SVD奇异值分解

1、特征值分解

若矩阵A是$n \times n$大小的实对称矩阵，那么它可以被分解为如下形式：

$$A = Q\Sigma Q^T = Q\begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix} Q^T$$

其中Q为标准正交阵（即有$QQ^T = I$），Σ为对角阵，且上面的矩阵的维度均为$n \times n$。λ_i称为特征值，q_i是Q（特征矩阵）的列向量，称为特征向量。

上面的特征值分解，对矩阵有着较高的要求，它需要被分解的矩阵A为实对称矩阵，但是现实中，我们所遇到的问题一般不是实对称矩阵。那么当我们碰到一般性的矩阵，即有一个$m \times n$的矩阵A，它是否能被分解成上面的式的形式呢？

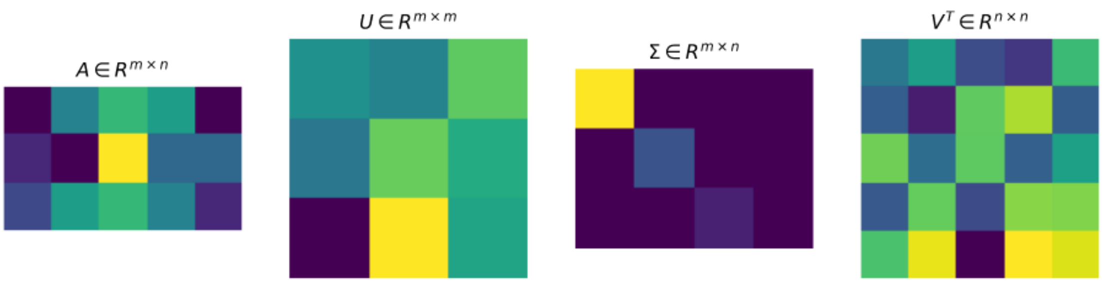
2、奇异值分解（SVD）

有一个$m \times n$的实数矩阵A，我们想要把它分解成如下的形式

\begin{equation} A = U\Sigma V^T \end{euation}

其中U和V均为单位正交阵，即有$UU^T = I$和$VV^T = I$，U称为左奇异矩阵，V称为右奇异矩阵，Σ仅在主对角线上有值，我们称它为奇异值，其它元素均为0。上面矩阵的维度分别为$U \in R_{m\times m}$, $\Sigma \in R_{m\times n}$, $V \in R_{n\times n}$。其中：

$$\Sigma = \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \ddots \end{bmatrix}$$



对于奇异值分解，我们可以利用上面的图形象表示，图中方块的颜色表示值的大小，颜色越浅，值越大。对于奇异值矩阵Σ，只有其主对角线有奇异值，其余均为0。

3、具体方法

$$AA^T = (U\Sigma V^T)(V\Sigma^T U^T) = U\Sigma\Sigma^T U^T$$
$$A^T A = (V\Sigma^T U^T)(U\Sigma V^T) = V\Sigma^T\Sigma V^T$$

- 求解U

通过AA^T的特征值分解可以得到U

- 求解V

通过A^TA的特征值分解可以得到U

- 求解奇异值

虽然$\Sigma\Sigma^T$和$\Sigma^T\Sigma$是不同维度的矩阵，但是它们在主对角线的奇异值是相等的。因此，可以直接对这两个矩阵中的特征值进行开方，可以得到所有的奇异值。

