Ex1

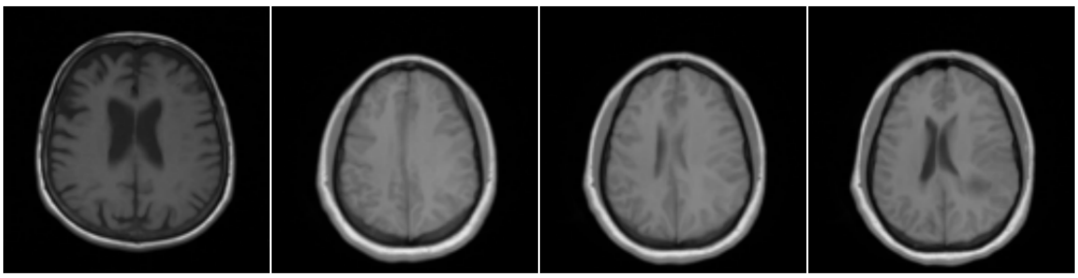
To compute the angle between image pairs, cosine similarity is first computed as $sim = cos(\theta) = \dfrac{u \cdot v}{\|u\|\|v\|}$, then the angle is $\theta = arccos(sim)$.

```
import numpy as np
import os
import glob
import cv2
import math

# exercise 1
root_dir = "BIOE580_hw01_data"
image_dir = "hw01_ex01_fastMRI-T1-brain-slices"
images = glob.glob(os.path.join(root_dir, image_dir+"/*.png"))

print(",".join(images)+"\n")
for img_a in images:
    frame_a = cv2.imread(img_a)
    # normalize image
    frame_a = frame_a / 255
    print("{}:".format(img_a), end="")
    vector_a = np.ravel(frame_a)
    for img_b in images:
        frame_b = cv2.imread(img_b)
        frame_b = frame_b / 255
        vector_b = np.ravel(frame_b)
        # compute cosine similarity
        cos_sim = np.inner(vector_a, vector_b) / (np.linalg.norm(vector_a) * np.linalg.norm(
            vector_b))
        angle = np.arccos(cos_sim) / math.pi * 180
        print("{}/{},".format(cos_sim, angle), end="")
    print("\n")
```

The sample images are shown as Fig. 1 and the results are shown in the following table. We can observe that when two images are more visually similar, the angle between them is smaller, like image 62631 and 62626.



62571 **62618** **62626** **62631**

Figure 1: FastMRI examples

Image ID	62571	62631	62618	62626
62571	1.00	0.76	0.77	0.76
62831	0.76	1.00	0.89	0.97
62618	0.77	0.89	1.00	0.96
62626	0.76	0.97	0.96	1.00

Table 1: Cosine similarity between image pairs.

Image ID	62571	62631	62618	62626
62571	0.00	40.56	39.88	40.12
82831	40.56	0.00	27.55	14.99
62618	39.88	27.55	0.00	17.01
62626	40.12	14.99	17.01	0.00

Table 2: Angles between image pairs.

Ex2

a. Operator H is a 16384×16384-dimensional matrix which maps images from 16384-dimensional space to objects in another 16384-dimensional space.

b. $\mathbf{H} : \mathbb{R}^{16384} \rightarrow \mathbb{R}^{16384}$

c. Because $H = H^T$, so H is symmetric matrix. Also, $det(H) = 0$, which means H is singular matrix. H is not full rank and nullity > 0. Therefore, H has null space.

Ex3

a. For $\mathbb{C}^2$, its standard basis vectors are $\{(1,0),(0,1)\}$.

$S_1\begin{pmatrix}1&0\\0&1\end{pmatrix}=\mathcal{B}_1\rightarrow S_1=\mathcal{B}_1=\begin{pmatrix}3&4\\4&-3\end{pmatrix}$

Similarly, the standard basis vectors of $\mathbf{C}^3$ are $\{(1,0,0),(0,1,0),(0,0,1)\}$.

