Science 2 - Assignment 1

Name: Sreeja Guduri
Roll number: 2021102007

QUESTION 1:

a)

(1) SVD refers to singular value decomposition and it is a technique used to factorize matrices into three different matrices - two orthogonal matrices and one diagonal matrix.

$A = U \Sigma V^T$ [SVD]

Given, $A = \begin{bmatrix} 0 & 7 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix}$

STEPS TO CALCULATE SVD:

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

$A = U \Sigma V^T$

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

$= V \Sigma^T \Sigma V^T$ —(1) [$U^T U = I$; orthogonal]

and, $AV = U \Sigma$ —(2)

These two equations are used to find the (U, V, Σ) matrices.

(ii) We first find A^TA (NOTE: The computations are done using python code that has also been submitted)

$$A^TA = \begin{bmatrix} 0 & 4 & 7 & 10 \\ 7 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix} \begin{bmatrix} 0 & 7 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix}$$

$$= \begin{bmatrix} 165 & 186 & 207 \\ 186 & 259 & 255 \\ 207 & 255 & 270 \end{bmatrix}$$

(iii) We then find the eigenvalues and corresponding eigenvectors of this matrix by solving the characteristic equation : $\det(A - \lambda I) = 0$

$\lambda_1 = 672.07$; $\lambda_2 = 21.47$; $\lambda_3 = 0.452$

and the corresponding eigenvectors are :-

$$v_1 = \begin{bmatrix} -0.48 \\ -0.61 \\ -0.63 \end{bmatrix}$$

$$v_2 = \begin{bmatrix} -0.66 \\ 0.72 \\ -0.19 \end{bmatrix} ; \quad v_3 = \begin{bmatrix} -0.57 \\ -0.32 \\ 0.75 \end{bmatrix}$$

(iv) Now, the V matrix is just the eigenvectors normalised and stacked together.

$$V = \begin{bmatrix} -0.48 & -0.66 & -0.57 \\ -0.60 & 0.72 & -0.32 \\ -0.63 & -0.19 & 0.75 \end{bmatrix}$$

(v) the Σ matrix has its diagonal entries as the square roots of the eigenvalues, arranged in descending values.

$$\therefore \Sigma = \begin{bmatrix} 25.92 & 0 & 0 \\ 0 & 4.63 & 0 \\ 0 & 0 & 0.67 \\ 0 & 0 & 0 \end{bmatrix}$$

(vi) we use equation [2] to find the U matrix using the computed V and Σ matrix.

We see that the relation is : $u_i = \frac{1}{\sigma_i} A v_i$

where $\sigma_i = \sqrt{\lambda_i}$

Thus, the U matrix obtained after normalising -

$$= \begin{bmatrix} -0.23 & 0.97 & -0.05 & 0 \\ -0.33 & -0.03 & 0.84 & 0 \\ -0.53 & -0.12 & 0.17 & 0 \\ -0.73 & -0.20 & -0.49 & 0 \end{bmatrix}$$

(vii) Thus, we found the three SVD matrices. To verify, we can multiply the three matrices and see if we get A back

ie, A $\boxed{U \Sigma V^T = A}$

Code:

```python
import numpy as np

A = np.array([[0, 7, 3],
       [4, 5, 6],
       [7, 8, 9],
       [10, 11, 12]])

At = np.transpose(A)
AA = np.dot(At,A)
eigenvalues, eigenvectors = np.linalg.eig(AA) #calculates eigen values and eigen vectors
print(f'AA = {AA}\n')
print(f'eigenvalues = {eigenvalues}\n')
print(f'eigenvectors = {eigenvectors}\n')

#normalized eigenvectors
normal = [v/np.linalg.norm(v) for v in np.transpose(eigenvectors)]
```

```python
V = np.column_stack(normal) #This is the required V matrix in SVD
print(f'V = {V}\n')

#now to find the sigma matrix - diagonal matrix which is same size as original
eigenvalues = sorted(eigenvalues,reverse=True)
S = np.zeros((4,3))
S[0,0] = np.sqrt(eigenvalues[0])
S[1,1] = np.sqrt(eigenvalues[1])
S[2,2] = np.sqrt(eigenvalues[2])
print(f'S = {S}\n')

# To find U we first do A x V, normalise and then multiply by sigma matrix
U = np.dot(A,V)
U = [v/np.linalg.norm(v) for v in np.transpose(U)]
U = np.column_stack(U)

U = np.dot(U,np.transpose(S))

res = []
for v in np.transpose(U):
  if np.all(v == 0):
    res.append(v)
  else:
    res.append(v / np.linalg.norm(v))
U = np.column_stack(res)
print(f'U = {U}\n')


print(f'A = {np.dot(np.dot(U,S),np.transpose(V))}\n')
```

b)

b) Standard diagonilization of a matrix refers to factorizing the matrix as :-

$A = PDP^{-1}$

For this to work, A must be square and invertible.

