Name : Pranali Dhwade Date : 21/06/2021

Subject : FDS Roll No. : 10

Aim : Image compression using svd in R and Python

In R :

To get dimension :

Code :

```
> x=readPNG(system.file("img", "Rlogo.png", package="png"))
> grid::grid.raster(x)
> dim(x)
```

Output :

```
> x=readPNG(system.file("img", "Rlogo.png", package="png"))
> grid::grid.raster(x)
> dim(x)
[1]  76 100   4
```

Code :

```
> svd.r=svd(x)
> x1=x[1:76,1:100,1]
> svd.r=svd(x1)
> grid::grid.raster(x1)
> grid::grid.raster(x)
> grid::grid.raster(x1)
> u1=as.matrix(svd.r$u[-1,1])
> dim(u1)
> v1=as.matrix(svd.r$v[-1,1])
> dim(v1)
> d1=diag(svd.r$v[1:1])
> dim(d1)
> d1=as.matrix(svd.r$v[1:1])
> dim(d1)
```

Output :

```
> svd.r=svd(x)
> x1=x[1:76,1:100,1]
> svd.r=svd(x1)
> grid::grid.raster(x1)
> grid::grid.raster(x)
> grid::grid.raster(x1)
> u1=as.matrix(svd.r$u[-1,1])
> dim(u1)
[1] 75  1
> v1=as.matrix(svd.r$v[-1,1])
> dim(v1)
[1] 99  1
> d1=diag(svd.r$v[1:1])
> dim(d1)
[1] 0 0
> d1=as.matrix(svd.r$v[1:1])
> dim(d1)
[1] 1 1
```

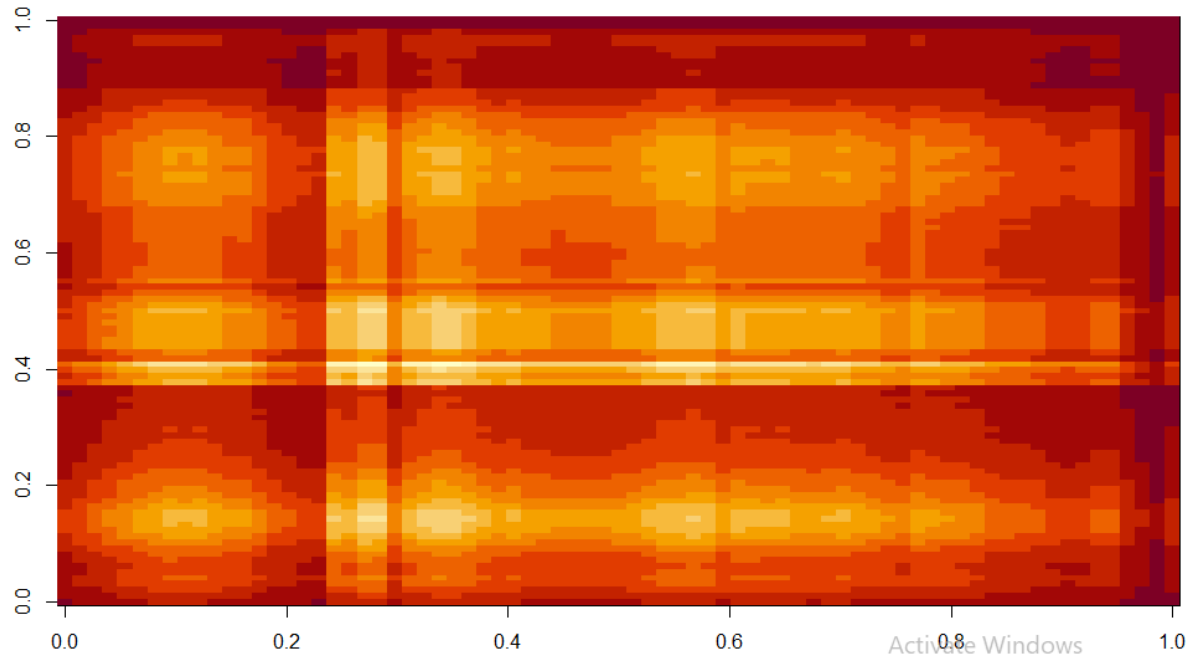
Code :

```
> i1= u1 %*% d1 %*% t(v1)

> image(i1)

> u5=as.matrix(svd.r$u[,1:5])

> v5=as.matrix(svd.r$v[,1:5])

> dim(u5)

> dim(v5)

> d1=diag(svd.r$d[1:1])

> dim(d1)
```

