m248Week7_Action_of_SVD

March 3, 2021

1 Let's use the singular value decomposition to explore the action of a matrix A on space. We will work with two dimensional matrices because they are easy to visualize.

$$A = \begin{pmatrix} 1 & 5 \\ -1 & 2 \end{pmatrix}$$

The singular value decomposition of A is

$$A = U\Sigma V^t = \begin{pmatrix} 0.93788501 & 0.34694625 \\ 0.34694625 & -0.93788501 \end{pmatrix} \begin{pmatrix} 5.41565478 & 0 \\ 0 & 1.29254915 \end{pmatrix} \begin{pmatrix} 0.10911677 & 0.99402894 \\ 0.99402894 & -0.10911677 \end{pmatrix}$$

is equivalent to

$$AV = U\Sigma,$$

which means that the action of A on the orthonormal columns of V is the same as stretching/squeezing the columns of U by the singular values. That is,

$$Av_1 = \sigma_1 u_1$$

and

$$Av_2 = \sigma_2 u_2$$

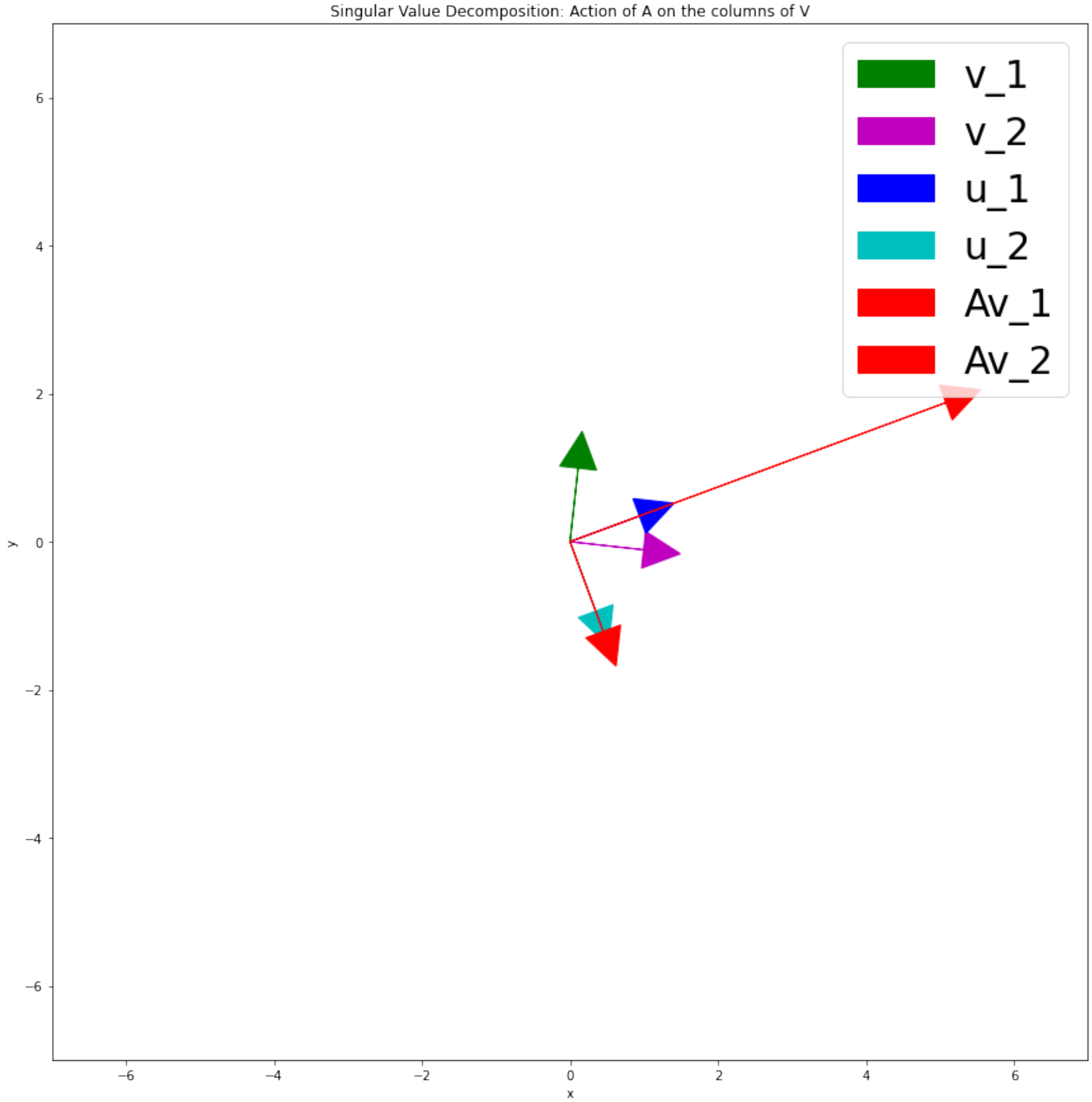
[44]:
```
import numpy as np
import matplotlib.pyplot as plt

# define A as a numpy array
A=np.array([[1,5],[-1,2]])


# perform SVD on A
U,sigma,Vt=np.linalg.svd(A)
print("U=\n",U)
print("sigma=\n",sigma)
# store sigma in a diagonal matrix that has the same shape as A
Sigma=np.diag(sigma)
print("Sigma=\n",Sigma)
print("Vt=\n",Vt)
```

```
# Check whether you can recover A
print('U.Sigma.Vt=\n',U.dot(Sigma.dot(Vt)))
print('A=\n',A)

