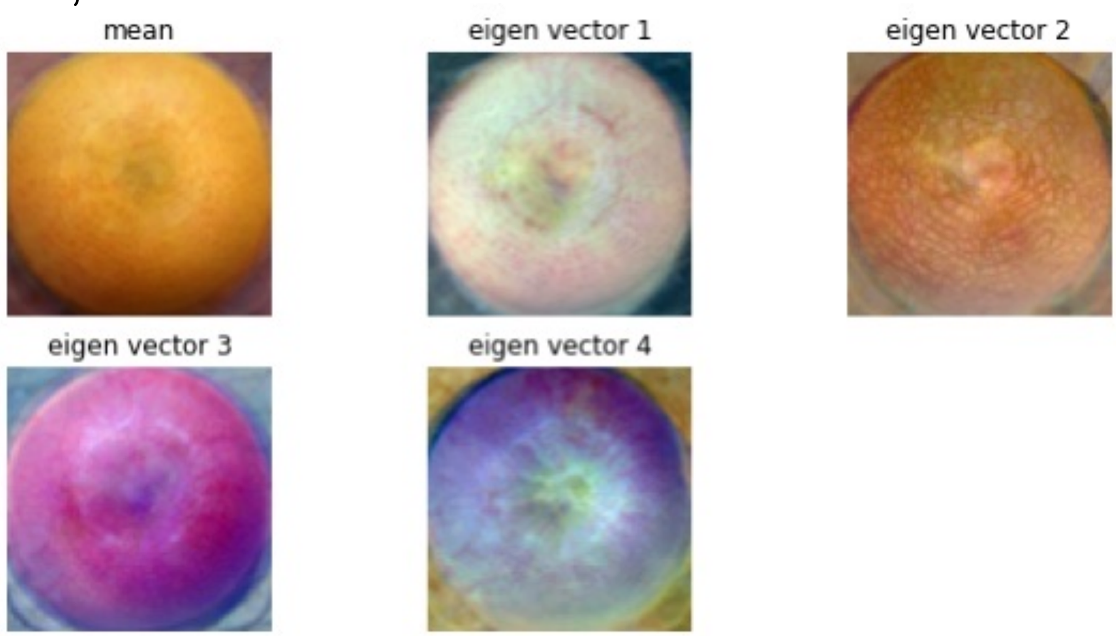
6. code ⇒ q6.py, q6.ipynb.

a. Similar to the analysis done in the prev. question

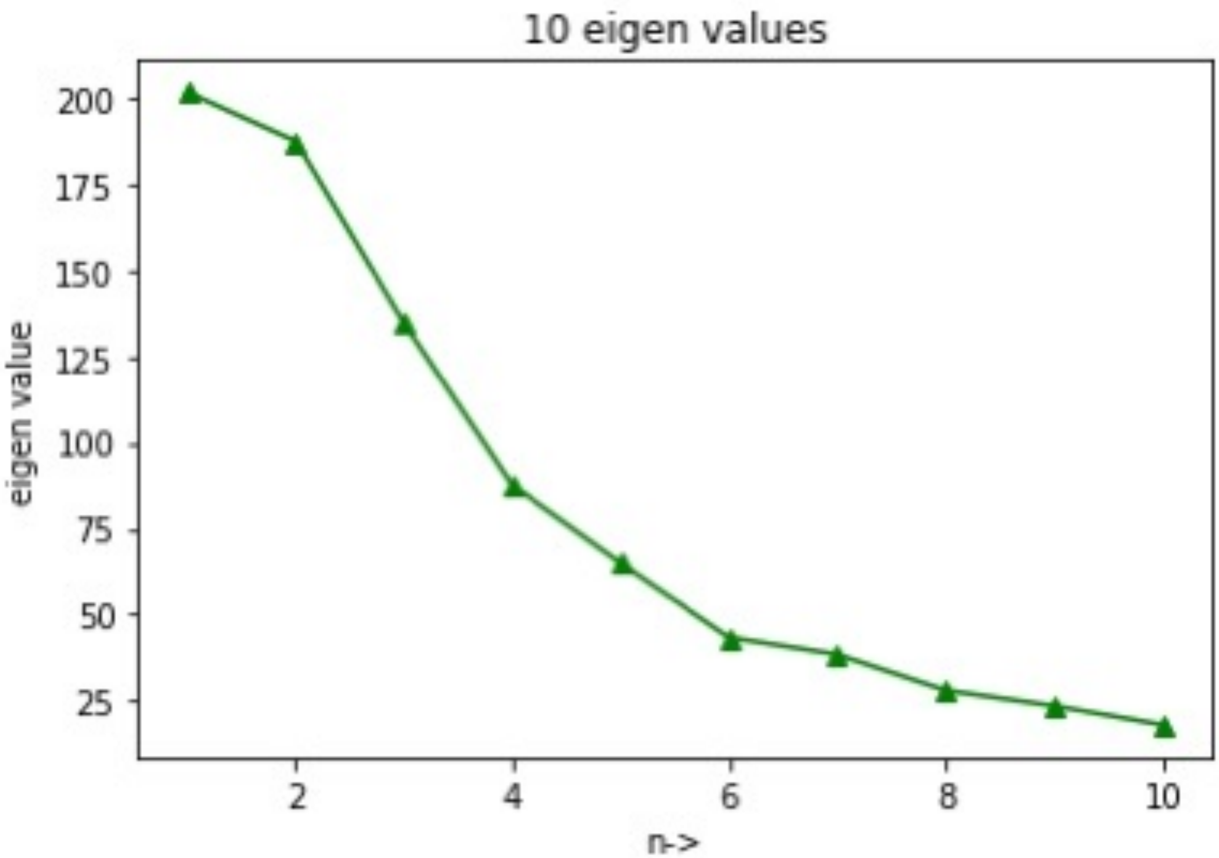
For this part we calculated then mean and covariance.