- SVD is like a generalization of diagonalization to non-square matrices.

For SVD, $A = U \Sigma V^T$

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

$\Rightarrow A^T A = V \Sigma^T U^T U \Sigma V^T$

$A^T A = V \Sigma^T \Sigma V^T$

$= V \Sigma^T \Sigma V^{-1}$ [V is orthogonal]

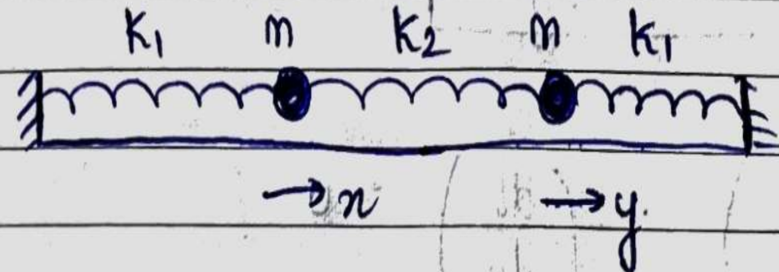
This is the same as diagonalization except that is now $A^T A$.

for SVD to be equal to diagonilization, $A^T A = A$ → which is true for the identity matrix.

QUESTION 2:

a)

2



x and y are the displacements of the two masses.

Then, the speeds of the masses are given by $\dot{x}$ and $\dot{y}$!

$\Rightarrow$ Total KE, $T = \frac{1}{2} m\dot{x}^2 + \frac{1}{2} m\dot{y}^2$

Total PE $= V = \frac{1}{2} k_1 x^2 + \frac{1}{2} k_2 (y-x)^2 + \frac{1}{2} k_1 y^2$

[$(y-x)$ is the compression of second string]

Applying Lagrange's equation of motion,

$$L = KE - PE$$

$$= \frac{1}{2} m\dot{x}^2 + \frac{1}{2} m\dot{y}^2 - \frac{1}{2} k_1 x^2 - \frac{1}{2} k (y-x)^2 - \frac{1}{2} k_1 y^2$$

Now, $\frac{d}{dt}\left(\frac{dL}{dx}\right) - \left(\frac{dL}{dx}\right) = 0$

$$\Rightarrow \boxed{m\ddot{x} + (k_1+k_2)x - k_2 y = 0}$$

Similarly, $\frac{d}{dt}\left(\frac{dL}{dy}\right) - \left(\frac{dL}{dy}\right) = 0$

$$\Rightarrow \boxed{m\ddot{y} + (k_1+k_2)y - k_2 x = 0}$$

Thus, the Langrangian equations of motion for this system have been obtained.

b)

b) Yes, the solution to the above system of equations can be done through eigenanalysis.

we know the eqn's of motion: $m\ddot{x} + (k_1 + k_2)x - k_2 y = 0$

$m\ddot{y} + (k_1+k_2)y - k_2 x = 0$

$$m_1\ddot{x} + k_1(x-y) + k_2 x = 0$$

$$m_2\ddot{y} + k_1(y-x) + k_2 y = 0$$

In matrix form, this can be written as:

$$M\ddot{X} + KX = 0$$

where,

$$M = \begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix}$$

$$K = \begin{bmatrix} k_2 + k_1 & -k_2 \\ -k_2 & k_1 + k_2 \end{bmatrix}$$

$$X = \begin{bmatrix} x \\ y \end{bmatrix}$$

now,

$$\ddot{X} = \frac{1}{M}(-KX)$$

$$= \frac{1}{m}\begin{bmatrix} -(k_2+k_1) & k_2 \\ k_2 & -(k_2+k_1) \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}$$

$$= -\omega^2 \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Thus, we can find a solution using eigenvector analysis.