# These are the columns of U
u_1=U[:,0]
u_2=U[:,1]

# These are the columns of V (not Vt, I transpose Vt first)
V=Vt.T
v_1=V[:,0]
v_2=V[:,1]

# This is A acting on the columns of V
Av_1=A.dot(v_1)
Av_2=A.dot(v_2)

# Plot the vectors using arrow in matplolib.pyplot.axes
# set the figure and labels
plt.figure(figsize=(15,15))
vec= plt.axes()
plt.axis('scaled') # the scale on the x-axis is the same as the y-axis
plt.xlim(-7,7)
plt.ylim(-7,7)
plt.title('Singular Value Decomposition: Action of A on the columns of V')
plt.xlabel('x')
plt.ylabel('y')

# plot the vectors as arrows
arrow_v_1=vec.arrow(0, 0, *v_1, head_width=0.5, head_length=0.5, color='g',␣
↪label='v_1')
arrow_v_2=vec.arrow(0, 0, *v_2, head_width=0.5, head_length=0.5,␣
↪color='m',label='v_2')
arrow_u_1=vec.arrow(0, 0, *u_1, head_width=0.5, head_length=0.5, color='b',␣
↪label='u_1')
arrow_u_2=vec.arrow(0, 0, *u_2, head_width=0.5, head_length=0.5, color='c',␣
↪label='u_2')
arrow_Av_1=vec.arrow(0, 0, *Av_1,head_width=0.5, head_length=0.5, color='r',␣
↪label='Av_1')
arrow_Av_2=vec.arrow(0, 0, *Av_2,head_width=0.5, head_length=0.5, color='r',␣
↪label='Av_2')

# set the legend
plt.legend([arrow_v_1,arrow_v_2,arrow_u_1,arrow_u_2,arrow_Av_1,arrow_Av_2],␣
↪['v_1','v_2','u_1','u_2','Av_1','Av_2'],loc=1, prop={'size': 30})
```

