Matrix A

```
import numpy as np
A=np.array([[8,3],[0,4],[9,9],[7,2],[4,6]])
print(A)
```

```
[[8 3]
 [0 4]
 [9 9]
 [7 2]
 [4 6]]
```

Scaled Matrix

```
golden_ratio=-0.618

scaled_matrix=A*golden_ratio

print(scaled_matrix)
```

```
[[-4.944 -1.854]
 [-0.    -2.472]
 [-5.562 -5.562]
 [-4.326 -1.236]
 [-2.472 -3.708]]
```

Matrix M

```
scaled_matrix_T=np.transpose(scaled_matrix)

M=np.dot(scaled_matrix,scaled_matrix_T)

print("Matrix M",M)
```

```
Matrix M [[27.880452  4.583088 37.810476 23.679288 19.0962  ]
 [ 4.583088  6.110784 13.749264  3.055392  9.166176]
 [37.810476 13.749264 61.871688 30.935844 34.37316 ]
 [23.679288  3.055392 30.935844 20.241972 15.27696 ]
 [19.0962    9.166176 34.37316  15.27696  19.860048]]
```

Inverse and PseudoInverse of M

```python
A_inv=np.linalg.inv(M)
print("Inverse of M",A_inv)

A_pinv=np.linalg.pinv(M)
print("Pseudoinverse of M",M)
```

```
Inverse of M [[-3.34251535e+14  2.81474977e+14 -2.70919665e+14  6.29800260e+14
   1.75921860e+14]
 [ 2.85873023e+14  6.33318698e+14 -6.03021044e+13  2.72678884e+13
  -4.83785116e+14]
 [-1.40737488e+14  0.00000000e+00 -1.97032484e+14  2.53327479e+14
   2.81474977e+14]
 [ 5.62949953e+14 -0.00000000e+00  3.62789970e+14 -7.88129935e+14
  -5.62949953e+14]
 [-0.00000000e+00 -5.62949953e+14  3.50279971e+14 -4.50359963e+14
  -0.00000000e+00]]
Pseudoinverse of M [[27.880452  4.583088 37.810476 23.679288 19.0962  ]
 [ 4.583088  6.110784 13.749264  3.055392  9.166176]
 [37.810476 13.749264 61.871688 30.935844 34.37316 ]
 [23.679288  3.055392 30.935844 20.241972 15.27696 ]
 [19.0962    9.166176 34.37316  15.27696  19.860048]]
```

Eigen values and Eigenvec

```python
eigenvalues,eigenvectors=np.linalg.eig(M)

print("Eigenvalues",eigenvalues)
print("Eigenvectos",eigenvectors)
```

```
Eigenvalues [ 1.23948349e+02  2.07214129e-15  1.20165952e+01 -2.02667470e-15
  1.48267950e-15]
Eigenvectos [[-0.45070951 -0.75632676  0.47416323  0.12502829  0.67324839]
 [-0.13880727 -0.26622287 -0.55658732  0.07693483 -0.04118159]
 [-0.70224593  0.26220881 -0.24926728 -0.61834381  0.00482383]
 [-0.37268219  0.53671878  0.5018596   0.23118021 -0.7347846 ]
 [-0.38151294 -0.01657419 -0.38907911  0.73665161 -0.07147735]]
```

Engery of M using Frobenious Norm

```python
frobenious_norm=np.linalg.norm(M,'fro')

Energy= frobenious_norm ** 2

print("Enegry of M: ",Energy)
```

```
Enegry of M:  15507.591733973664
```

SVD

```
U,Sigma,VT=np.linalg.svd(M)

print("Matrix U",U)

print("Singular Matiex",Sigma)

print("V^T",VT)
```

```
Matrix U [[-0.45070951  0.47416323 -0.57342049  0.17106861 -0.4625523 ]
 [-0.13880727 -0.55658732  0.15105564  0.72236291 -0.35541119]
 [-0.70224593 -0.24926728  0.36885831 -0.51119174 -0.21758454]
 [-0.37268219  0.5018596   0.48295819  0.42943494  0.43770075]
 [-0.38151294 -0.38907911 -0.52826704  0.05653305  0.64869351]]
Singular Matiex [1.23948349e+02 1.20165952e+01 3.50505485e-15 3.24149410e-15
 2.50004586e-16]
V^T [[-0.45070951 -0.13880727 -0.70224593 -0.37268219 -0.38151294]
 [ 0.47416323 -0.55658732 -0.24926728  0.5018596  -0.38907911]
 [ 0.51441368  0.29576792 -0.62247284 -0.07594395  0.50463846]
 [ 0.54409721  0.23297742  0.19759174 -0.57107389 -0.53339654]
 [ 0.10661594 -0.72745169  0.1349131  -0.52664225  0.40483759]]
```