$$S_2=\mathcal{B}_2=\begin{pmatrix}1&2&2\\6&3&-6\\6&-6&3\end{pmatrix}$$

b. $T\begin{pmatrix}x\\y\end{pmatrix}=\begin{pmatrix}x+yi\\x-yi\\2x\end{pmatrix}$. Easy to know: $T=\begin{pmatrix}1&i\\1&-i\\2&0\end{pmatrix}$

$$T_{new}=S_2^{-1}TS_1=\begin{pmatrix}-0.6-44/15i&-0.8+2.2i\\1.4+2.4i&28/15-1.8i\\0.4+16/15i&8/15-0.8i\end{pmatrix}$$

c. $v=T_{new}u=\begin{pmatrix}-2.8-56/15i\\3.2+64/15i\\1.2+1.6j\end{pmatrix}$

d. $u_e=S_1u=\begin{pmatrix}3+4j\\4-3j\end{pmatrix}$

$$v_e=S_2v=\begin{pmatrix}6+8j\\0\\6+8j\end{pmatrix}$$

We compute $T(u_e)=TS_1u=\begin{pmatrix}6+8j\\0\\6+8j\end{pmatrix}=v_e$

Ex4

According to Euler's equation,
$f(N) = \sum_{n=0}^{N-1} a_n(\frac{exp(in\pi x) - exp(-in\pi x)}{2i}) + b_n(\frac{exp(in\pi x) + exp(-in\pi x)}{2})$
$= \sum_{n=0}^{N-1} \frac{b_n - ia_n}{2} exp(in\pi x) + \frac{b_n + ia_n}{2} exp(-in\pi x)$
$f(N) = \overline{f(N)}$. Since $u_n = \frac{1}{\sqrt{2}} exp(in\pi x)$, then $f(N) = \sum_{n=0}^{N-1} \frac{b_n - ia_n}{2}\sqrt{2}u_n + \frac{b_n + ia_n}{2}\sqrt{2}\overline{u_n}$
$= c_1 u_n + c_2 \overline{u_n}$
So f(N) is a linear combination of u_n and its conjugate $\overline{u_n}$. So it seems that the dimension of $f(N)$ is 2 and u_n could be a basis of f(N).
For u_n, its inner product is:
$< u_n, u_n >= \int_{-1}^{1} \frac{1}{\sqrt{2}} exp(-in\pi x) \frac{1}{\sqrt{2}} exp(in\pi x) dx = \int_{-1}^{1} \frac{1}{2} dx = 1.$
So u_n is orthornormal basis.

5

If $< \mathbf{f}, A\mathbf{g} >=< A\mathbf{f}, \mathbf{g} >$, then A is Hermitian.

For $A = iD$

$$< A\mathbf{f}, \mathbf{g} >= \int_{-1}^{1} \overline{Af} g dx = \int_{-1}^{1} (-i\frac{d}{dx}\overline{f}) g dx = -i\int_{-1}^{1} g d\overline{f}$$

$$< \mathbf{f}, A\mathbf{g} >= \int_{-1}^{1} \overline{f} i \frac{d}{dx} g dx = i\int_{-1}^{1} \overline{f} dg$$

$$< A\mathbf{f}, \mathbf{g} > - < \mathbf{f}, A\mathbf{g} >= -i([g\overline{f}]_{-1}^{1} - \int_{-1}^{1} \overline{f} dg) - i\int_{-1}^{1} \overline{f} dg = -i[\overline{f}g]_{-1}^{1} = 0$$

So $< A\mathbf{f}, \mathbf{g} >=< \mathbf{f}, A\mathbf{g} >$, A is hermitian.

Ex6

Let $H = g(x', y')$,

$H(af_1(x,y)+bf_2(x,y)) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} sinc(x'-x)sinc(y'-y)(af_1(x,y)+bf_2(x,y))dxdy$

$= a\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} sinc(x'-x)sinc(y'-y)f_1(x,y)dxdy$

$+ b\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} sinc(x'-x)sinc(y'-y)f_2(x,y))dxdy$

$= aH(f_1(x,y)) + bH(f_2(x,y))$

So $H = g(x', y')$ is a linear operator.

Since sinc function is y-axis symmetric, $sinc(\theta) = \frac{sin(\theta)}{\theta} \rightarrow sinc(\theta)^{\dagger} = \frac{sin(-\theta)}{-\theta} = sinc(\theta)$

$H^{\dagger} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} sinc(x'-x)^{\dagger}sinc(y'-y)^{\dagger}f(x,y)dxdy$

$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} sinc(x'-x)sinc(y'-y)f(x,y)dxdy = H$. So it's Hermitian.

Its domain should be $\mathbb{R}^2(-\infty,\infty)$ and its range may be $\mathbb{C}^1$ if sinc(x'-x) or sinc(y'-y) is complex.

Ex7

a.