U=

```
 [[ 0.93788501  0.34694625]
 [ 0.34694625 -0.93788501]]
sigma=
 [5.41565478 1.29254915]
Sigma=
 [[5.41565478 0.        ]
 [0.         1.29254915]]
Vt=
 [[ 0.10911677  0.99402894]
 [ 0.99402894 -0.10911677]]
U.Sigma.Vt=
 [[ 1.  5.]
 [-1.  2.]]
A=
 [[ 1  5]
 [-1  2]]
```

[44]: <matplotlib.legend.Legend at 0x7fd1f68e1130>



2 Let's use the SVD to explore the action of a matrix on a unit circle in 2D space.

$$A = U\Sigma V^t$$

U and V could be rotation or reflection matrices.

The transpose of a rotation matrix is a rotation in the opposite direction. So if a matrix rotates clockwise its transpose rotates counterclockwise.

A rotation matrix clockwise by an angle θ is:

$$\begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}$$

A rotation matrix counterclockwise by an angle θ is the transpose of the above matrix:

$$\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

A reflection matrix about a line L making an angle θ with the x-axis is:

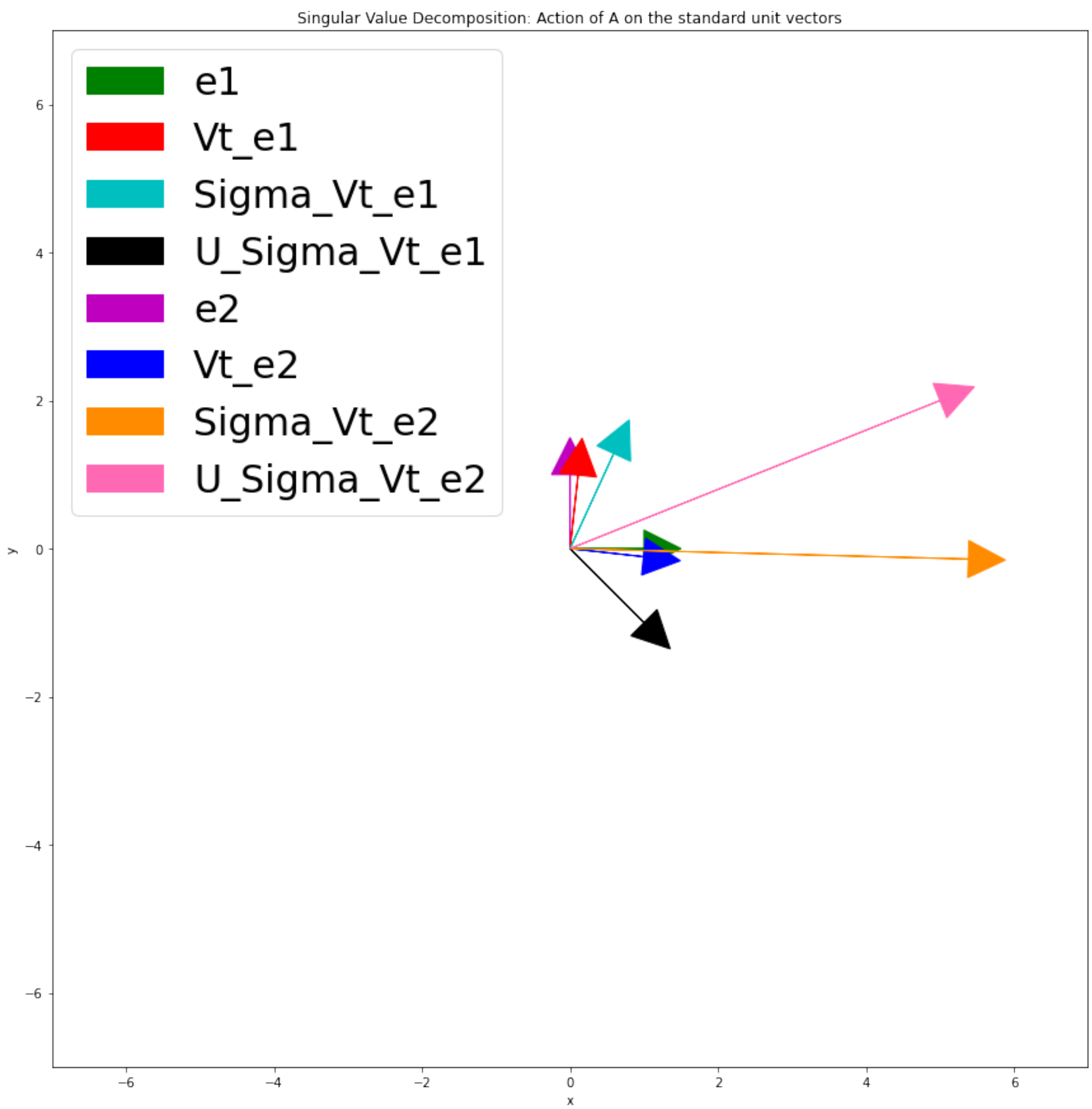
$$\begin{pmatrix} \cos 2\theta & \sin 2\theta \\ \sin 2\theta & -\cos 2\theta \end{pmatrix}$$

The determinant of a rotation matrix is 1 and the determinant of the reflection matrix is -1. Notice that both rotations and reflections have orthonormal rows and columns. Also, their inverse is their transpose.