Output :

```
> i1= u1 %*% d1 %*% t(v1)
> image(i1)
> u5=as.matrix(svd.r$u[,1:5])
> v5=as.matrix(svd.r$v[,1:5])
> dim(u5)
[1] 76  5
> dim(v5)
[1] 100   5
> d1=diag(svd.r$d[1:1])
> dim(d1)
[1] 28 28
```

Image :



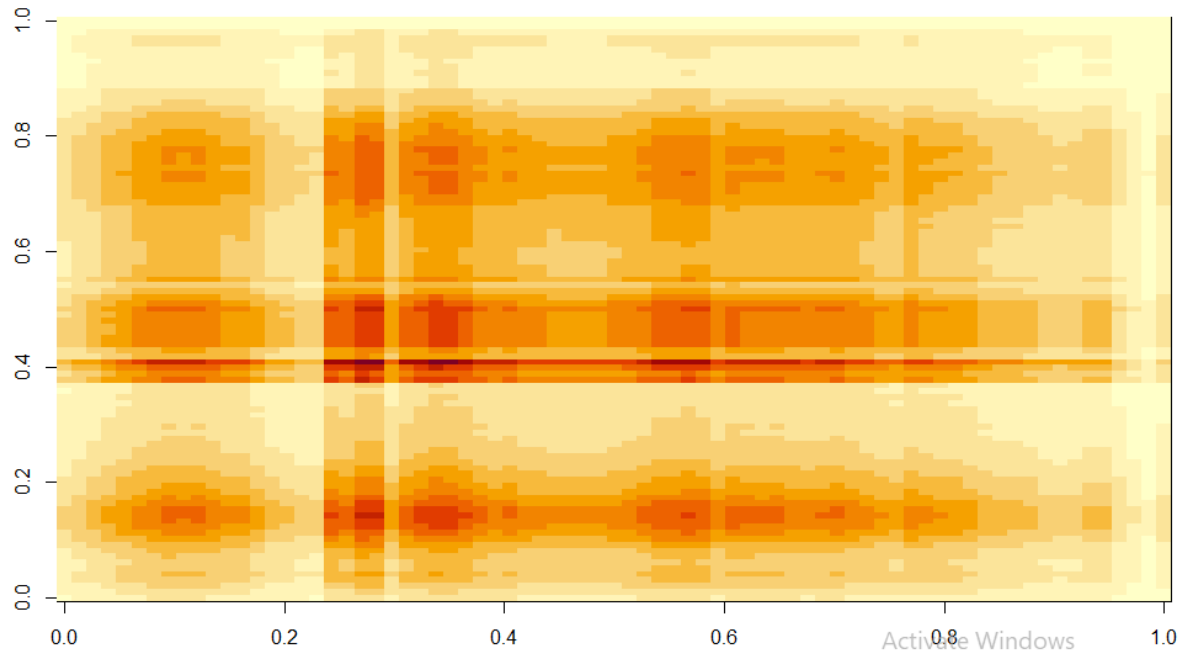
Code:

```
> d1=as.matrix(svd.r$d[1:1])
> dim(d1)
> i1= u1 %*% d1 %*% t(v1)
> image(i1)
```

Output :

```
> d1=as.matrix(svd.r$d[1:1])
> dim(d1)
[1] 1 1
> i1= u1 %*% d1 %*% t(v1)
> image(i1)
```

Image :