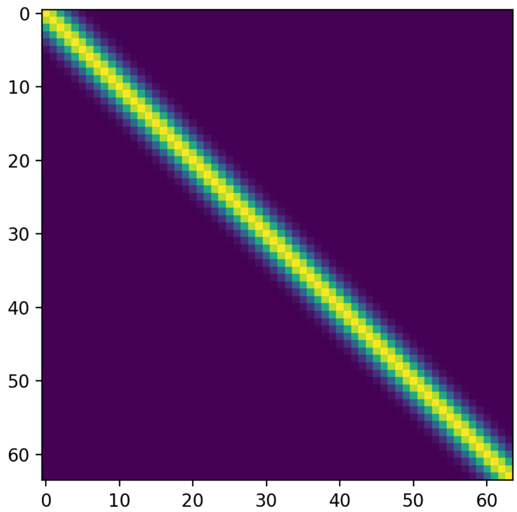


Figure 2: Heatmap of H when $\sigma = 8$

b. $rank(H) = 64$, so nullity(H)=64-64=0

c.

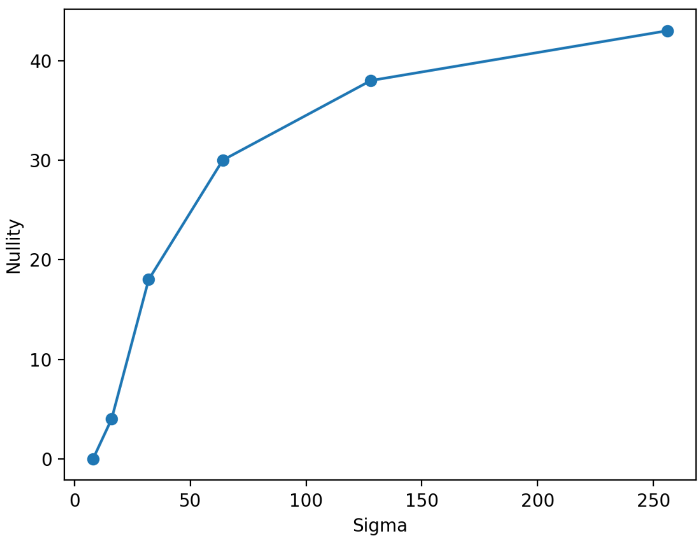


Figure 3: Heatmap of H when $\sigma = 8$

The nullity of operator H increases as σ increases. When σ increases sharply, the values which is closer to the diagonal line of H increases rapidly to 1. The column vectors become less orthogonal and more linearly dependent. So the rank of its column space decreases and the nullity increases.

Ex8

a. To check whether A is orthogonal, we compute $np.matmul(A, A.transpose())$ and the result shows that only diagonal values are close to 1 and the other values are trivial. So we can say A is still orthogonal.

b. Since A is nearly-orthogonal, P should be still nearly-orthogonal. Only diagonal values of P are about 1, so $\sum_r \sum_c P_{r,c} = 64$

c. $P_{null} \perp P \rightarrow < P_{null}, P >= 0$. Since P is nearly-identity matrix, we could simply set $P_{null} = I - P$.

d.

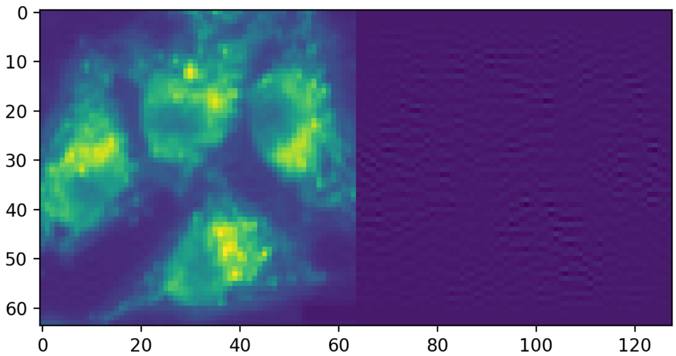


Figure 4: Projection result with P and P_{null}

When applying P, the objective is projected into object space while preserving its major feature information. When applying P_{null}, the objective is projected into the null space, losing almost all information.

e. Use the code show in Ex1, the cosine similarity between the images is 0, which means the angle is 90°. The result is straightforward since $P_{null} \perp P \rightarrow P_{null}P = 0$

f. Orthornormalization could make the basis orthogonal and normalized. On one hand, orthogonal matrix are computation efficiency which could keep the matrix sparse. On the other hand, normalization could keep matrix values within $[0, 1]$ boundary, which is important to make space closed when employing operator for multiple times. Without normalization, the P could grow extremely big so that the computer cannot handle it.