Working with 2D arrays

INTRODUCTION TO DATA VISUALIZATION IN PYTHON

Bryan Van de Ven
Core Developer of Bokeh

Reminder: NumPy arrays

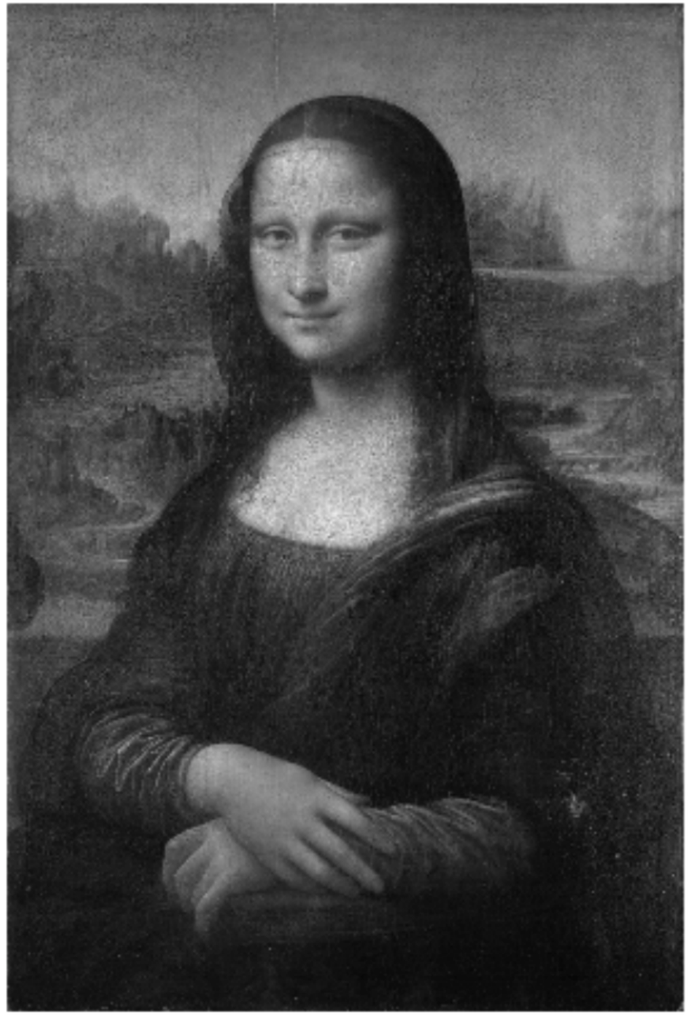
- Homogeneous in type
- Calculations all at once
- Indexing with brackets:
  - `A[index]` for 1D array
  - `A[index0, index1]` for 2D array

Reminder: slicing arrays

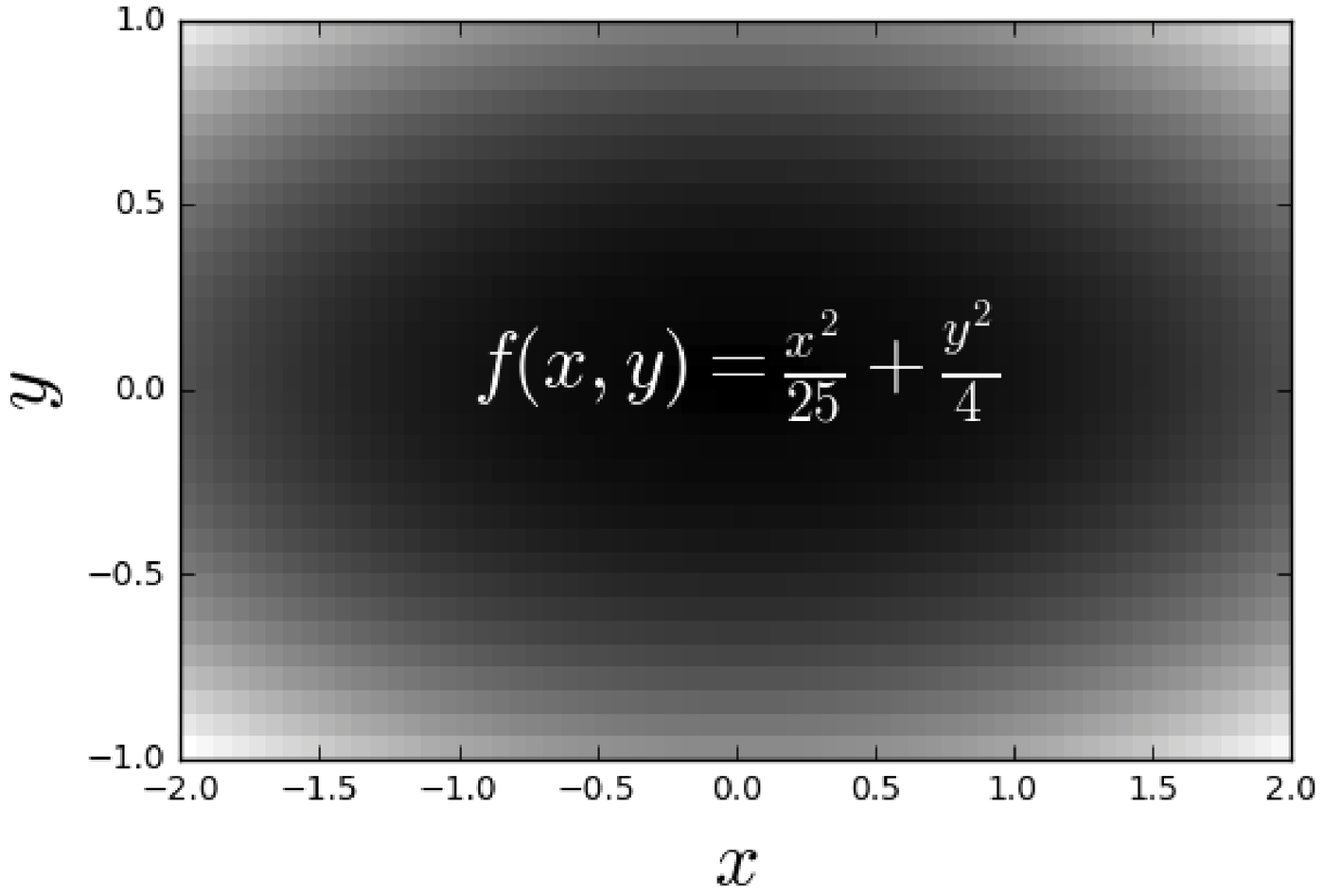
- Slicing: 1D arrays: `A[slice]` , 2D arrays: `A[slice0, slice1]`
- Slicing: `slice = start:stop:stride`
- Indexes from `start` to `stop-1` in steps of `stride`
- Missing `start` : implicitly at beginning of array
- Missing `stop` : implicitly at end of array
- Missing `stride` : implicitly stride 1
- Negative indexes/slices: count from end of array

2D arrays & images

0.434	0.339	0.337	0.367	...
0.434	0.421	0.404	0.395	...
0.350	0.388	0.340	0.340	...
0.328	0.384	0.308	0.308	...
...	...	...	...	...



2D arrays & functions



Using meshgrid()

`meshgrids.py`:

```
import numpy as np
u = np.linspace(-2, 2, 3)
v = np.linspace(-1, 1, 5)
X,Y = np.meshgrid(u, v)
```

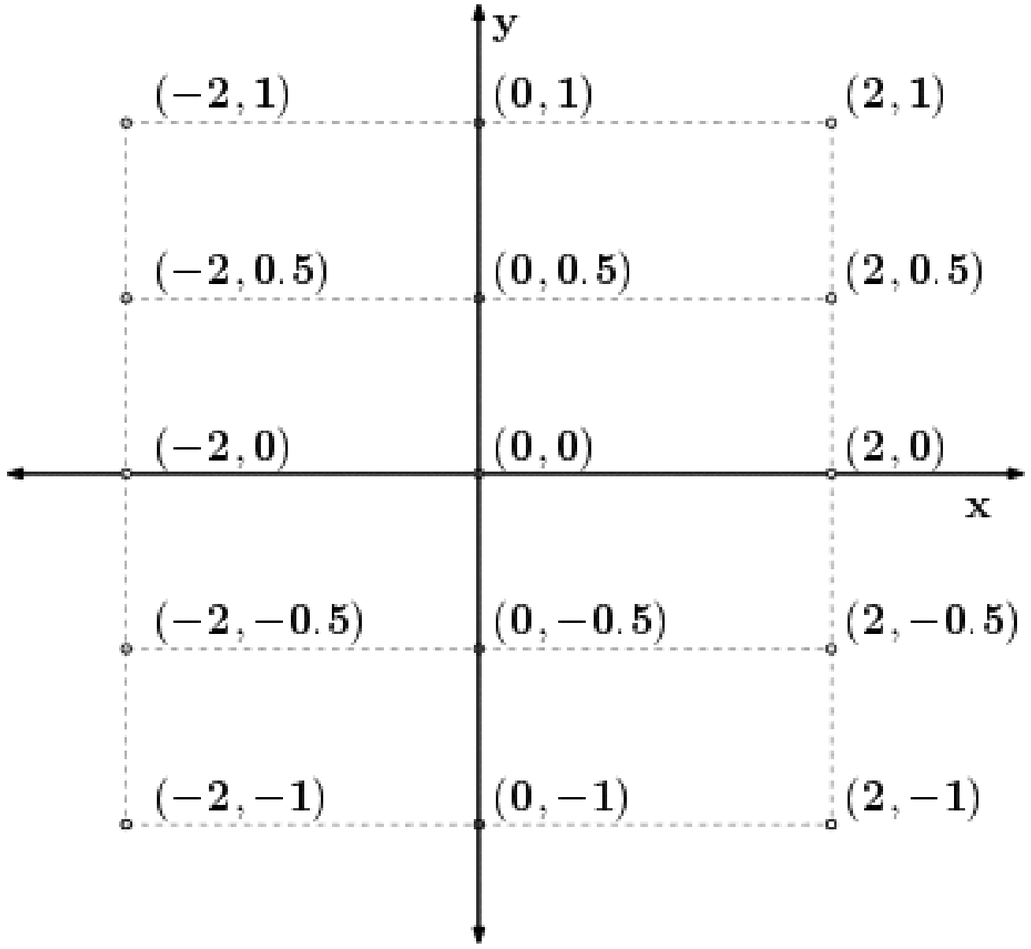
`X`:

```
[[-2.  0.  2.]
 [-2.  0.  2.]
 [-2.  0.  2.]
 [-2.  0.  2.]
 [-2.  0.  2.]]
```