Also we calculated the top 4 principle eigen vectors and showed them as images in the same plot below

In the following plot the first image is for mean and the rest are for eigen vectors.



Also we have found the top 10 eigen values and plotted them in the following graph.



b. Let A be a vector image and

$x_1 v_1 + x_2 v_2 + x_3 v_3 + x_4 v_4$ be the closest representation and

$\| A - x_1 v_1 - x_2 v_2 - x_3 v_3 - x_4 v_4 \|_2$ be the norm of closeness

First we obtain that $\langle V_i\, V_j \rangle = 0$ if $i \neq j$;

We claim that

$$x_1 = \langle AV_1 \rangle$$

$$x_2 = \langle AV_2 \rangle$$

$$x_3 = \langle AV_3 \rangle$$

$$x_4 = \langle AV_4 \rangle$$

Best guess of a vector if we know the eigen components and the PCA reduction at that vector.

Definition :

$\langle AV \rangle = A^T V$ is like a dot product at A, V column vectors

Frobius Norm:

$\|e\|_2$ for one-column is same as frobius but

$$\|A\|_{Fro} = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} |a_{ij}|^2}$$

Let $\mu + x_1 v_1 + x_2 v_2 + \cdots + x_n v_n$ be the best approximation we can make by minimizing frobius norm where

$x_1, x_2, \ldots, x_n \subseteq R$

$v_1, v_2, \ldots, v_n$ are eigen vectors / principal directions.

$\mu \to$ mean.

→ Let A be the original vector

$\| A - (\mu + x_1 v_1 + x_2 v_2 + \cdots + x_n v_n)\|_2$ should be minimum.

We see that,

$\| A - (\mu + x_1 v_1 + x_2 v_2 + \cdots + x_n v_n)\|_{Fro} \cdot \| A - (\mu + x_1 v_1 + \cdots + x_n v_n)\|_{Fro}$

$= \text{trace}\left(\langle (A - \mu) - (x_1 v_1 + \cdots + x_n v_n), (A - \mu) - (x_1 v_1 + \cdots x_n v_n) \rangle \right)$

Observe that $\langle v_i \, v_j \rangle = 0$ [symmetric matrix ⇒ eigen vector are orthogonal]

$$= \text{trace}\Big(\langle A-\mu \;\; A-\mu \rangle + \sum_{i=1}^{n} x_i^2 \left(\langle V_i \; V_i \rangle\right) - x_i \langle V_i \; A-\mu \rangle$$

$$- x_i \langle A-\mu \;\; V_i \rangle \Big)$$

$$= \text{trace}\left(\langle A-\mu \; A-\mu \rangle + \sum_{i=1}^{n} x_i^2 \langle V_i V_i \rangle - 2x_i \langle V_i \; A-\mu \rangle \right)$$

$$= \text{trace}\left(\langle A-\mu \;\; A-\mu \rangle + \sum_{i=1}^{n} x_i^2 - 2x_i \langle A-\mu \, V \rangle \right)$$

should be minimum.