[46]:
```
e1=[1,0]
e2=[0,1]

Vt_e1=(Vt).dot(e1)
Vt_e2=(Vt).dot(e2)
Sigma_Vt_e1=Sigma.dot(Vt_e1)
Sigma_Vt_e2=Sigma.dot(Vt_e2)
U_Sigma_Vt_e1=U.dot(Sigma_Vt_e1)
U_Sigma_Vt_e2=U.dot(Sigma_Vt_e2)

# Plot the vectors using arrow in matplolib.pyplot.axes
# set the figure and labels
plt.figure(figsize=(15,15))
vec= plt.axes()
plt.axis('scaled') # the scale on the x-axis is the same as the y-axis
plt.xlim(-7,7)
plt.ylim(-7,7)
plt.title('Singular Value Decomposition: Action of A on the standard unit␣
 ↪vectors')
plt.xlabel('x')
plt.ylabel('y')

# plot the vectors as arrows
arrow_e1=vec.arrow(0, 0, *e1, head_width=0.5, head_length=0.5, color='g',␣
 ↪label='e1')
arrow_e2=vec.arrow(0, 0, *e2, head_width=0.5, head_length=0.5,␣
 ↪color='m',label='e2')
arrow_Vt_e1=vec.arrow(0, 0, *Vt_e1, head_width=0.5, head_length=0.5,␣
 ↪color='r',label='Vt_e1')
arrow_Vt_e2=vec.arrow(0, 0, *Vt_e2, head_width=0.5, head_length=0.5,␣
 ↪color='b',label='Vt_e2')
arrow_Sigma_Vt_e1=vec.arrow(0, 0, *Sigma_Vt_e1, head_width=0.5, head_length=0.
 ↪5, color='c',label='Sigma_Vt_e1')
```

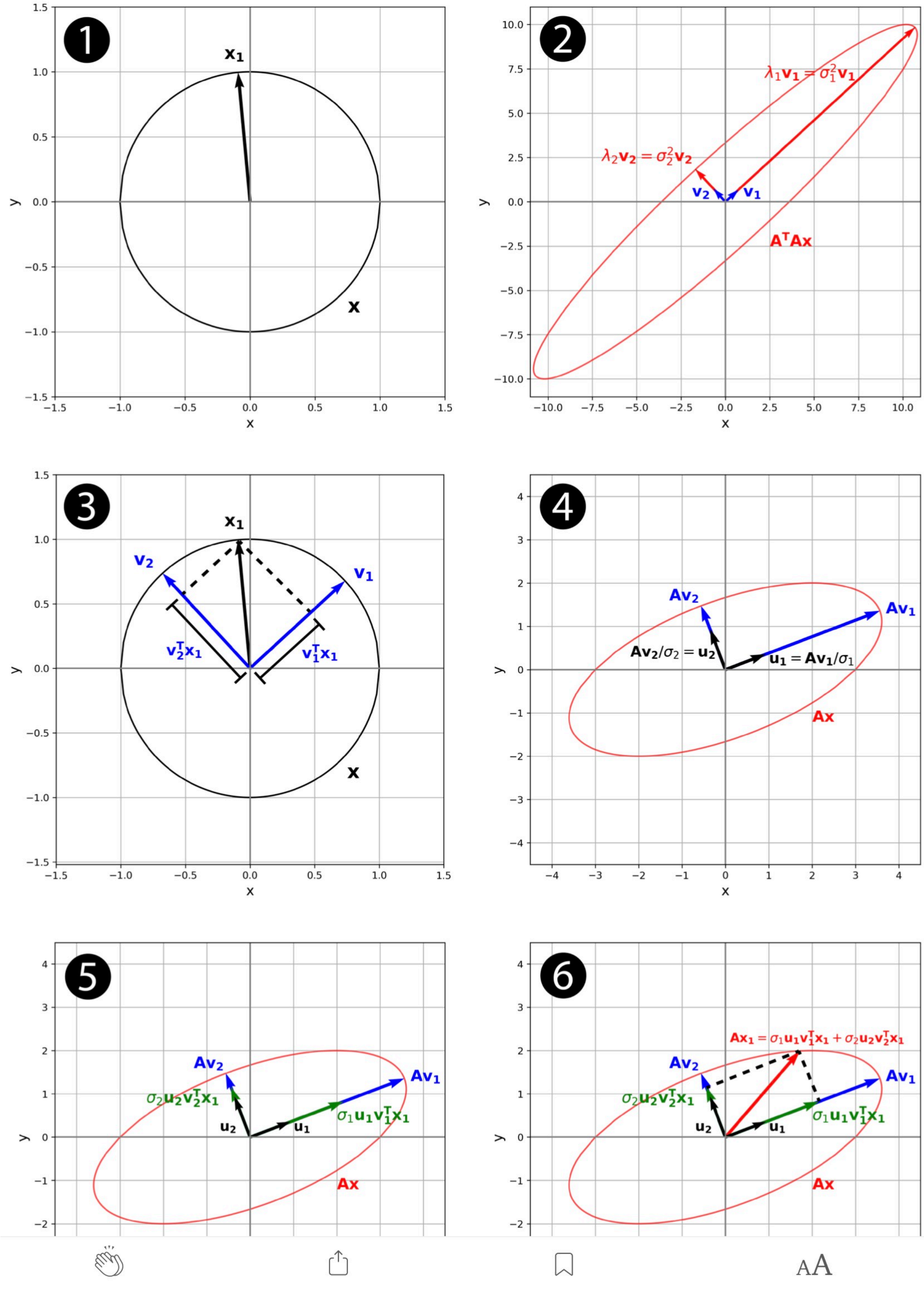
```
arrow_Sigma_Vt_e2=vec.arrow(0, 0, *Sigma_Vt_e2, head_width=0.5, head_length=0.
↪5, color='darkorange',label='Sigma_Vt_e2')
arrow_U_Sigma_Vt_e1=vec.arrow(0, 0, *U_Sigma_Vt_e1, head_width=0.5,␣
↪head_length=0.5, color='k',label='U_Sigma_Vt_e1')
arrow_U_Sigma_Vt_e2=vec.arrow(0, 0, *U_Sigma_Vt_e2, head_width=0.5,␣
↪head_length=0.5, color='hotpink',label='U_Sigma_Vt_e2')


# set the legend
plt.legend([arrow_e1,arrow_Vt_e1,arrow_Sigma_Vt_e1, arrow_U_Sigma_Vt_e1,␣
↪arrow_e2,arrow_Vt_e2, arrow_Sigma_Vt_e2, arrow_U_Sigma_Vt_e2],␣
↪['e1','Vt_e1','Sigma_Vt_e1','U_Sigma_Vt_e1','e2','Vt_e2','Sigma_Vt_e2','U_Sigma_Vt_e2'],loc
↪prop={'size': 30})
```

[46]: <matplotlib.legend.Legend at 0x7fd1d72b3940>



3 Let's convince ourselves with this picture

(credit: Reza Bagheri, published in towards Data Science)



4 There are three ways to multiply two matrices $A_{m\times n}$ and $B_{n\times s}$ together:

1.

4.1 Produce one entry ab_{ij} at a time by taking the dot product of a row with a column:

$$ab_{ij} = \sum_{k=1}^{n} a_{ik}b_{kj}$$

2.

4.2 Produce one column AB_l at a time by linearly combining the columns of A with the entries in the columns of B:

$$AB_l = b_{1l}A_1 + b_{2l}A_2 + \cdots + b_{nl}A_n$$

3.

4.3 Produce rank one pieces of the product one at a time by multiplying a column of A with the corresponding row of B, then add all these rank one matrices together to get the final product AB:

$$AB = A_1B_1^r + A_2B_2^r + \cdots + A_nB_n^r$$

where A_l is the lth column of A and B_l^r is the lth row of B.

5 How does this help us understand the usefulness of the Singular Value Decomposition?

Recall that $A = U\Sigma V^t$. We can expand this product using the sum of rank one matrices method to multiply $U\Sigma$ with V^t (note that $U\Sigma$ scales each colum U_i of U by σ_i):

$$A = U\Sigma V^t = \sigma_1 U_1 V_1^t + \sigma_2 U_2 V_2^t + \cdots + \sigma_r U_r V_r^t$$

where r is the rank of the matrix A. The great thing about this expression is that it splits A into a sum of rank one matrices arranged according to their order of importance. Moreover, it provides a straightforward way to approximate A by lower rank matrices by setting lower sigular values to zero.

```
[ ]:
```