

```

## APM120 HW-00
import sys
from math import cos
import numpy as np
from matplotlib import pyplot as plt

## Plot  $f(x)=\cos(3x)$  vs  $x$  for  $x=(-2\pi,\pi)$ :
x = np.linspace(-2*np.pi, np.pi, 1000)
f = np.cos(3*x)

# always label axes and provide titles:
plt.plot(x, f, 'b')
plt.xlabel('x')
plt.ylabel('f(x)=cos(3x)')
plt.title('Plot of f(x)=cos(3x)')

## Solve a linear equations  $Ax=b$ :
A = np.array([[8, 2, 7], [4, 2, 2], [6, 7, 0]])
b = np.array([3, 4, 1])

print 'determinant of A:'
print np.linalg.det(A)
print 'solution using matrix inverse is:'
x2 = np.linalg.inv(A).dot(b)
print 'checking that x is a solution; A*x2:'
print x2
print 'b:'
print b

## Calculate the eigenvalues and vectors of  $A$ :
print 'eigenvectors/ values of A:\n'

[V, D] = np.linalg.eig(A)
print "V=" print D

# Reminder: MATLAB indexes from 1, Python from 0.
## verify that V_1 is an eigenvector of A
print 'A*v1:\n'
print A.dot(V[:,0])
print 'lambda1*v1:\n'
print V[:,0]*D[0,0]

## verify that V_2 is an eigenvector of A
print 'A*v2:\n'
print A.dot(V[:,1])
print 'lambda2*v2:\n'
print V[:,1]*D[1,1]

## Calculate the eigenvalues and vectors of  $B$ :
B = np.array([[1, 3j, 8, 10j, 0, 3j], [7, 3j, 4, 9j, 3, 2j], [11, 3j, 16, 12j, 6, 5j], [17, 5j, 20, 14j, 11, 8j], [23, 7j, 28, 18j, 17, 12j], [29, 9j, 36, 24j, 23, 16j]])
print 'eigenvectors/ values of B:\n'

[V, D] = np.linalg.eig(B)
print "V=" print D

# Reminder: MATLAB indexes from 1, Python from 0.
## verify that V_1 is an eigenvector of B
print 'B*v1:\n'
print B.dot(V[:,0])
print 'lambda1*v1:\n'
print V[:,0]*D[0,0]

## verify that V_2 is an eigenvector of B
print 'B*v2:\n'
print B.dot(V[:,1])
print 'lambda2*v2:\n'
print V[:,1]*D[1,1]

```

```
print 'lambda2*v2:\n'  
print 1 1      1
```