`Y`:

```
[[-1.  -1.  -1. ]
 [-0.5 -0.5 -0.5]
 [ 0.   0.   0. ]
 [ 0.5  0.5  0.5]
 [ 1.   1.   1. ]]
```

Meshgrid



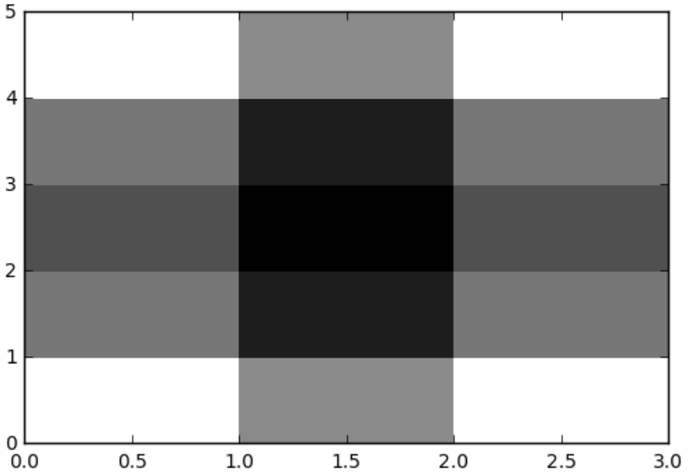
Sampling on a grid

`meshgrids.py`

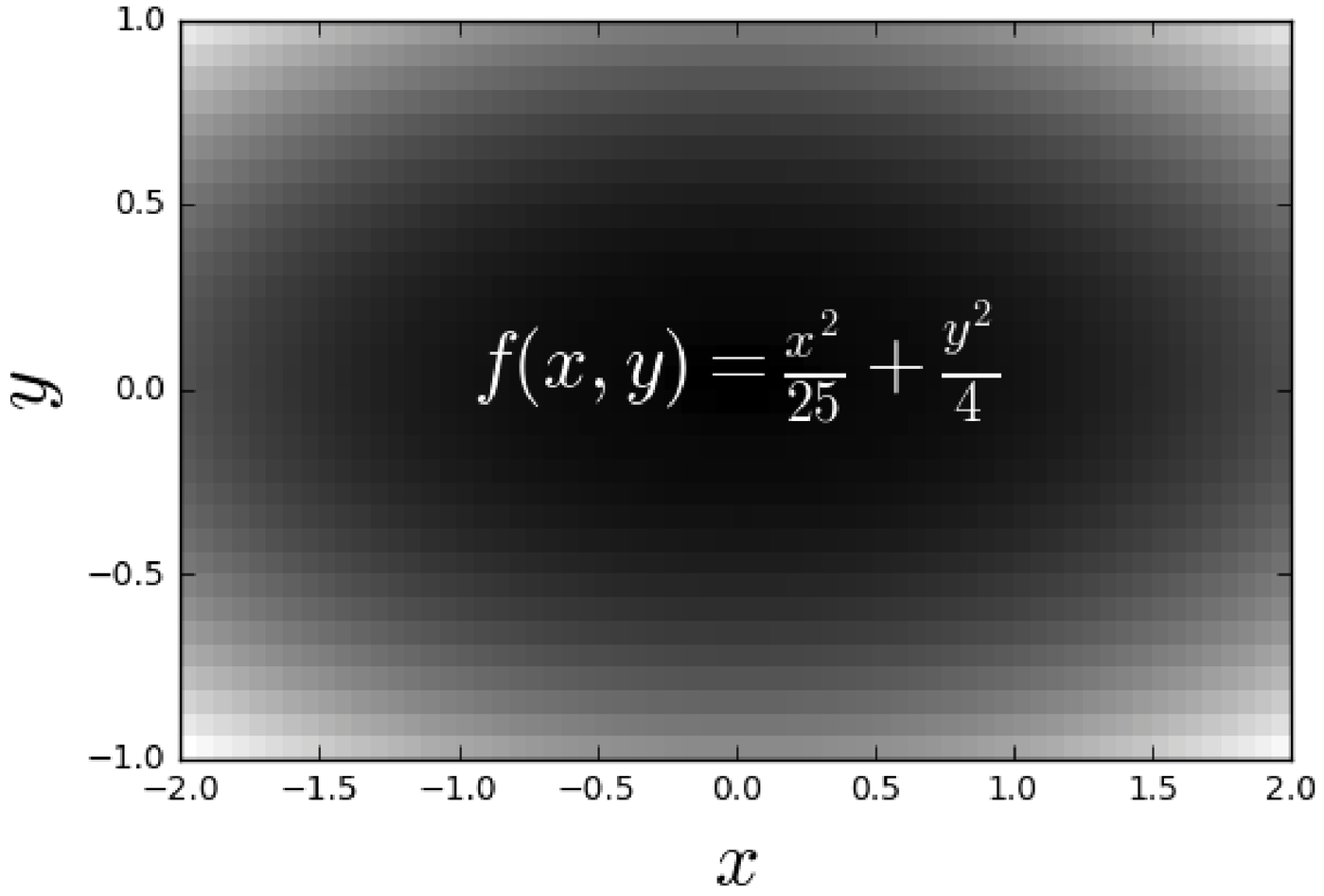
```
import numpy as np
import matplotlib.pyplot as plt

u = np.linspace(-2, 2, 3)
v = np.linspace(-1, 1, 5)
X,Y = np.meshgrid(u, v)

Z = X**2/25 + Y**2/4

print(Z)
plt.set_cmap('gray')
plt.pcolor(Z)
plt.show()
```

`Z`:

```
[[ 0.41    0.25    0.41  ]
[ 0.2225  0.0625  0.2225]
[ 0.16    0.      0.16  ]
[ 0.2225  0.0625  0.2225]
[ 0.41    0.25    0.41  ]]
```



Sampling on a grid



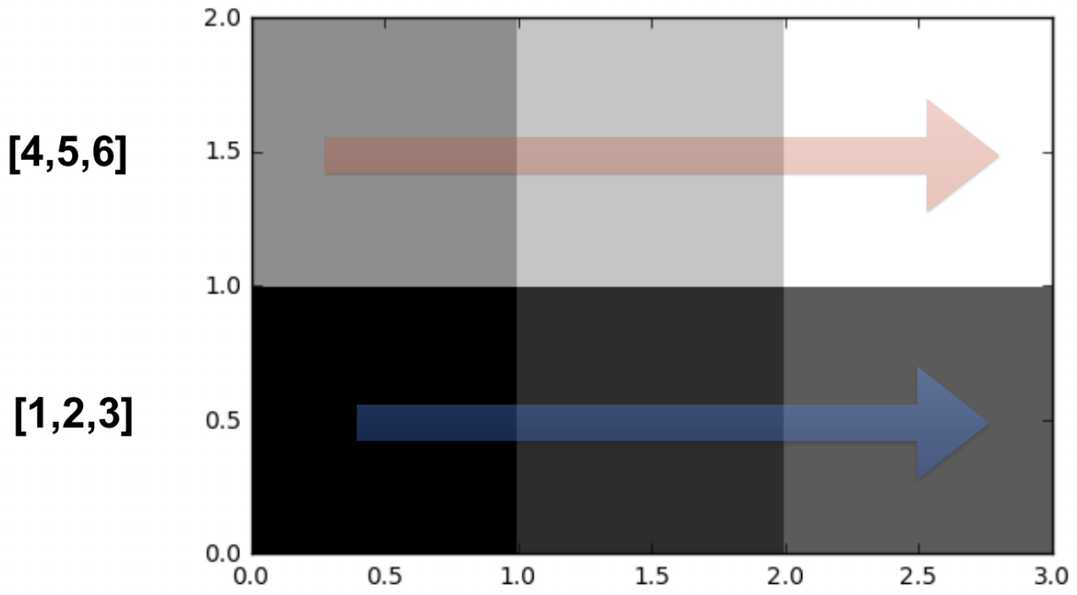
Orientations of 2D arrays & images

`orientation.py`

```
import numpy as np
import matplotlib.pyplot as plt
Z = np.array([[1, 2, 3], [4, 5, 6]])
print(z)
plt.pcolor(Z)
plt.show()
```

`Z`:

```
[[1 2 3]
 [4 5 6]]
```



Let's practice!

