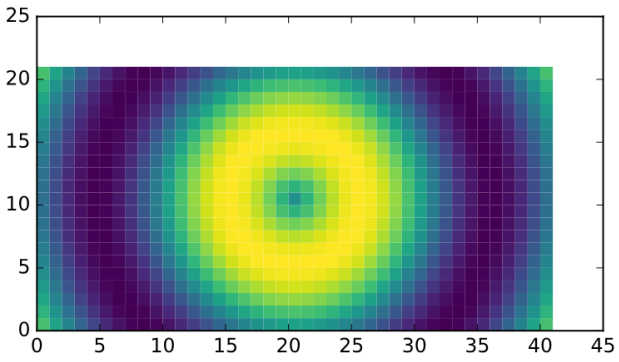
Introduction to Data Visualization with Python

2). Plotting 2D Arrays:

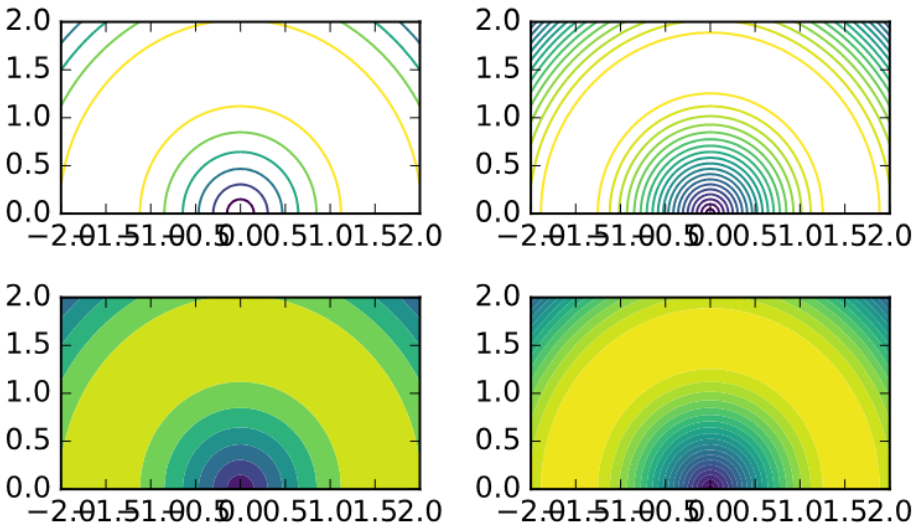
a).

```
# Import numpy and matplotlib.pyplot
import numpy as np
import matplotlib.pyplot as plt


# Generate two 1-D arrays: u, v
u = np.linspace(-2, 2, 41)
v = np.linspace(-1,1,21)


# Generate 2-D arrays from u and v: X, Y
X,Y = np.meshgrid(u,v)


# Compute Z based on X and Y
Z = np.sin(3*np.sqrt(X**2 + Y**2))


# Display the resulting image with pcolor()
plt.pcolor(Z)
plt.show()


# Save the figure to 'sine_mesh.png'
plt.savefig('sine_mesh.png')
```



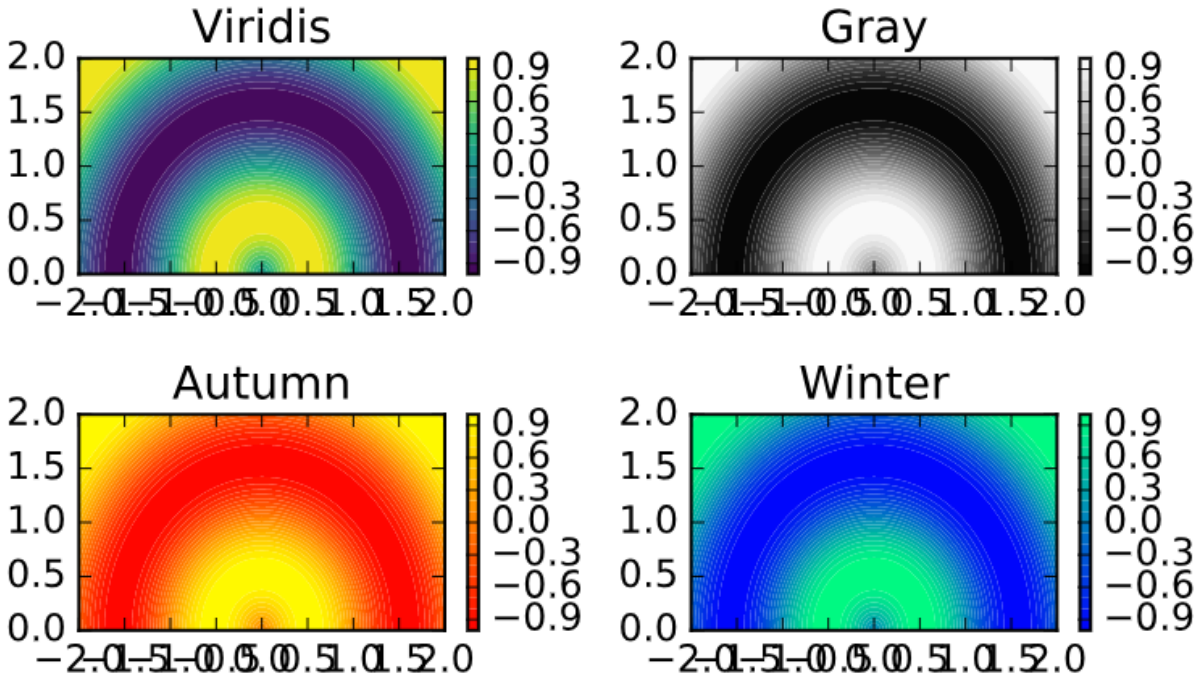
b).

```
#Generate a default contour map of the array Z
plt.subplot(2,2,1)
plt.contour(X,Y,Z)

# Generate a contour map with 20 contours
plt.subplot(2,2,2)
plt.contour(X,Y,Z,20)

# Generate a default filled contour map of the array Z
plt.subplot(2,2,3)
plt.contourf(X,Y,Z)

# Generate a default filled contour map with 20 contours
plt.subplot(2,2,4)
plt.contourf(X,Y,Z,20)

# Improve the spacing between subplots
plt.tight_layout()
# Display the figure
plt.show()
```



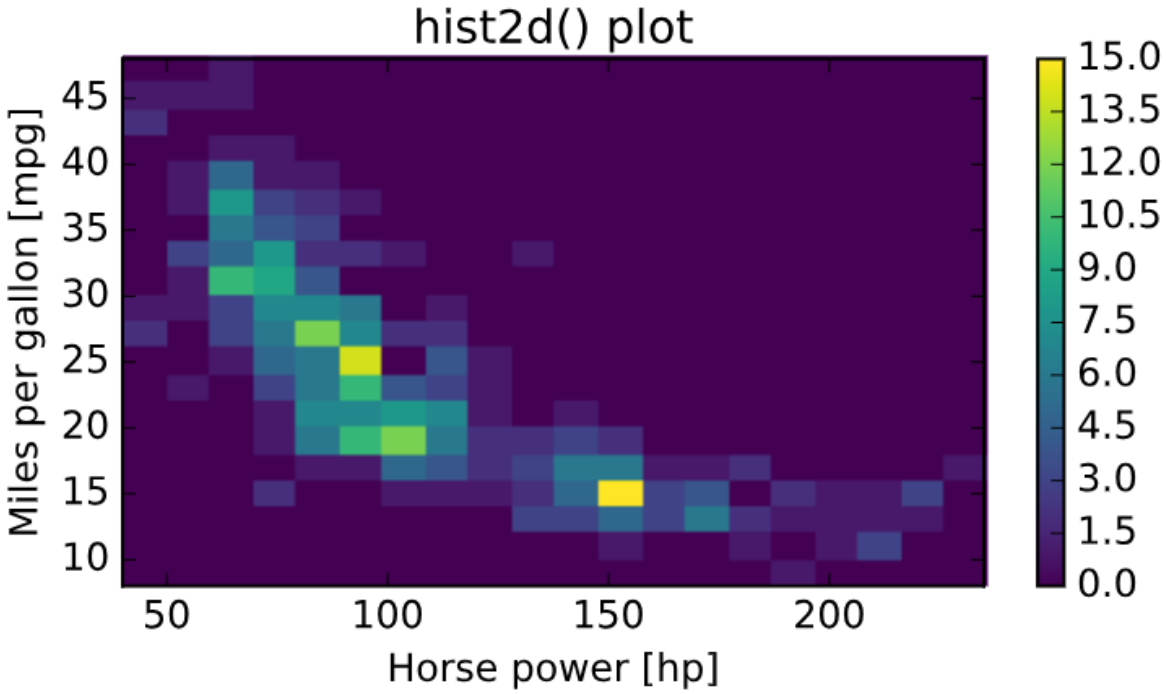
c).

```
# Create a filled contour plot with a color map of 'viridis'
plt.subplot(2,2,1)
plt.contourf(X,Y,Z,20, cmap='viridis')
plt.colorbar()
plt.title('Viridis')

# Create a filled contour plot with a color map of 'gray'
plt.subplot(2,2,2)
plt.contourf(X,Y,Z,20, cmap='gray')
plt.colorbar()
plt.title('Gray')

# Create a filled contour plot with a color map of 'autumn'
plt.subplot(2,2,3)
plt.contourf(X,Y,Z,20, cmap='autumn')
plt.colorbar()
plt.title('Autumn')

# Create a filled contour plot with a color map of 'winter'
plt.subplot(2,2,4)
plt.contourf(X,Y,Z,20, cmap='winter')
plt.colorbar()
plt.title('Winter')

# Improve the spacing between subplots and display them
plt.tight_layout()
plt.show()
```


Viridis
Gray
Autumn
Winter
2.0
1.5
1.0
0.5
0.0
0.9
0.6
0.3
0.0
−0.3
−0.6
−0.9
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0

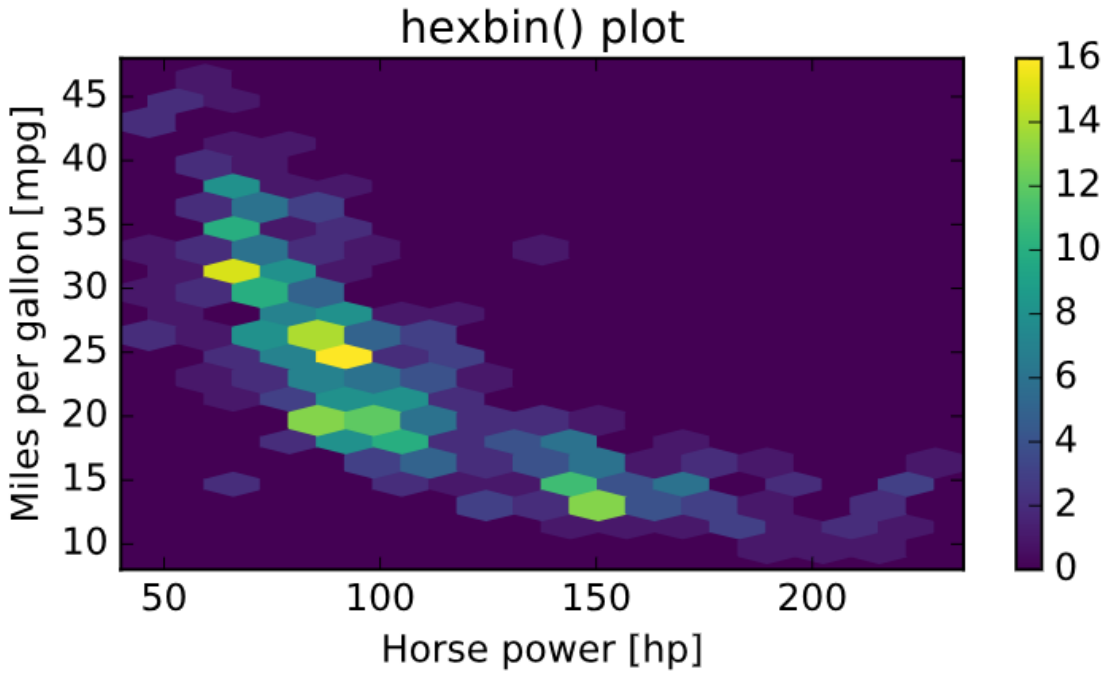
d). HIST2d

```
# Generate a 2-D histogram
plt.hist2d(hp,mpg,bins=(20,20),range=((40,235),(8,48)))

# Add a color bar to the histogram
plt.colorbar()

# Add labels, title, and display the plot
plt.xlabel('Horse power [hp]')
plt.ylabel('Miles per gallon [mpg]')
plt.title('hist2d() plot')
plt.show()
```



e).

```
# Generate a 2d histogram with hexagonal bins
plt.hexbin(hp,mpg,gridsize=(15,12),extent=(40,235,8,48))



# Add a color bar to the histogram
plt.colorbar()


# Add labels, title, and display the plot
plt.xlabel('Horse power [hp]')
plt.ylabel('Miles per gallon [mpg]')
plt.title('hexbin() plot')
plt.show()
```

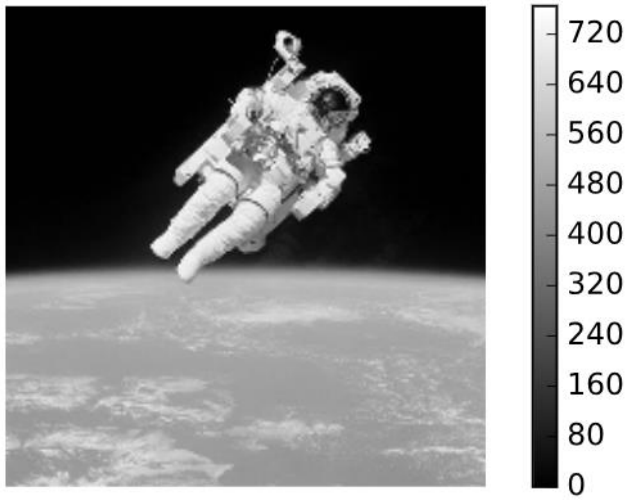


f).

```
# Load the image into an array: img
img = plt.imread('480px-Astronaut-EVA.jpg')

# Print the shape of the image
print(img.shape)

# Display the image
plt.imshow(img)

# Hide the axes
plt.axis('off')
plt.show()
```

g).