We can clearly see that for (1-D)

(1-column matrix) $\text{trace}(\langle AB \rangle) = \langle AB \rangle$ and for

$$\sum_{i=1}^{n} x_i^2 - 2x_i \langle V_i \, A-\mu \rangle$$ to be minimum

$$x_i = \langle V_i \;\; A-\mu \rangle = \langle A-\mu \; V_i \rangle$$

Hence the best approximation of A is

$$\mu_1 + \sum_{i=1}^{n} \langle A-\mu \;\; V_i \rangle V_i$$

where $\langle A - \mu \ V_i \rangle$ = i^{th} value of PCA analysis of A

$$PCA(A) = (A - \mu)^T \cdot V = J$$

$$\langle A - \mu \ V_i \rangle = J[0, i] \Rightarrow 1 \times n \text{ vector.}$$

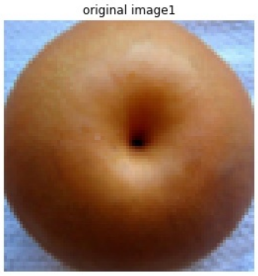
Best approximation gives PCA J is

$$\mu + \sum_{i=1}^{n} J[0, i] \, V_i$$

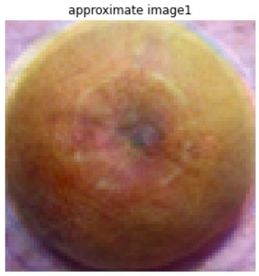
Hence we can see that PCA component which are determent by $\langle A \ V_i \rangle$ are the best approximation to that coordinates

$\Rightarrow$ The following are image retransformed compared to original ones.

original image1



approximate image1



original image2

approximate image2

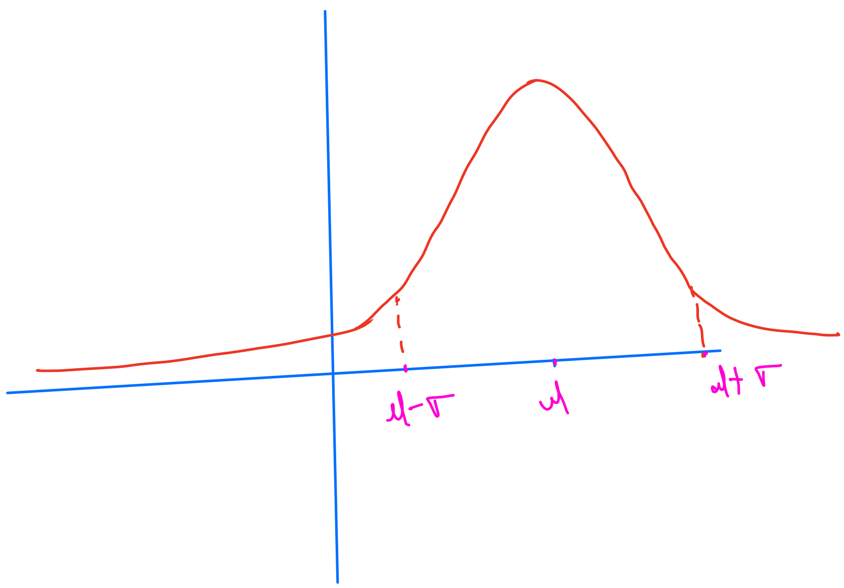

original image3
approximate image3
original image4
approximate image4
original image5
approximate image5
original image6
approximate image6
original image7
approximate image7
original image8
approximate image8
original image9
approximate image9
original image10
approximate image10

c. Generating random Images,

elf $+ k_1 v_1 + k_2 v_2 + k_3 v_3 + k_4 v_4$

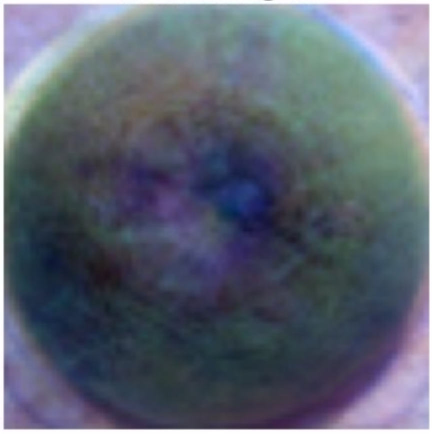
best random image has variance λ_1 in that axis.

It is best to have $-\sqrt{\lambda_1} < k_1 < \sqrt{\lambda_1}$ to get an image in an acceptable range because just like a gaussian has most of its probability inside the width of $\mu-\sigma$ to $\mu+\sigma$ where σ is the standard deviation similarly $\sqrt{\lambda_1}$ is like standard deviation.



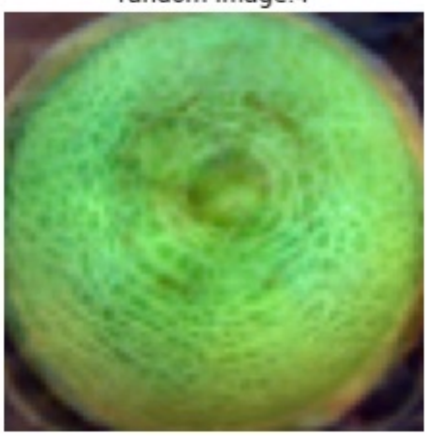
→ Using this algorithm in code q6.py to generate 4 random images.

random image:1



random image:2

random image:4



random image:3