INTRODUCTION TO DATA VISUALIZATION IN PYTHON

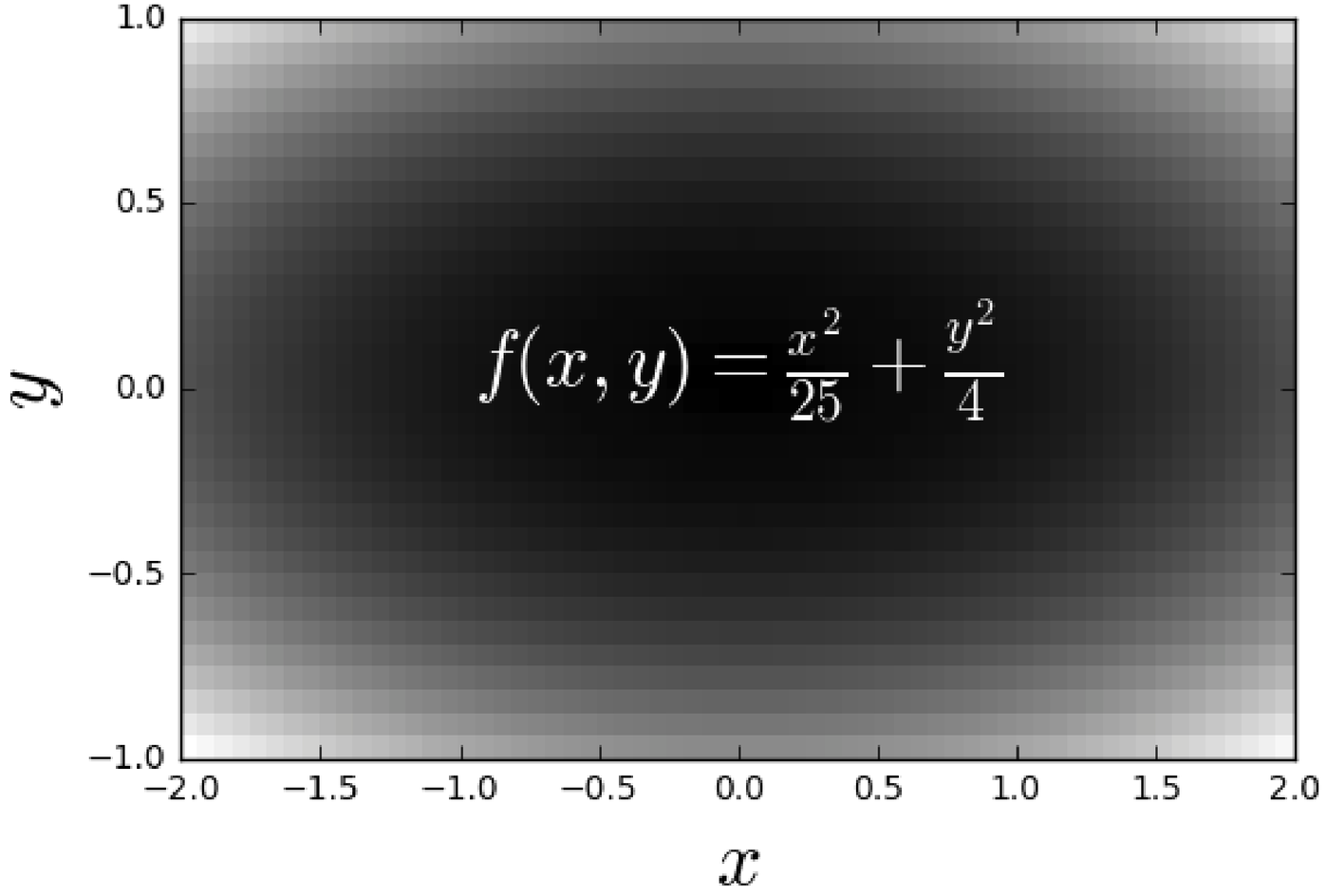
Visualizing bivariate functions

INTRODUCTION TO DATA VISUALIZATION IN PYTHON

Bryan Van de Ven
Core Developer of Bokeh

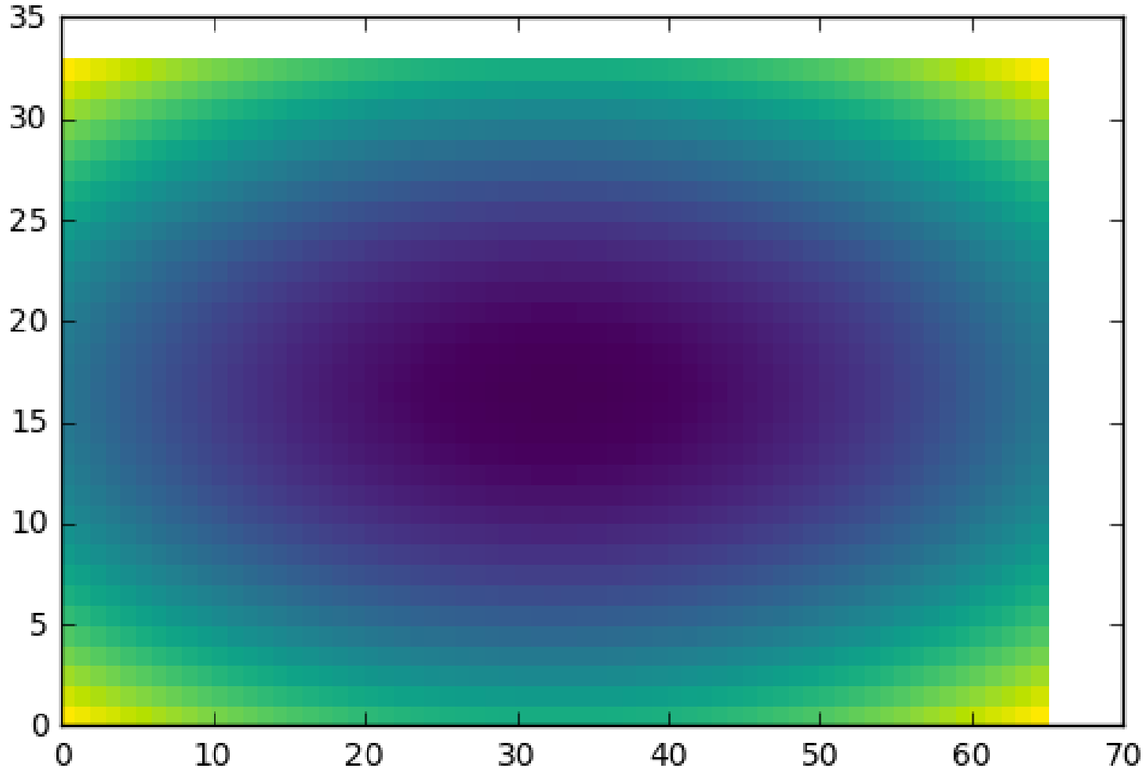
datacamp

Bivariate functions



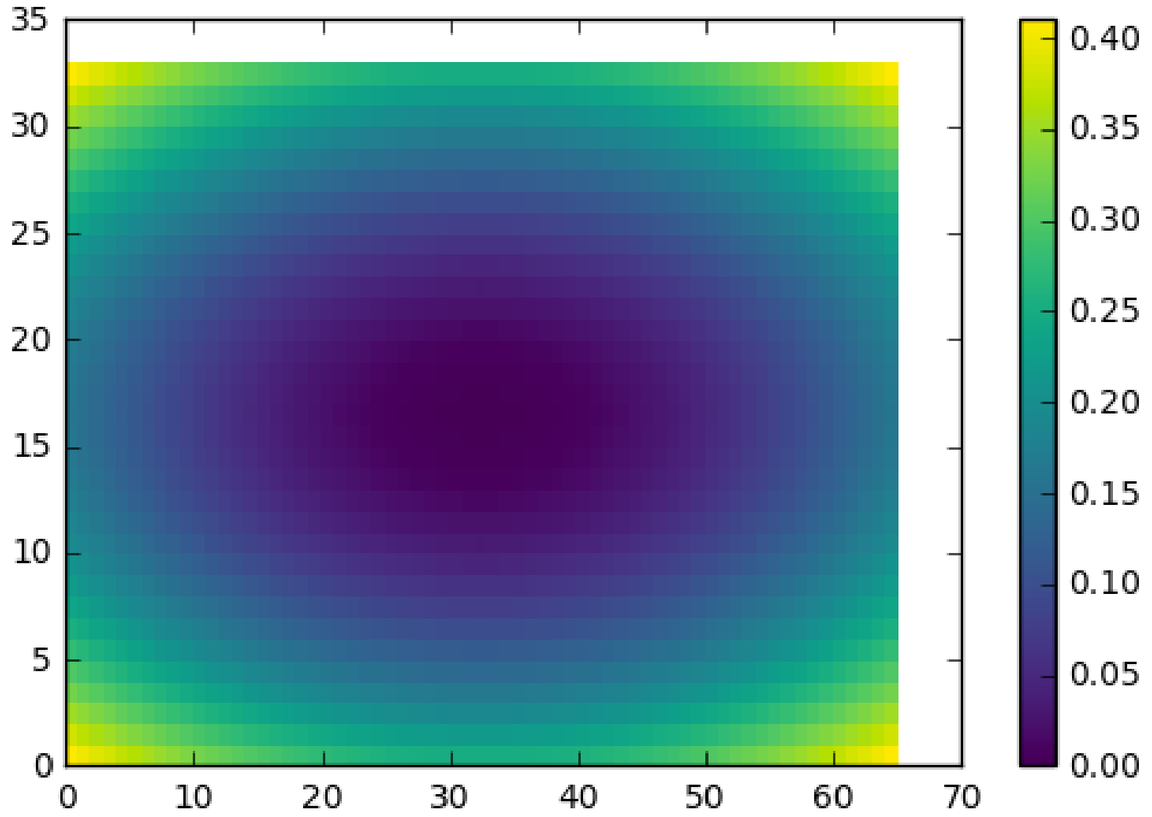
Pseudocolor plot

```
import numpy as np