SVD示例1

In [1]:
```python
import numpy as np

A = np.array([
    [2,5,5,3,3],
    [2,1,6,7,2],
    [8,3,6,2,7]
])

AAT = np.dot(A,A.T)
ATA = np.dot(A.T,A)

uu,ssT,uuT = np.linalg.svd(AAT)

print('===== Calculate U by AAT =====')
print(uu)
print('===== Calculate Sigma bt AAT =====')
print(np.sqrt(ssT))

vv,sTs,vvT = np.linalg.svd(ATA)

print('===== Calculate VT by ATA =====')
print(vvT)
print('===== Calculate Sigma bt ATA =====')
print(np.sqrt(sTs))

u,sigma,vT = np.linalg.svd(A)

print('===== Calculate U by np.linalg.svd =====')
print(u)
print('===== Calculate VT by np.linalg.svd =====')
print(vT)
print('===== Calculate Sigma by np.linalg.svd =====')
print(sigma)
```

```
===== Calculate U by AAT =====
[[-0.47309358  0.16241173  0.86591275]
 [-0.50036649  0.75942783 -0.41581575]
 [-0.7251316  -0.62999349 -0.27801504]]
===== Calculate Sigma bt AAT =====
[16.63388983  6.26586351  3.47169461]
===== Calculate VT by ATA =====
[[-0.46579441 -0.30306977 -0.58425639 -0.38307993 -0.45064233]
 [ 0.51010827  0.05083002 -0.25354282 -0.72507852  0.38364442]
 [-0.38134873  0.88708923  0.04798205 -0.25031063 -0.05184746]
 [ 0.45390327  0.34231265 -0.64994202  0.44586121 -0.23574777]
 [ 0.4139687   0.03820698  0.41186599 -0.25702776 -0.769073  ]]
===== Calculate Sigma bt ATA =====
[1.66338898e+01 6.26586351e+00 3.47169461e+00 1.18774955e-07
 4.58547981e-08]
===== Calculate U by np.linalg.svd =====
[[-0.47309358  0.16241173  0.86591275]
 [-0.50036649  0.75942783 -0.41581575]
 [-0.7251316  -0.62999349 -0.27801504]]
===== Calculate VT by np.linalg.svd =====
[[-0.46579441 -0.30306977 -0.58425639 -0.38307993 -0.45064233]
 [-0.51010827 -0.05083002  0.25354282  0.72507852 -0.38364442]
 [-0.38134873  0.88708923  0.04798205 -0.25031063 -0.05184746]
 [ 0.15076263  0.26463732 -0.76931999  0.51369973  0.22692964]
 [-0.59554084 -0.220465    0.0143172  -0.03111406  0.77172095]]
===== Calculate Sigma by np.linalg.svd =====
[16.63388983  6.26586351  3.47169461]
```

SVD示例2

In [3]:

```
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import numpy as np

img_eg = mpimg.imread("1.jpg")
print(img_eg.shape)

img_temp = img_eg.reshape(img_eg.shape[0], img_eg.shape[1] * img_eg.shape[2])

U,Sigma,VT = np.linalg.svd(img_temp)

sval_nums = 20
img_restruct1 = (U[:,0:sval_nums]).dot(np.diag(Sigma[0:sval_nums])).dot(VT[0:sval_nums,:])
img_restruct1 = img_restruct1.reshape(img_eg.shape[0],img_eg.shape[1],img_eg.shape[2])

sval_nums = 60
img_restruct2 = (U[:,0:sval_nums]).dot(np.diag(Sigma[0:sval_nums])).dot(VT[0:sval_nums,:])
img_restruct2 = img_restruct2.reshape(img_eg.shape[0],img_eg.shape[1],img_eg.shape[2])

sval_nums = 180
img_restruct3 = (U[:,0:sval_nums]).dot(np.diag(Sigma[0:sval_nums])).dot(VT[0:sval_nums,:])
img_restruct3 = img_restruct3.reshape(img_eg.shape[0],img_eg.shape[1],img_eg.shape[2])

fig, ax = plt.subplots(1,4,figsize = (32,32))

ax[0].imshow(img_eg)
ax[0].set(title = "src")
ax[1].imshow(img_restruct1.astype(np.uint8))
ax[1].set(title = "nums of sigma = 30")
ax[2].imshow(img_restruct2.astype(np.uint8))
ax[2].set(title = "nums of sigma = 90")
ax[3].imshow(img_restruct3.astype(np.uint8))
ax[3].set(title = "nums of sigma = 180")
```

```
(899, 601, 3)
```

Out[3]:

```
[Text(0.5,1,'nums of sigma = 180')]
```

In [4]:
```python
fig = plt.figure(2)
plt.plot(Sigma)

per_sigma = Sigma
ss = 0
sum_sigma = np.sum(per_sigma)
for i in range(len(Sigma)):
    ss += per_sigma[i]
    per_sigma[i] = ss/sum_sigma

fig = plt.figure(3)
plt.plot(per_sigma)
```

Out[4]: [<matplotlib.lines.Line2D at 0x1e8b70877b8>]