$$\begin{bmatrix} \omega^2 - \frac{k_2+k_1}{m} & \frac{k_2}{m} \\ \frac{k_2}{m} & \omega^2 - \frac{k_2+k_1}{m} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

We find the solutions when determinant is zero

$$\Rightarrow \left(\omega^2 - \frac{k_1+k_2}{m}\right)^2 - \left(\frac{k_2}{m}\right)^2 = 0$$

$$\boxed{\omega_\pm^2 = \frac{k_1 + k_2 \pm k_2}{m}}$$

natural frequency solution $= \omega_\pm$

now if $k_1 = k_2 = k$ and $m_1 = m_2 = m$

$$\omega_\pm^2 = \frac{k+k \pm k}{m}$$

$$= \frac{2k \pm k}{m}$$

Thus, the frequencies are:

$$\omega_+ = \sqrt{\frac{3k}{m}} \; ; \quad \omega_- = \sqrt{\frac{k}{m}}$$

c)

c) Given:- $\dot{x}_0 = 0$, $\dot{y}_0 = 0$ (initial velocities are zero)

- $x = C$; $y = -C$

where $C = 7 + 0 + 1 = \underline{\underline{8}}$

Now we know $M\ddot{x} + kx = 0$

where $M = \begin{bmatrix} m & 0 \\ 0 & m \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

$k = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$

The nature of the eigenvalues of $M^{-1}k$ will tell us about the motion of the system.

Eigenvalues : $\lambda_1 = 1$ and $\lambda_2 = 3$

Since bothe eigenvalues are positive $\Rightarrow$ undamped and oscillatory motion.

This is also verified by the graph obtained using code ; they have equal magnitude displacement but in opposite direction.

[NOTE: The code uses python libraries to solve initial value problem (ivp) ordinary differential equations].

(ii) Similarly, for $x = -C$, $y = -C$
the motion will be oscillatory and in-phase because they are displaced by the same amount in the same direction.

(iii) for $x = -C/2$ and $y = -C$:-
The eigenvalues are still positive so the motion will be undamped and oscillatory; however both motions will have different magnitudes and thus will be out of phase.

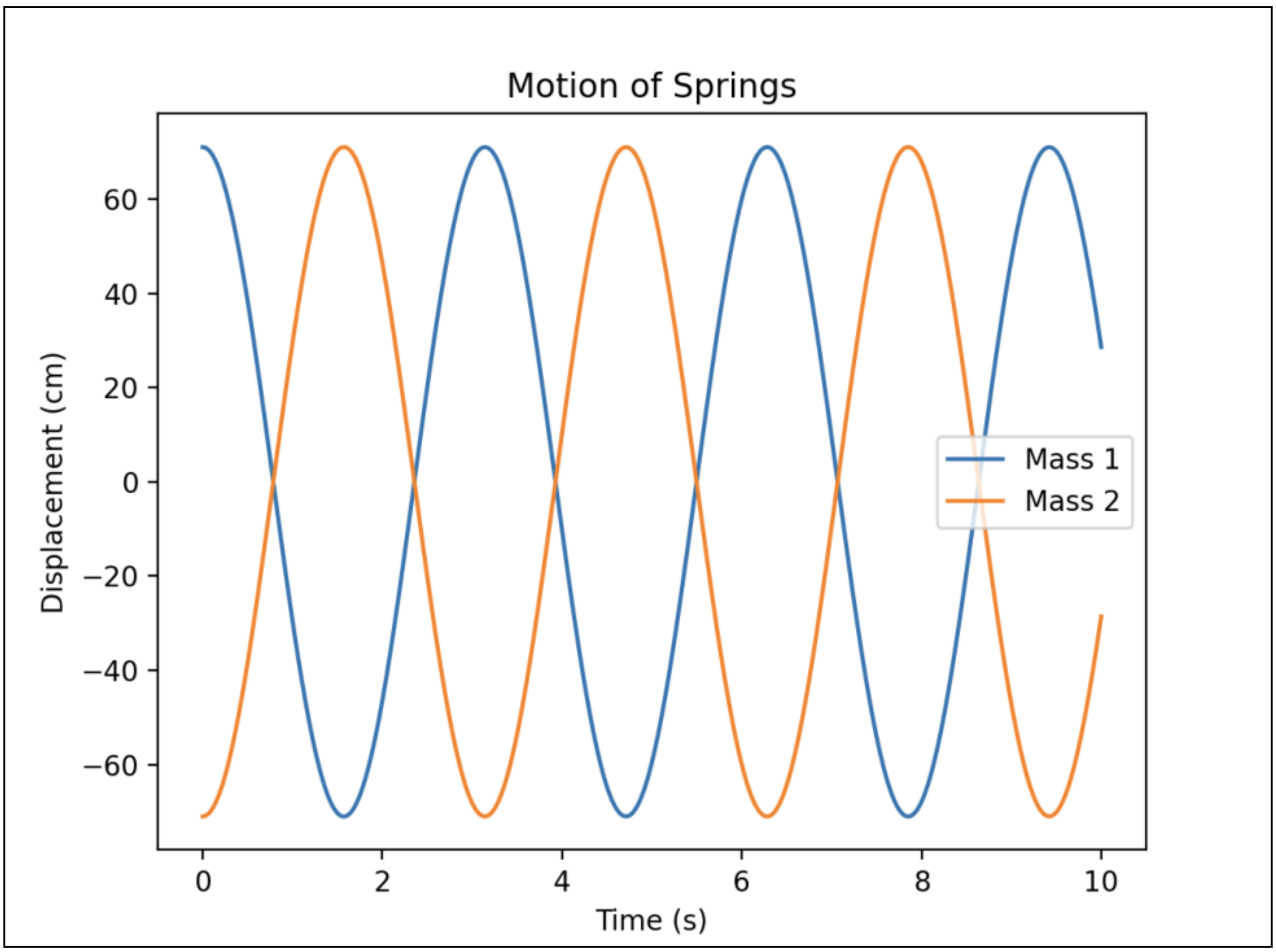
Code:

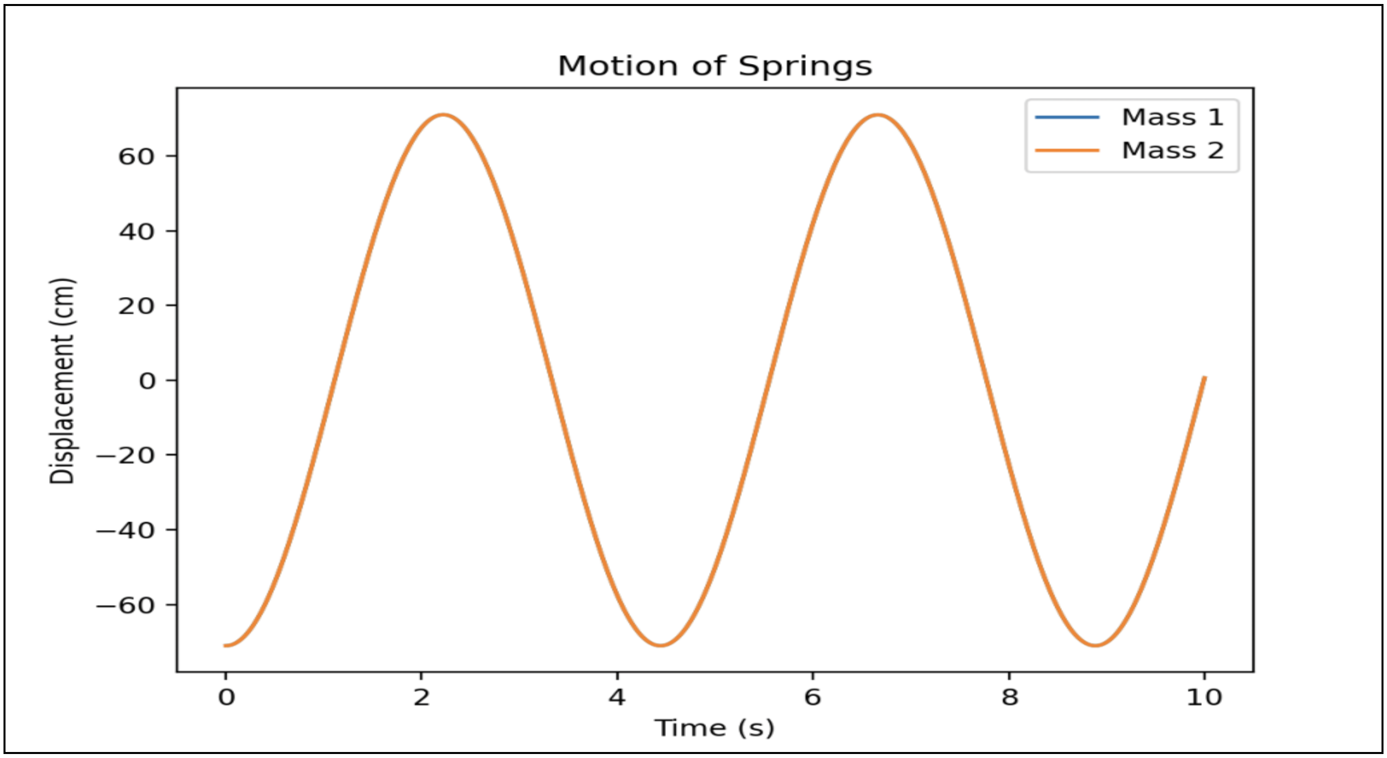
```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import solve_ivp

def spring(t, y, m, k1, k2):
  dydt = [y[2],
      y[3],
      -(k2 + k1) / m * y[0] + k1 / m * (y[1] - y[0]),
      -(k1+ k2) / m * y[1] - k2 / m * (y[1] - y[0])]
  return dydt

m = 1
k1 = 1
k2 = 1
C = 70 + 1
t = np.linspace(0, 10, 500)

initial_conditions = [C, -C, 0, 0]

# Solve ODEs
solution = solve_ivp(
```