import matplotlib.pyplot as plt
u = np.linspace(-2, 2, 65)
v = np.linspace(-1, 1, 33)
X,Y = np.meshgrid(u, v)
Z = X**2/25 + Y**2/4
plt.pcolor(Z)
plt.show()
```



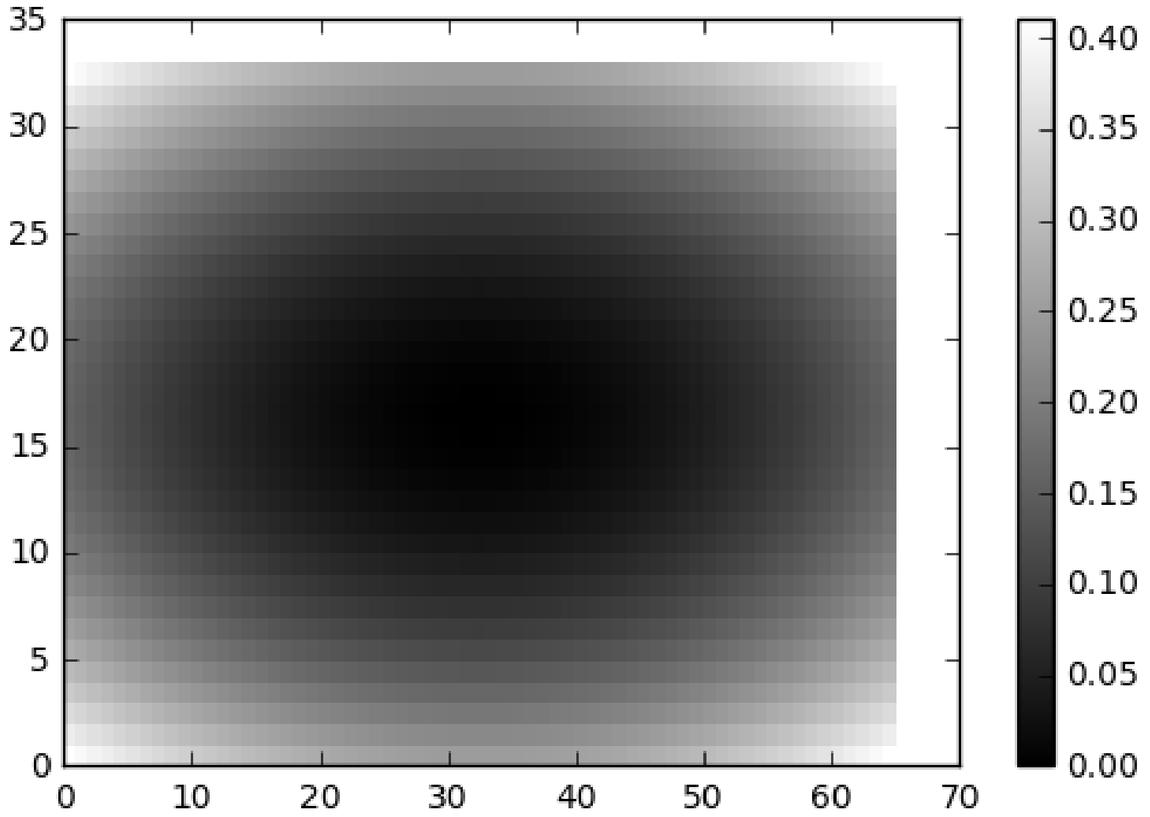
Color bar

```
plt.pcolor(Z)
plt.colorbar()
plt.show()
```



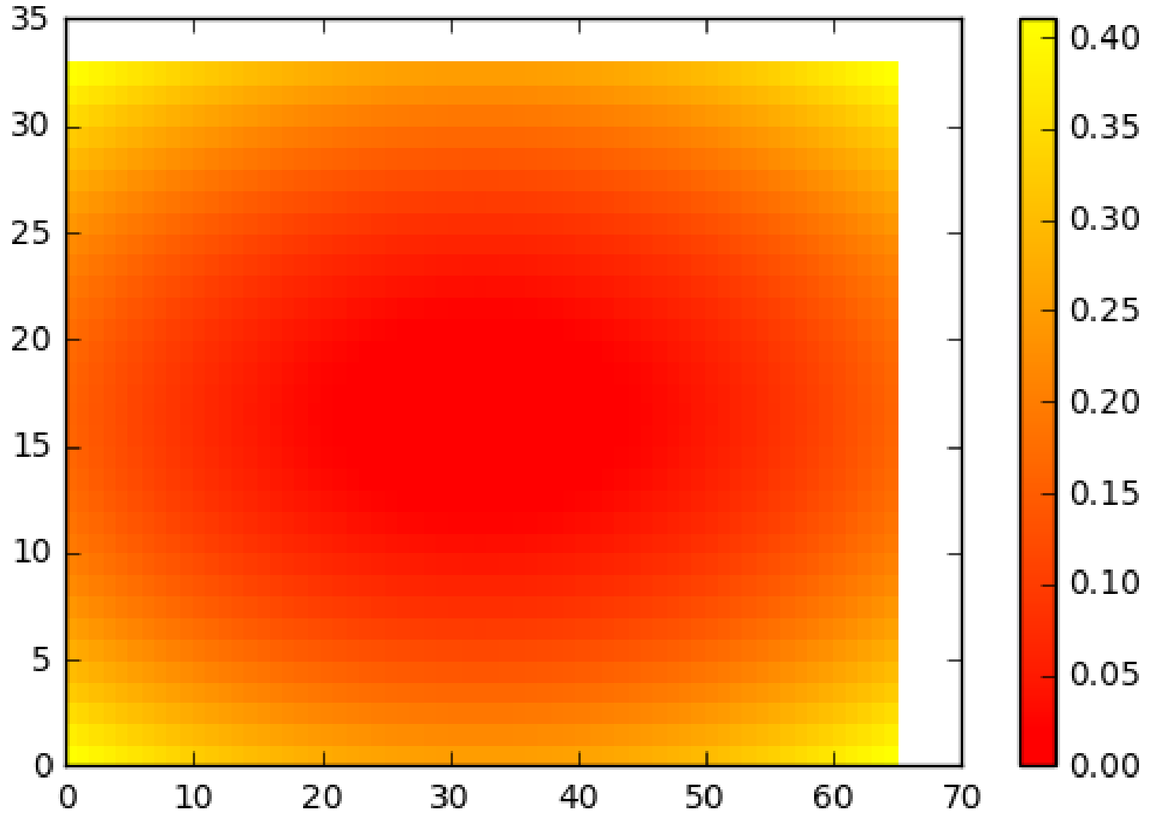
Color map

```
plt.pcolor(Z, cmap= 'gray')
plt.colorbar()
plt.show()
```



