Q 1.Perform svd on the metrix getting X,B,T

Perform the following operations.

X+T

X-T

2X^3

Code:

```
import numpy
import numpy as  np
from scipy.linalg import svd
a =
np.array([[13,15,29,17],[1,11,15,7],[19,1,5,23],[15,1
7,9,5]])
print("Matrix is:\n",a)
X,B,T = svd(a)
print("Decomposition:\n",X)
print("\nInverse:",B)
print("\nTranspose:\n",T)
print("Matrix Addition:")
print(numpy.add(X,T))
print("Matrix Subtraction:")
print(numpy.subtract(X,T))
print("Cube of the Matrix X:")
print(numpy.multiply(X,X,X))
print("\n 2X^3::")
print(numpy.dot(2,X))
```

Output:

```
Matrix is:
 [[13 15 29 17]
 [ 1 11 15  7]
 [19  1  5 23]
 [15 17  9  5]]
Decomposition:
 [[-0.72017841 -0.29467198 -0.35628629 -0.5172732 ]
 [-0.33088475 -0.39548562 -0.21005478  0.83065239]
 [-0.44950585  0.86307431 -0.10375149  0.20562795]
 [-0.41207139 -0.10891413  0.90452899  0.01273521]]

Inverse: [53.04573528 21.52569661 12.92602038  3.42233122]

Transpose:
 [[-0.46026106 -0.41279714 -0.59956972 -0.50820707]
 [ 0.48957666 -0.45335989 -0.51764305  0.53556062]
 [ 0.52258002  0.58938049 -0.45343583 -0.41705719]
 [-0.52477098  0.52601708 -0.40860465  0.53006008]]
Matrix Addition:
[[-1.18043946 -0.70746912 -0.95585602 -1.02548027]
 [ 0.15869191 -0.84884551 -0.72769783  1.36621301]
 [ 0.07307416  1.4524548  -0.55718732 -0.21142924]
 [-0.93684238  0.41710295  0.49592434  0.54279528]]
```

```
 Matrix Subtraction:
 [[-0.25991735  0.11812517  0.24328343 -0.00906613]
  [-0.82046141  0.05787427  0.30758826  0.29509177]
  [-0.97208587  0.27369381  0.34968435  0.62268515]
  [ 0.11269959 -0.63493121  1.31313364 -0.51732487]]
 Cube of the Matrix X:
 [[5.18656935e-01 8.68315737e-02 1.26939923e-01 2.67571568e-01]
  [1.09484719e-01 1.56408875e-01 4.41230126e-02 6.89983393e-01]
  [2.02055511e-01 7.44897263e-01 1.07643716e-02 4.22828542e-02]
  [1.69802834e-01 1.18622878e-02 8.18172693e-01 1.62185451e-04]]

  2X^3::
 [[1.03731387e+00 1.73663147e-01 2.53879847e-01 5.35143135e-01]
  [2.18969438e-01 3.12817751e-01 8.82460252e-02 1.37996679e+00]
  [4.04111022e-01 1.48979453e+00 2.15287431e-02 8.45657084e-02]
  [3.39605669e-01 2.37245755e-02 1.63634539e+00 3.24370901e-04]]

 Process finished with exit code 0
```

Q2. Program for NLP which perform chunking

Code:

```
import nltk
text = "Rama killed ravana for seetha, it is a famous
story that is created as several serials."
tokenize = nltk.word_tokenize(text)
new_tag = nltk.pos_tag(tokenize)
print(new_tag)
grammer = r" NP{<DT> ? <JJ> * <NN>}"
chunkgram = nltk.RegexpChunkParser(grammer)
chunked = chunkgram.parse(new_tag)
print(chunked)
chunked.draw()
```

Output:

```
  File "C:\python\lib\site-packages\nltk\chunk\regexp.py", line 1090, in parse
    self._notrace_apply(chunkstr)
  File "C:\python\lib\site-packages\nltk\chunk\regexp.py", line 1050, in _notrace_apply
    rule.apply(chunkstr)
AttributeError: 'str' object has no attribute 'apply'
[('Rama', 'NNP'), ('killed', 'VBD'), ('ravana', 'NN'), ('for', 'IN'), ('seetha', 'NN'), (',', ','), ('it', 'PRP'), ('is', 'VBZ'), ('a', 'DT'), ('famous', 'JJ')

Process finished with exit code 1
```