PCA para IMAGEM

```
ave = plt.imread('passaro.jpg') #carrega a imagem

ave.shape
```

```
(740, 740, 3)
```

```
from PIL import Image
from matplotlib.image import imread

# Exibe a imagem
plt.imshow(ave)
plt.axis('off') # Para ocultar os eixos
plt.show()
```

```
ave = plt.imread('passaro.jpg') #carrega a imagem

R, G, B = ave[:,:,0], ave[:,:,1], ave[:,:,2]
ave = 0.2989 * R + 0.5870 * G + 0.1140 * B

# Exibe a imagem
plt.imshow(ave, cmap='gray')
plt.axis('off') # Para ocultar os eixos
plt.show()

ave.shape
```

```
(740, 740)
```

Comece a programar ou gere código com IA.

```
pca2 = PCA(n_components=256)
pca2.fit(ave)

plt.grid()
plt.plot(np.cumsum(pca2.explained_variance_ratio_ * 100))
plt.xlabel('Número de Componentes')
plt.ylabel('Variância Explicada')
```

Note que com 100 componentes, consegue-se explicar quase 100% da variância da imagem.

```
pca_50 = PCA(n_components = 100)
imagem_50 = pca_50.fit_transform(ave)
imagem_nova=pca_50.inverse_transform(imagem_50)
plt.imshow(imagem_nova, cmap='gray_r')
plt.axis('off')
plt.show()
```