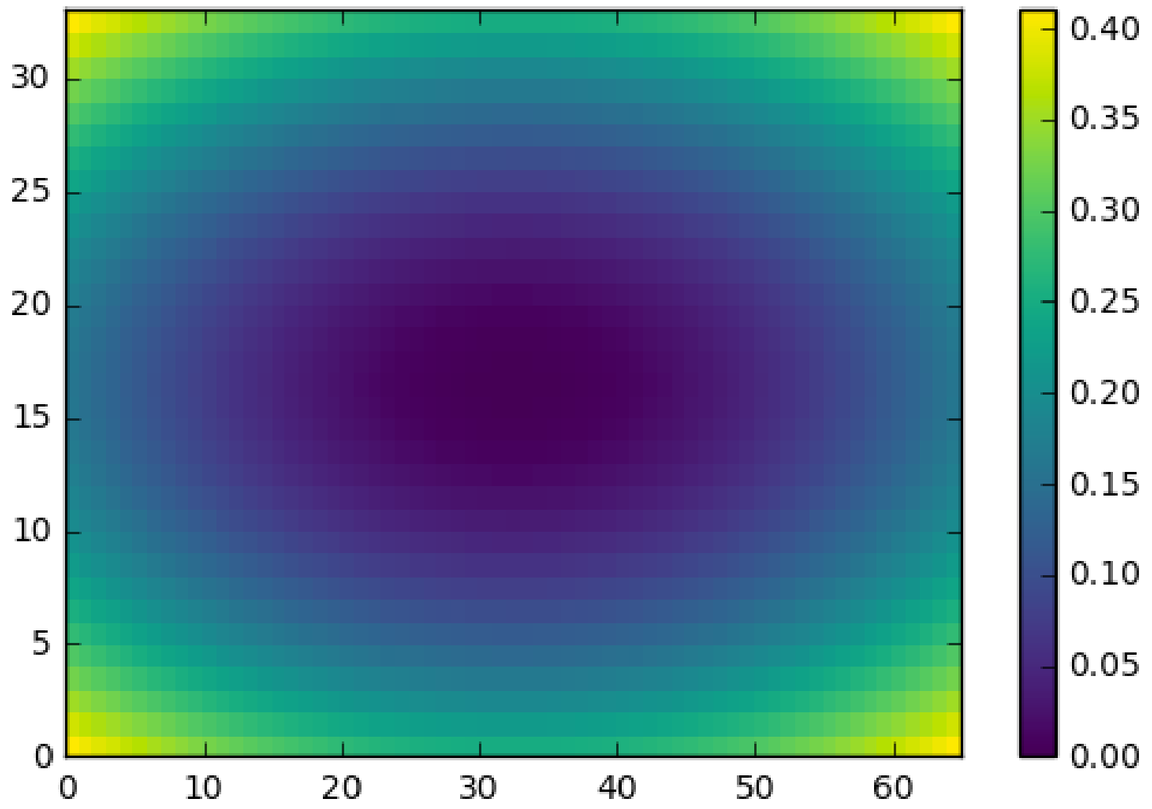
Color map

```
plt.pcolor(Z, cmap= 'autumn')
plt.colorbar()
plt.show()
```



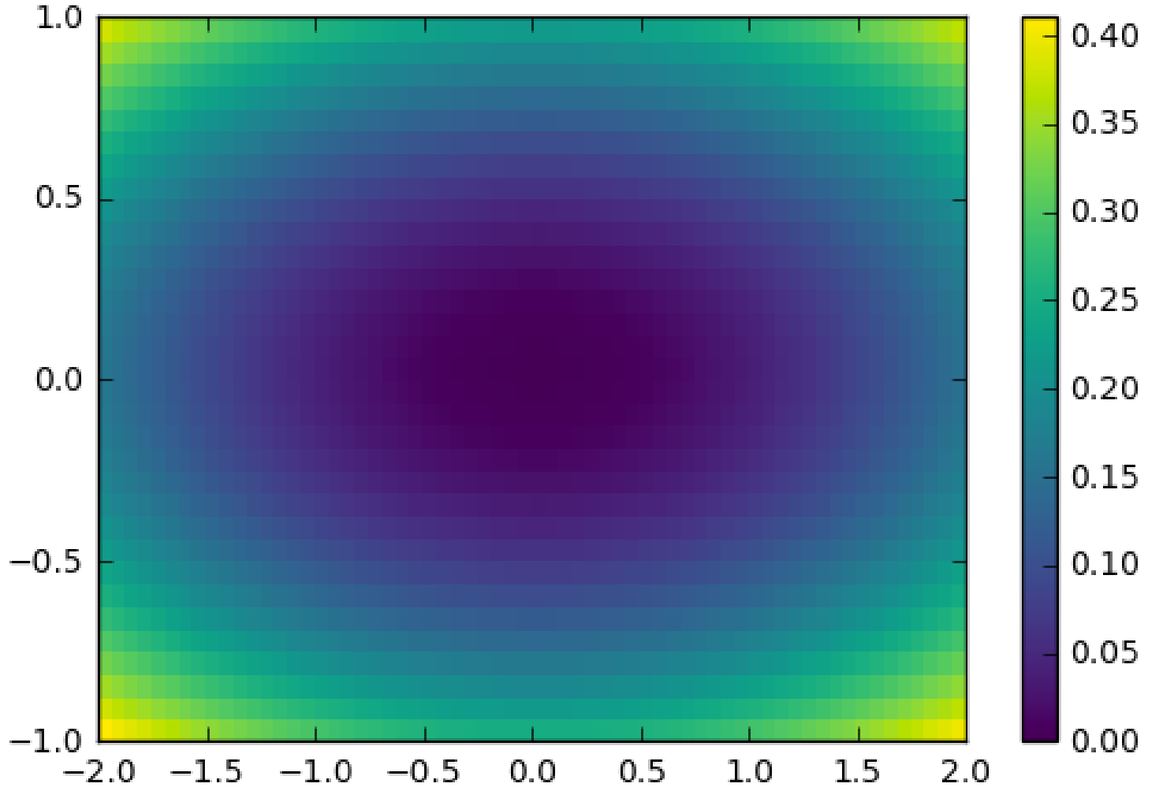
Axis tight

```
plt.pcolor(Z)
plt.colorbar()
plt.axis('tight')
plt.show()
```



Plot using mesh grid

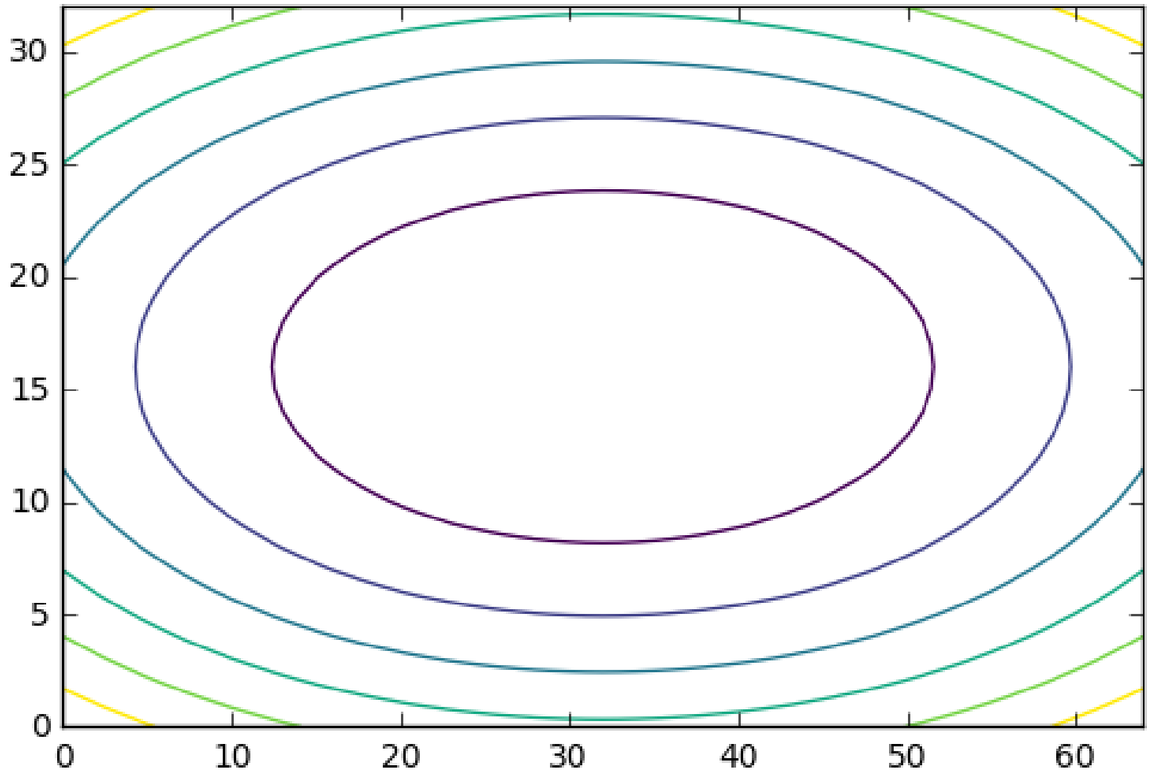
```
# X, Y are 2D meshgrid
plt.pcolor(X, Y, Z)
plt.colorbar()
plt.show()
```



- axes determined by arrays `X` , `Y`

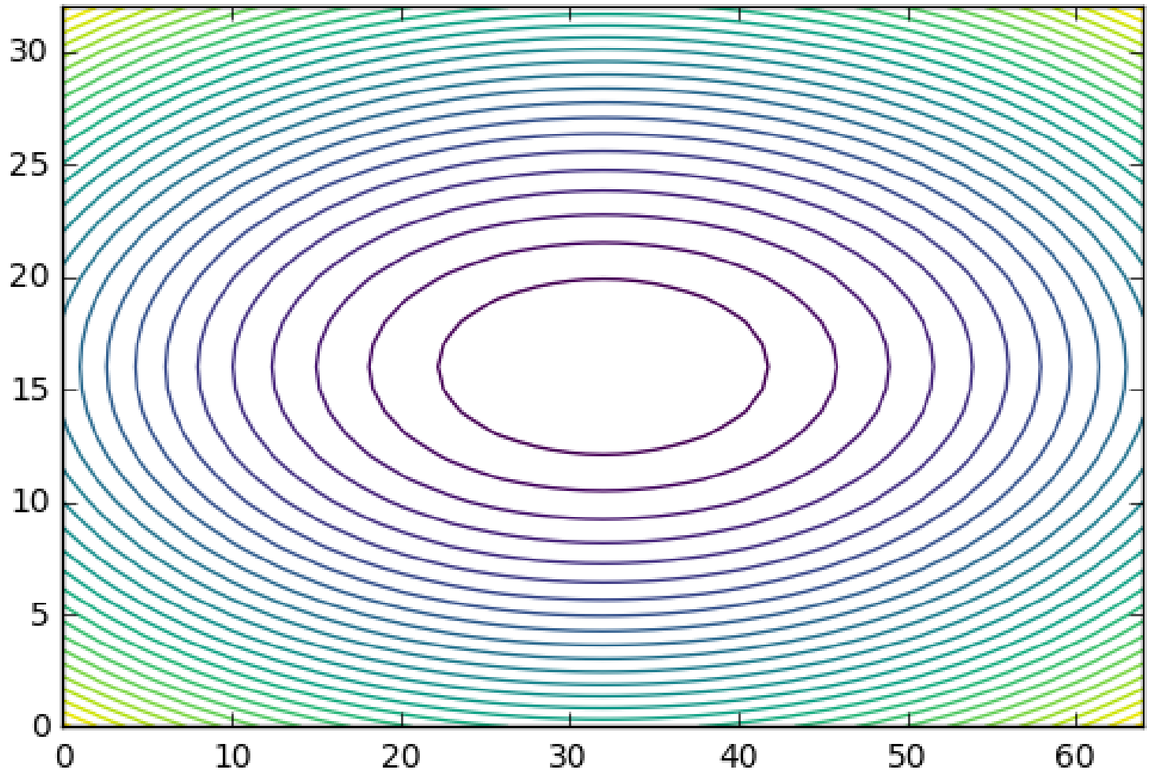
Contour plots

```
plt.contour(Z)
plt.show()
```



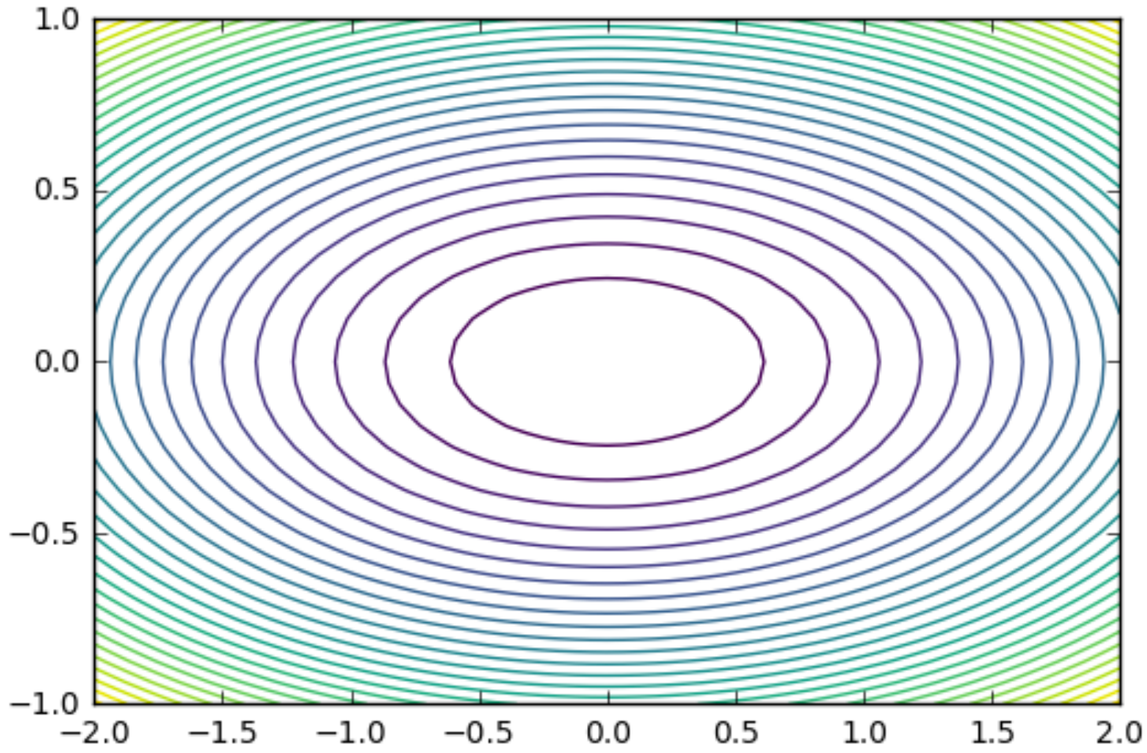
More contours

```
plt.contour(Z, 30)
plt.show()
```



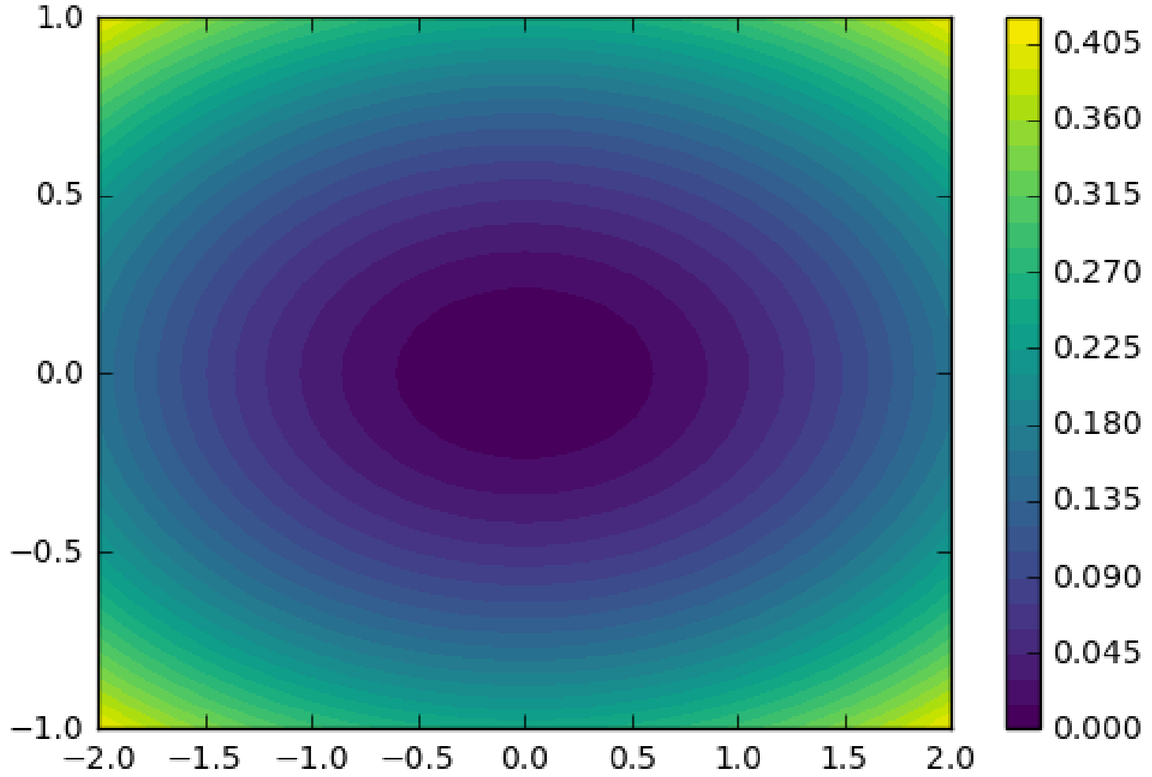
Contour plot using meshgrid

```
plt.contour(X, Y, Z, 30)
plt.show()
```



Filled contour plots

```
plt.contourf(X, Y, Z, 30)
plt.colorbar()
plt.show()
```



More information

- API has many (optional) keyword arguments
- More in matplotlib.pyplot documentation
- More examples: **http://matplotlib.org/gallery.html**

Let's practice!

INTRODUCTION TO DATA VISUALIZATION IN PYTHON

Visualizing bivariate distributions

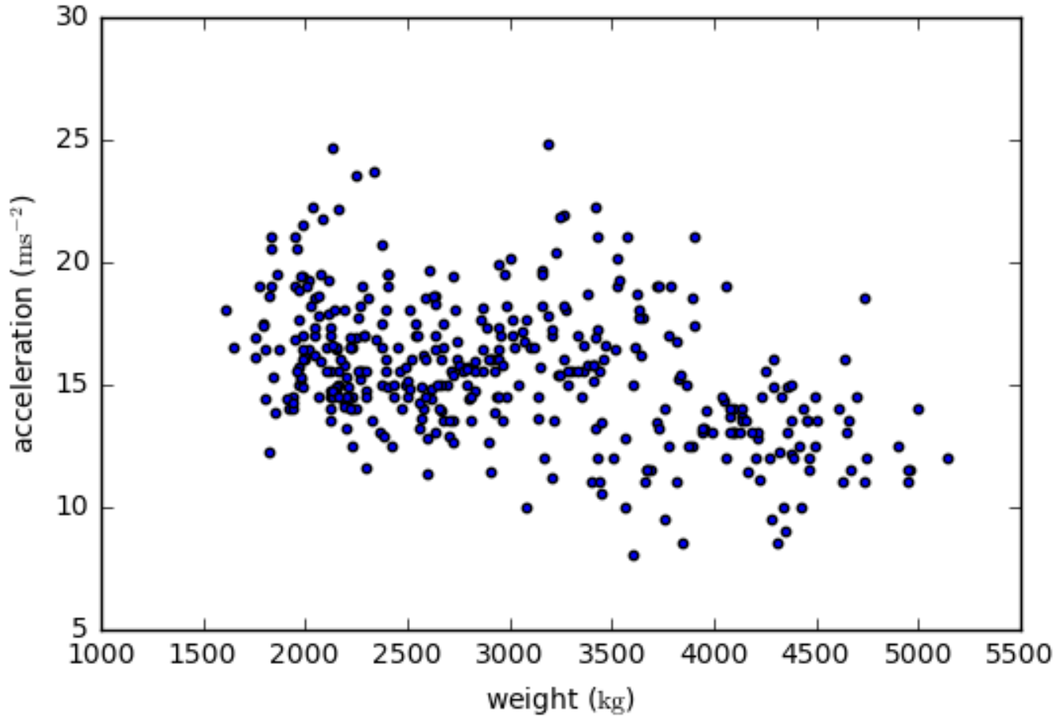
INTRODUCTION TO DATA VISUALIZATION IN PYTHON

Bryan Van de Ven
Core Developer of Bokeh

datacamp

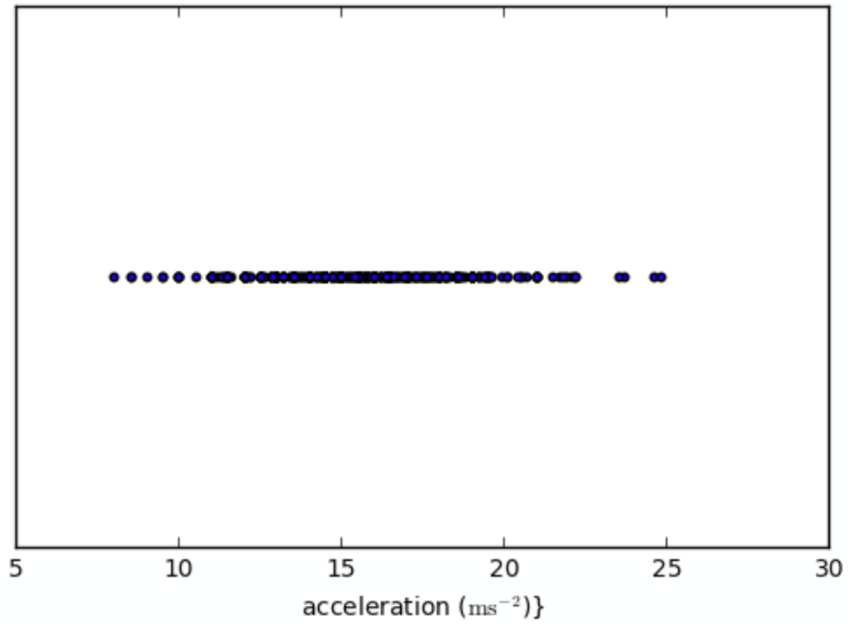
Distributions of 2D points

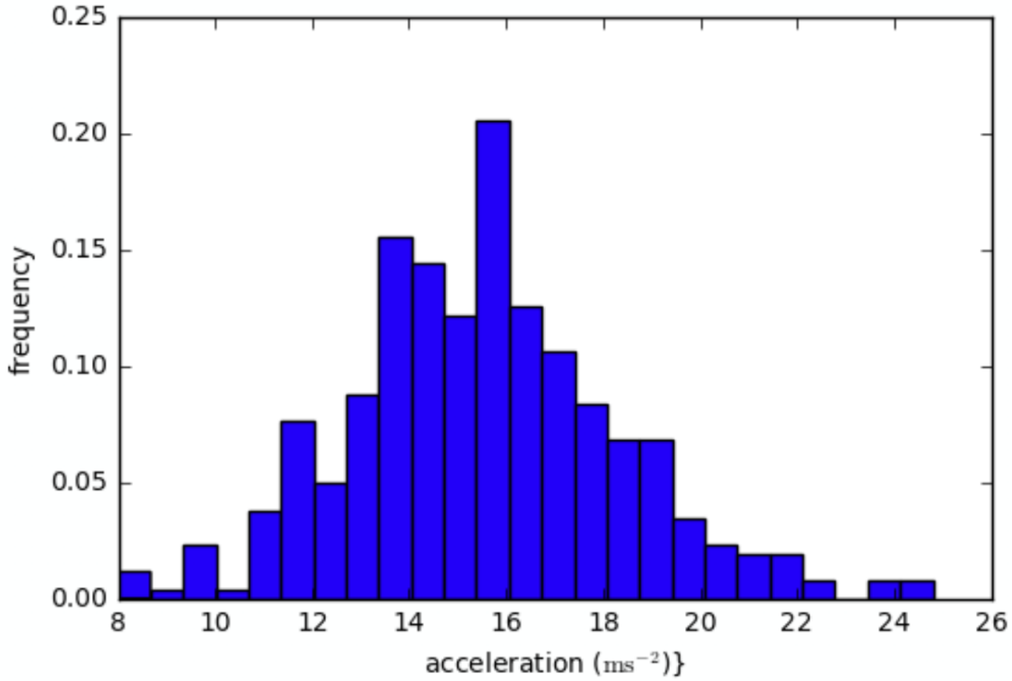
- 2D points given as two 1D arrays `x` and `y`
- Goal: generate a 2D histogram from `x` and `y`



Histograms in 1D

- Choose bins (intervals)
- Count realizations within bins & plot



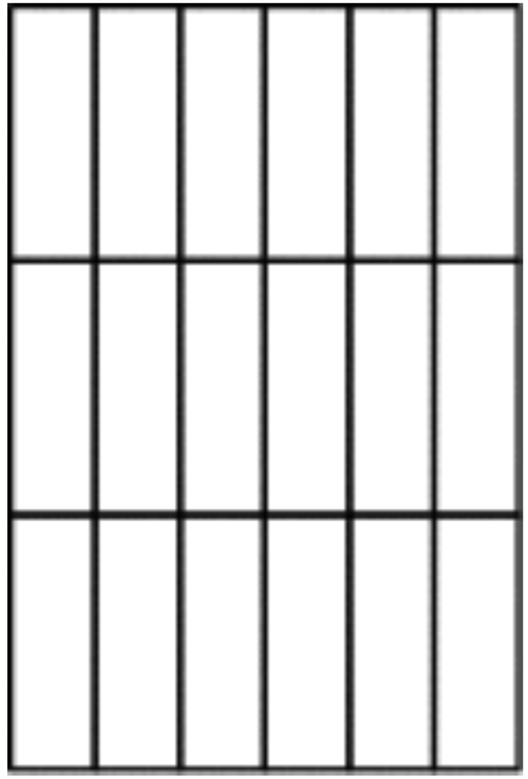


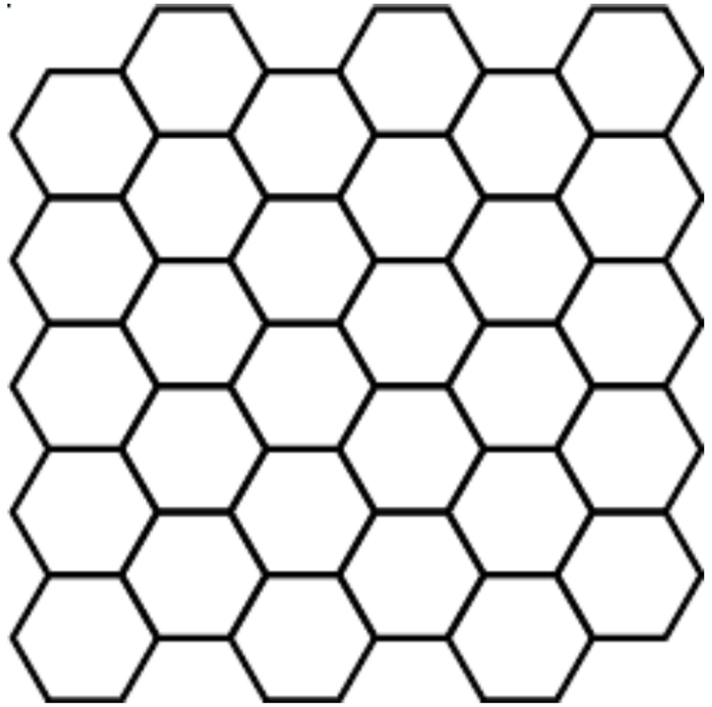
Histograms in 1D

```
counts, bins, patches = plt.hist(x, bins=25)
plt.show()
```