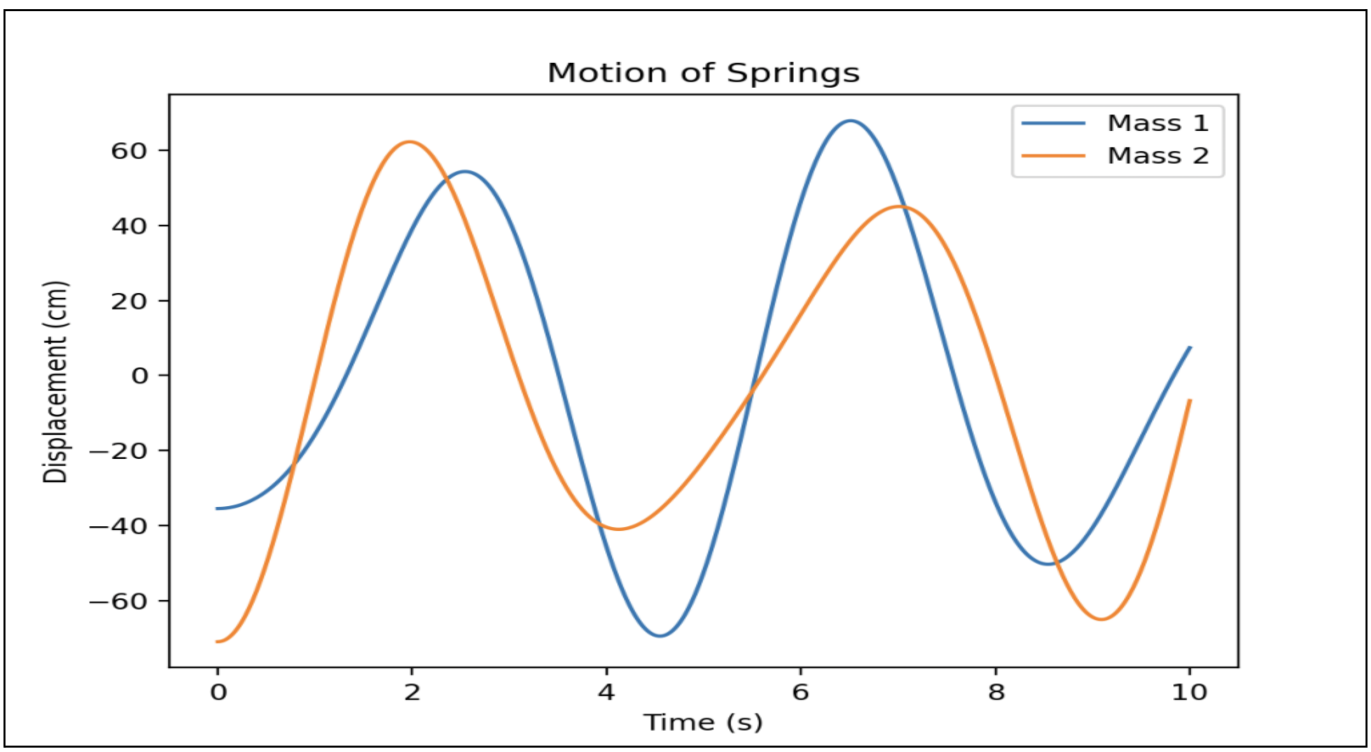
```
  spring,
  [t[0], t[-1]],
  initial_conditions,
  args=(m, k1, k2),
  t_eval=t
)

plt.plot(solution.t, solution.y[0], label='Mass 1')
plt.plot(solution.t, solution.y[1], label='Mass 2')
plt.xlabel('Time (s)')
plt.ylabel('Displacement (cm)')
plt.title('Motion of Springs')
plt.legend()
plt.show()

initial_conditions = [-C, -C, 0, 0]

# Solve ODEs
solution = solve_ivp(
  spring,
  [t[0], t[-1]],
  initial_conditions,
  args=(m, k1, k2),
  t_eval=t
)

plt.plot(solution.t, solution.y[0], label='Mass 1')
plt.plot(solution.t, solution.y[1], label='Mass 2')
plt.xlabel('Time (s)')
plt.ylabel('Displacement (cm)')
plt.title('Motion of Springs')
plt.legend()
plt.show()

initial_conditions = [-0.5*C, -C, 0, 0]

# Solve ODEs
solution = solve_ivp(
  spring,
  [t[0], t[-1]],
  initial_conditions,
  args=(m, k1, k2),
  t_eval=t
)
```

```
plt.plot(solution.t, solution.y[0], label='Mass 1')
plt.plot(solution.t, solution.y[1], label='Mass 2')
plt.xlabel('Time (s)')
plt.ylabel('Displacement (cm)')
plt.title('Motion of Springs')
plt.legend()
plt.show()
```

