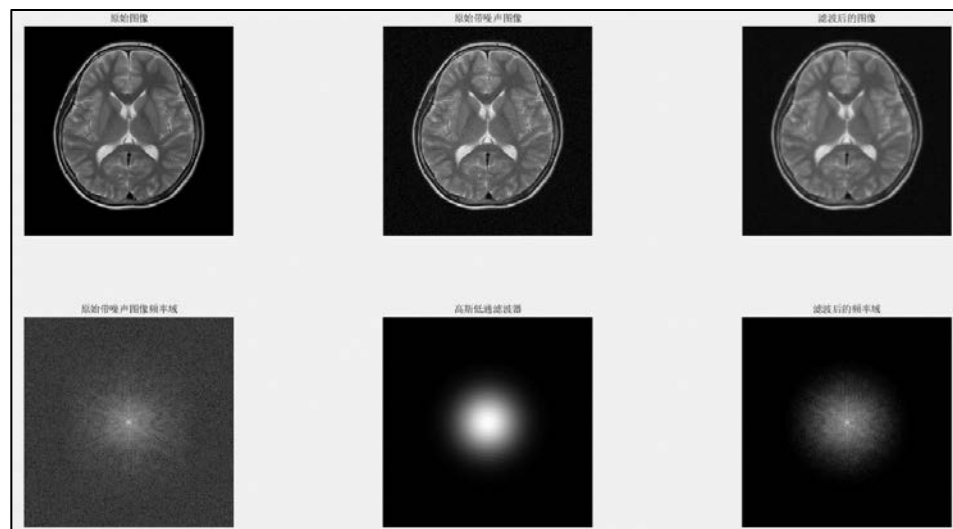
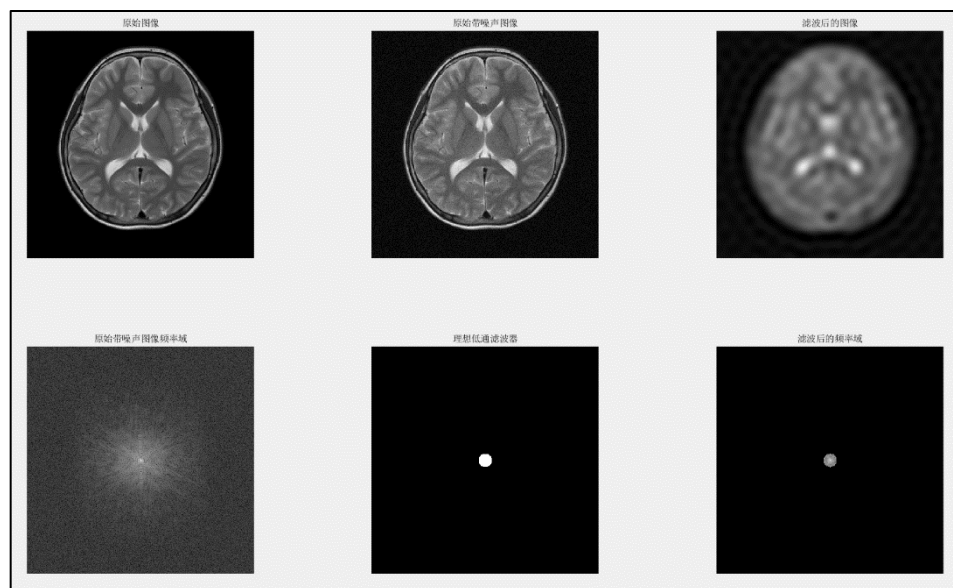
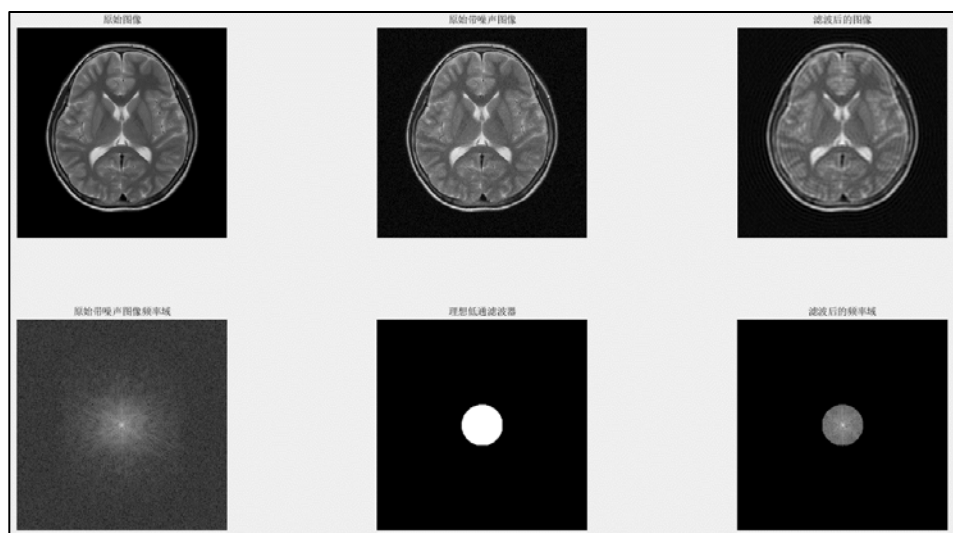