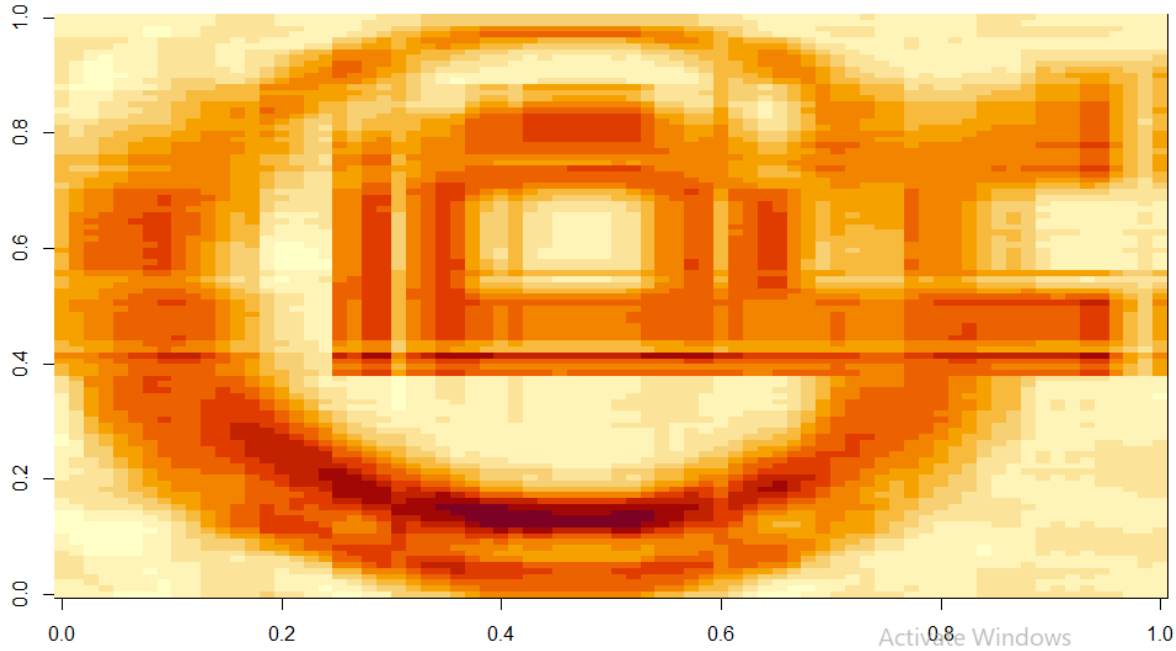
Code :

```
> d5=diag(svd.r$d[1:5])
> dim(d5)
> i5= u5 %*% d5 %*% t(v5)
> image(i5)
```

Output :

```
> d5=diag(svd.r$d[1:5])
> dim(d5)
[1] 5 5
> i5= u5 %*% d5 %*% t(v5)
> image(i5)
```

Image :



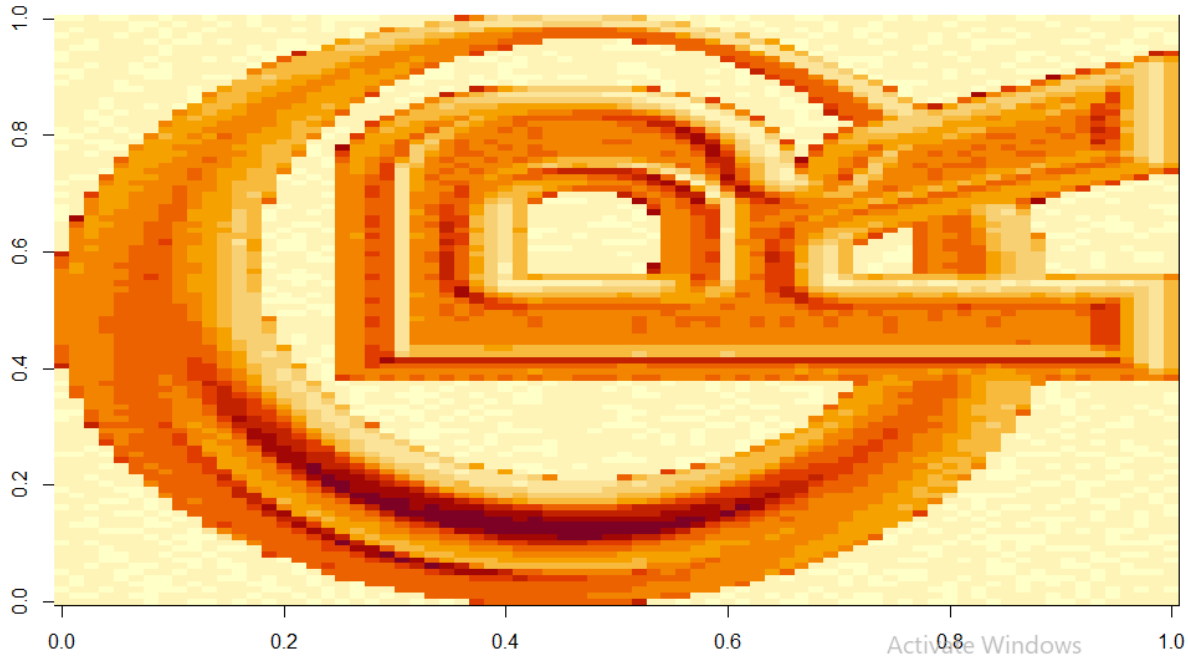
Code:

```
> u20=as.matrix(svd.r$u[,1:50])
> dim(u20)
> v20=as.matrix(svd.r$v[,1:50])
> dim(v20)
> d20=diag(svd.r$d[1:50])
> dim(d20)
> i20= u20 %*% d20 %*% t(v20)
> image(i20)
```