Results:




Motion of Springs
Mass 1
Mass 2
Displacement (cm)
Time (s)
60
40
20
0
−20
−40
−60
0
2
4
6
8
10


Motion of Springs
Mass 1
Mass 2
Displacement (cm)
Time (s)
60
40
20
0
−20
−40
−60
0
2
4
6
8
10

QUESTION 3:

3

The matrix has been constructed and the eigenvalues are plotted using code :-

(i) The shape of the distribution obtained is an oval in a complex plane.
This shape is characteristic of matrices with random entries that have been sampled from a normal distribution.
This type of distribution is usually called a scattering.

(ii)

• First thing we notice from the plots is that as the matrix S increases, the scattering becomes

The distribution becomes more scattered. As S decrease, the distribution becomes denser.

• C refers to the sparsity of the matrix and refers to the density of non-zero entries in the matrix. Higher C causes more zero values which effects the shape (more scattered)

Higher the value of σ, broader is the scattering

The value of the diagonal entries 'd' indicates that there is a shift in the eigenvalues along the real axis, by an amount proportional to 'd'.

<u>Code:</u>

Python

```python
import numpy as np
import matplotlib.pyplot as plt

def matrix(S, d, C, sigma):
  matrix = np.zeros((S, S))

  for i in range(S):
    for j in range(S):
      if i == j:
        matrix[i, j] = -d
      else:
        if np.random.rand() <= C:
          matrix[i, j] = np.random.normal(0, sigma)
  return matrix

S1 = 100
```

```python
S2 = 500
C = 8
d = 7
sigma = 1

matrix_1 = matrix(S1, d, C, sigma)
ev1 = np.linalg.eigvals(matrix_1)
matrix_2 = matrix(S2, d, C, sigma)
ev2 = np.linalg.eigvals(matrix_2)

plt.scatter(np.real(ev1), np.imag(ev1), label = "Matrix size = 100")
plt.scatter(np.real(ev2), np.imag(ev2), label = "Matrix size = 500")
plt.legend()
plt.title("Eigenvalues")
plt.xlabel("Real part of eigenvalues")
plt.ylabel("Imaginary part of eigenvalues")
plt.show()
```

Results: