PMDS508L - Python Programming
`scipy.linalg` Package

Scipy.linalg Package

- `scipy.linalg` package contains the linear algebraic functions to perform the operations on the matrices, solve the system of equations
- `scipy.linalg` functions are coplied with BLAS/LAPACK support, which results in a faster execution

Solving Linear Equations

- The `scipy.linalg.solve` feature solves the linear (system of) equations for the unknowns.
- As an example, consider solving the system

$$
\begin{aligned}
x + 2y + 5z &= 9 \\
2x - 5y + z &= 8 \\
2x - 3y + 8z &= 2
\end{aligned}
$$

[1]:
```
import numpy as np
from scipy.linalg import solve

A = np.array([[1,2,5],[2,-5,1],[2,-3,8]])
b = np.array([9,8,2])

x = solve(A,b)
print(x)
```

```
[11.71111111  2.75555556 -1.64444444]
```

Finding a Determinant, Eigenvalues and Eigenvectors

- We can use `scipy.linalg.det` function to find the determinant of the given matrix.

[2]:
```
from scipy.linalg import det
import numpy as np

A = np.array([[1,2],[3,4]])
d = det(A)

print(d)
```

```
-2.0
```

- For determining the eigenvalues and eigenvectors of a matrix, we can use `scipy.linalg.eig`

[3]:
```
from scipy.linalg import eig
import numpy as np

A = np.array([[1,2],[3,4]])
L,V = eig(A)

print('Eigenvalues are: ',L)
print('Eigenvectors are: ',V)
```

```
Eigenvalues are:  [-0.37228132+0.j  5.37228132+0.j]
Eigenvectors are:  [[-0.82456484 -0.41597356]
 [ 0.56576746 -0.90937671]]
```

Singular Value Decompostion

- A singular value decompostion (SVD) is a factorization of a real or complex matrix.
- It generalizes the eigendecomposition of a square normal matrix with an orthonormal eigenbasis to any $m \times n$ matrix.
- It can be tought of as an extension of the eigenvalue problem to matrices that are not square.
- For a matrix A of size $m \times n$ the factorization is of the form $M = U\Sigma V^*$, where U is an $m \times m$ (complex) matrix, Σ is a $m \times n$ rectangular diagonal matrix with non-negative real numbers on the diagonal, V is an $n \times n$ (complex) matrix and V^* is the conjugate transpose of V.
- SVD of a matrix can be found by using `scipy.linalg.svd`. The output of this function is U, s, Vh where U and Vh are two unitary matrices and s is a $1-$D array of singluar values (real, non-negative) such that that $\Sigma =$ diagonal matrix of s and $A = U\Sigma Vh$.

Python code sample to demonstrate SVD

[4]:
```
from scipy import linalg
import numpy as np

# Creating a random matrix A with real entries
A = np.random.rand(3,2)

# SVD demcomposition of A
U, s, Vh = linalg.svd(A)

print("U = ",U)
print("s = ",s)
print("Sigma = ",np.diag(s))
print("Vh =",Vh)
```

```
U =  [[-0.44523318  0.28584969 -0.84856194]
 [-0.32493203 -0.93465436 -0.14436205]
 [-0.83437797  0.21145018  0.50902085]]
```

```
s =  [1.36566551 0.43258201]
Sigma =  [[1.36566551 0.        ]
 [0.         0.43258201]]
Vh = [[-0.68392165 -0.72955546]
 [-0.72955546  0.68392165]]
```

[5]:
```
from scipy import linalg
import numpy as np

# Creating a random matrix A with complex entries
A = np.random.rand(3,2) + 1.j*np.random.randn(3,2)

# SVD demcomposition of A
U, s, Vh = linalg.svd(A)

print("U = ",U)
print("s = ",s)
print("Sigma = ",np.diag(s))
print("Vh =",Vh)
```

```
U =  [[-0.41486497+0.42009113j  0.7441306 -0.15704137j -0.1991394 -0.18265174j]
 [-0.36601581+0.01622119j -0.51437885-0.1103086j  -0.52433991-0.56043142j]
 [-0.52071958+0.49601505j -0.38009242-0.01959137j  0.54780063+0.19462937j]]
s =  [2.43101371 0.39425364]
Sigma =  [[2.43101371 0.        ]
 [0.         0.39425364]]
Vh = [[-0.68931548+0.j         -0.02392102-0.72406627j]
 [ 0.7244613 +0.j         -0.02276054-0.68893961j]]
```