左：傅里叶变换 右：傅里叶变换并 *fftshift*，图中心有个白点

- b. 假设 *fft2* 和 *fftshift* 后的结果为 F_c ，对 F_c 做对数变换（即 $\log(1+\text{abs}(F_c))$ ），并显示



- c. 选取一定频率域半径值，设计一个频率域理想低通滤波器，用于对 *img_noise* 平滑去噪；设计一个频率域高斯低通滤波器，用于对 *img_noise* 平滑去噪。将 *img*，*img_noise* 和平滑后的图像并排显示。



可以观察到，滤波器半径越小，高频信息丢失越多，图像越模糊。使用理想低通滤波器，会出现振铃效应。使用高斯低通滤波器可以避免这个现象。

4. 通过阅读论文“**Multiscale vessel enhancement filtering**”，了解 Frangi vesselness 滤波器，并简述其数学原理。Matlab 自带函数 `fibermetric` 实现了 Frangi 滤波器，请使用该函数对“2 Exercise.docx”中的 TOF 图像的最大强度投影图（沿第三维）进行血管增强，请深度使用不同的 `thickness`（见 `fibermetric` 函数的说明）（注：在使用 `fibermetric` 之前，建议将图像的数值归一化到 $[0, 1]$ ，否则需要自己指定“`StructureSensitivity`”参数的数值。Frangi 滤波器的另一个实现：

<https://ww2.mathworks.cn/matlabcentral/fileexchange/24409-hessian-based-frangi-vesselness-filter>）。

基本原理：通过计算空间中任一点的二阶偏导数，Hessian 矩阵是由二阶偏导数组成。

$$H = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{yx} & I_{yy} \end{bmatrix}$$

根据数学上极最值点的判定规则和 hessian 矩阵三个特征值的正负性和大小关系，判断当前空间点是球、管状、平面、噪声的可能性。通过设置滤波器的半径，可以选择增强更粗的血管还是更细的血管。

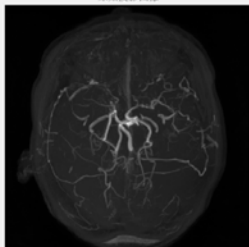
2D		3D			orientation pattern
λ_1	λ_2	λ_1	λ_2	λ_3	
N	N	N	N	N	noisy, no preferred direction
		L	L	H-	plate-like structure (bright)
		L	L	H+	plate-like structure (dark)
L	H-	L	H-	H-	tubular structure (bright)
L	H+	L	H+	H+	tubular structure (dark)
H-	H-	H-	H-	H-	blob-like structure (bright)
H+	H+	H+	H+	H+	blob-like structure (dark)

Table 1. Possible patterns in 2D and 3D, depending on the value of the eigenvalues λ_k (H=high, L=low, N=noisy, usually small, +/- indicate the sign of the eigenvalue). The eigenvalues are ordered: $|\lambda_1| \leq |\lambda_2| \leq |\lambda_3|$.

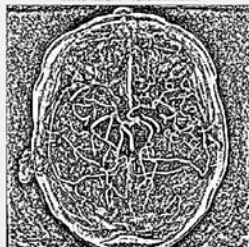
特征值大小与正负性对应可能的形状、亮血、黑血信号。

实验结果在下一页

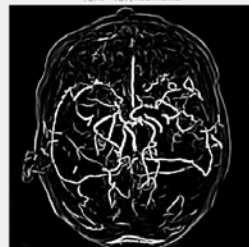
原始投影图像



默认参数无归一化的fibermetric



无归一化的fibermetric



修改thickness = [1,2,3,4,5]



修改thickness = [16,18,20,22,24]