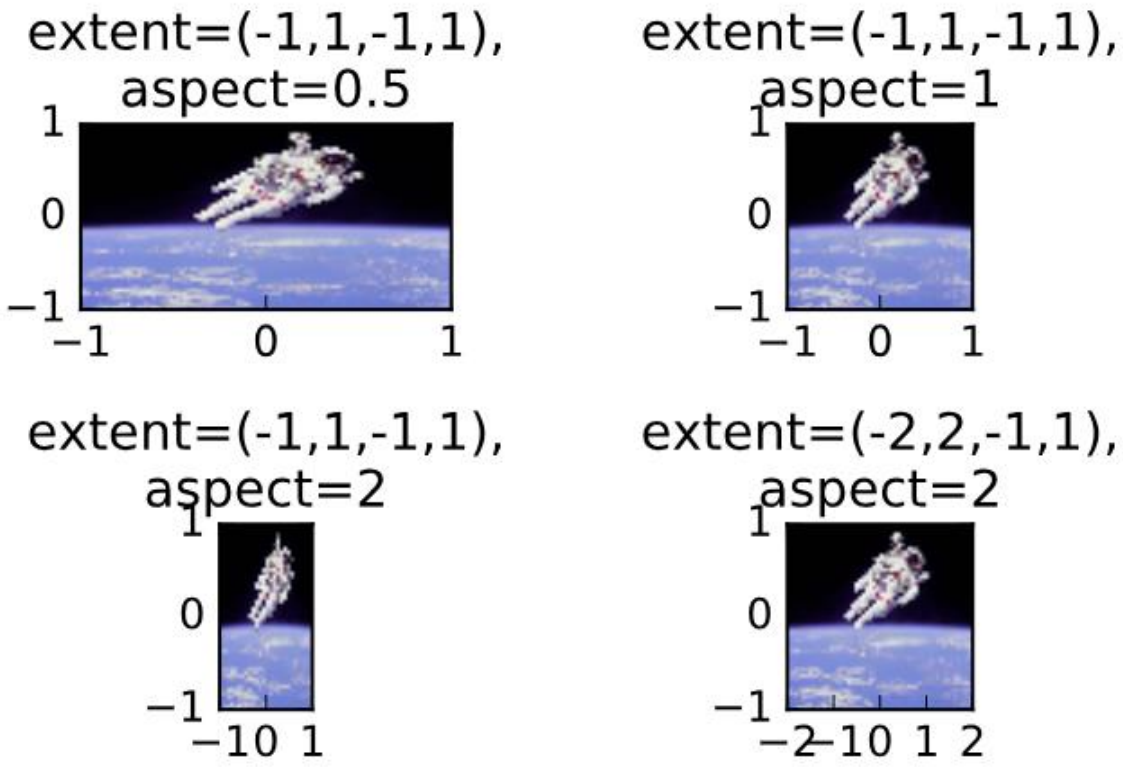
```
# Load the image into an array: img
img = plt.imread('480px-Astronaut-EVA.jpg')

# Print the shape of the image
print(img.shape)

# Compute the sum of the red, green and blue channels: intensity
intensity = img.sum(axis=2)

# Print the shape of the intensity
print(intensity.shape)

# Display the intensity with a colormap of 'gray'
plt.imshow(intensity, cmap='gray')
# Add a colorbar
plt.colorbar()
# Hide the axes and show the figure
plt.axis('off')
plt.show()
```



h).

```
# Load the image into an array: img
img = plt.imread('480px-Astronaut-EVA.jpg')

# Specify the extent and aspect ratio of the top left subplot
plt.subplot(2,2,1)
plt.title('extent=(-1,1,-1,1),\naspect=0.5')
plt.xticks([-1,0,1])
plt.yticks([-1,0,1])
plt.imshow(img, extent=(-1,1,-1,1), aspect=0.5)

# Specify the extent and aspect ratio of the top right subplot
plt.subplot(2,2,2)
plt.title('extent=(-1,1,-1,1),\naspect=1')
plt.xticks([-1,0,1])
plt.yticks([-1,0,1])
plt.imshow(img, extent=(-1,1,-1,1), aspect=1)

# Specify the extent and aspect ratio of the bottom left subplot
plt.subplot(2,2,3)
plt.title('extent=(-1,1,-1,1),\naspect=2')
plt.xticks([-1,0,1])
plt.yticks([-1,0,1])
plt.imshow(img, extent=(-1,1,-1,1), aspect=2)

# Specify the extent and aspect ratio of the bottom right subplot
plt.subplot(2,2,4)
plt.title('extent=(-2,2,-1,1),\naspect=2')
plt.xticks([-2,-1,0,1,2])
plt.yticks([-1,0,1])
plt.imshow(img,extent=(-2,2,-1,1),aspect=2)
```

```
# Improve spacing and display the figure
plt.tight_layout()
plt.show()
```



i).

```
# Load the image into an array: image
image = plt.imread('640px-Unequalized_Hawkes_Bay_NZ.jpg')

# Extract minimum and maximum values from the image: pmin, pmax
pmin, pmax = image.min(), image.max()
print("The smallest & largest pixel intensities are %d & %d." % (pmin, pmax))

# Rescale the pixels: rescaled_image
rescaled_image = 256*(image - pmin) / (pmax - pmin)
print("The rescaled smallest & largest pixel intensities are %.1f & %.1f." %
    (rescaled_image.min(), rescaled_image.max()))

# Display the original image in the top subplot
plt.subplot(2,1,1)
plt.title('original image')
plt.axis('off')
plt.imshow(image)

# Display the rescaled image in the bottom subplot
plt.subplot(2,1,2)
plt.title('rescaled image')
plt.axis('off')
plt.imshow(rescaled_image)

plt.show()
```

original image

rescaled image