Output :

```
> u20=as.matrix(svd.r$u[,1:50])
> dim(u20)
[1] 76 50
>
> v20=as.matrix(svd.r$v[,1:50])
> dim(v20)
[1] 100  50
> #[1] 442  20
> d20=diag(svd.r$d[1:50])
> dim(d20)
[1] 50 50
> #[1] 20 20
> i20= u20 %*% d20 %*% t(v20)
> image(i20)
```

Image :



In Python :

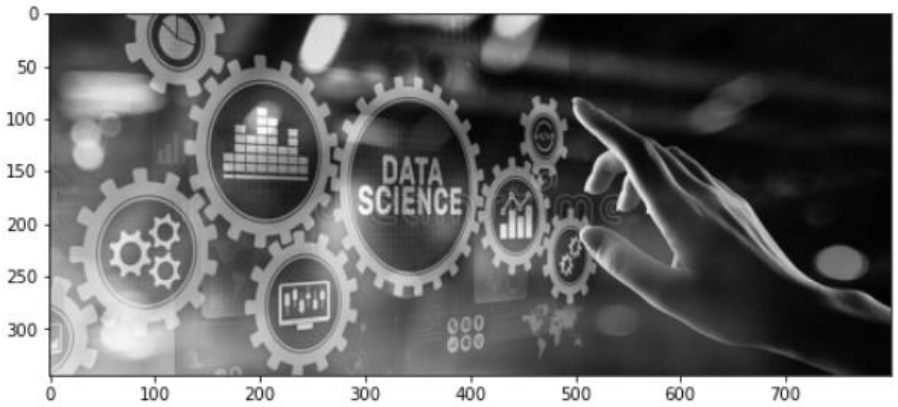
Importing Libraries :

In [1]:

```
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import time

from PIL import Image
```

Importing Image :

In [2]:
```
img = Image.open('data science.jpg')
imggray = img.convert('LA')
plt.figure(figsize=(9, 6))
plt.imshow(imggray);
```



Converting the image data into numpy matrix plotting the result to show the data is unchanged.

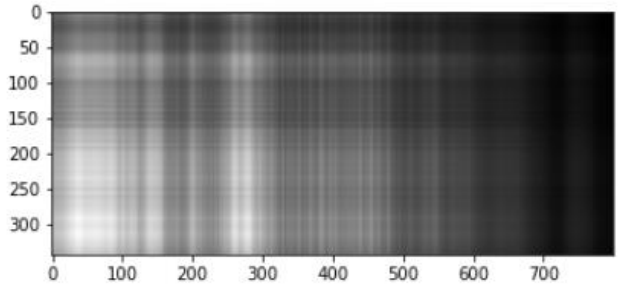
In [3]:
```
imgmat = np.array(list(imggray.getdata(band=0)), float)
imgmat.shape = (imggray.size[1], imggray.size[0])
imgmat = np.matrix(imgmat)
plt.figure(figsize=(9,6))
plt.imshow(imgmat, cmap='gray');
```

Now to compute the singular value decomposition

In [4]:
```
U, sigma, V = np.linalg.svd(imgmat)
```

Computing an approximation of the image

```
In [5]:
1 reconstimg = np.matrix(U[:, :1]) * np.diag(sigma[:1]) * np.matrix(V[:1, :])
2 plt.imshow(reconstimg, cmap='gray');
```



The loop below show that reconstructed image using the first n vectors of the singular value decomposition

```
In [8]:
1 for i in (5, 51, 5):
2     reconstimg = np.matrix(U[:, :i]) * np.diag(sigma[:i]) * np.matrix(V[:i, :])
3     plt.imshow(reconstimg, cmap='gray')
4     title = "n = %s" % i
5     plt.title(title)
6     plt.show()
```