Bins in 2D

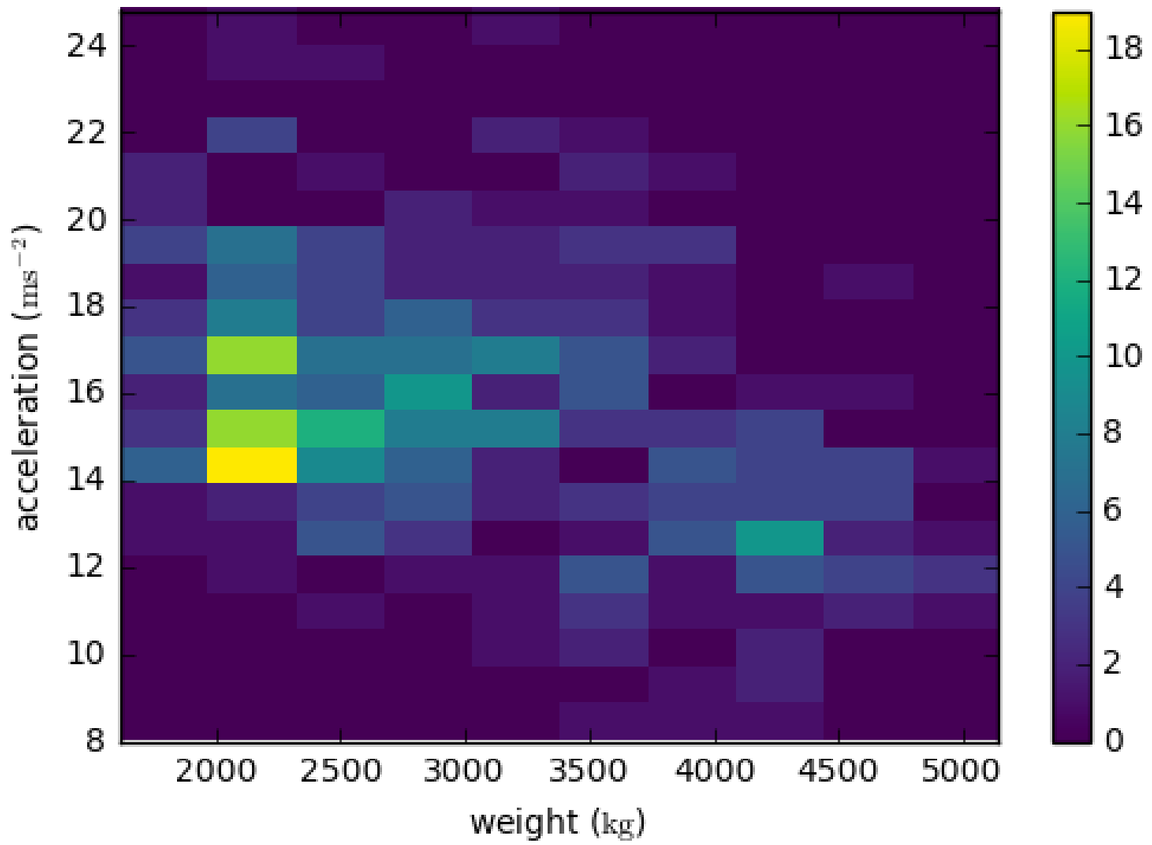
- Different shapes available for binning points
- Common choices: rectangles & hexagons





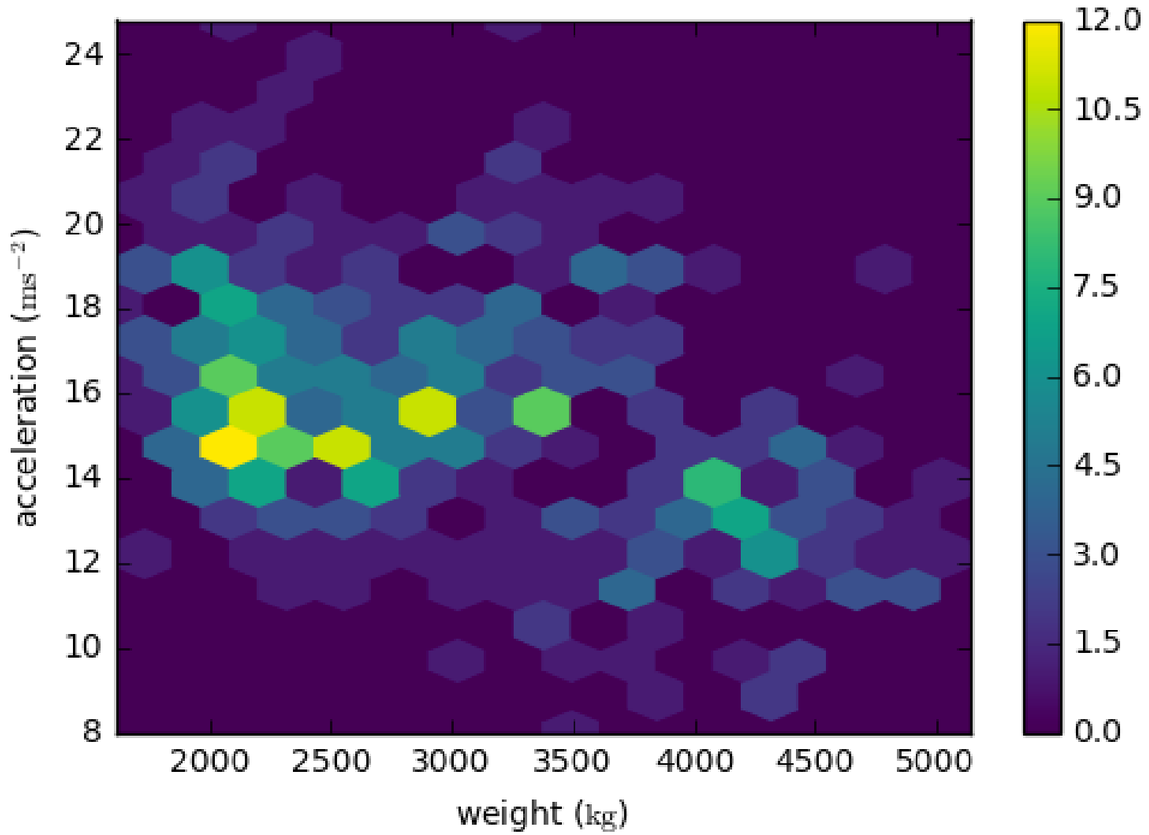
hist2d(): Rectangular binning

```
# x & y are 1D arrays of same length
plt.hist2d(x, y, bins=(10, 20))
plt.colorbar()
plt.xlabel('weight ($\\mathrm{kg}$)')
plt.ylabel('acceleration ($\\mathrm{ms}^{-2}$)}')
plt.show()
```



hexbin(): Hexagonal binning

```
plt.hexbin(x, y, gridsize=(15,10))
plt.colorbar()
plt.xlabel('weight ($\\mathrm{kg}$)')
plt.ylabel('acceleration ($\\mathrm{ms}^{-2}$)}')
plt.show()
```



Let's practice!

INTRODUCTION TO DATA VISUALIZATION IN PYTHON

Working with images

INTRODUCTION TO DATA VISUALIZATION IN PYTHON

Bryan Van de Ven
Core Developer of Bokeh

datacamp

Images

- Grayscale images: rectangular 2D arrays
- Color images: typically three 2D arrays (channels)
  - RGB (Red-Green-Blue)
  - Channel values:
    - 0 to 1 (floating-point numbers)
    - 0 to 255 (8 bit integers)

Loading images

```
img = plt.imread('sunflower.jpg')
print(img.shape)
```

```
(480, 640, 3)
```

```
plt.imshow(img)
plt.axis('off')
plt.show()
```

Reduction to gray-scale image

```
collapsed = img.mean(axis=2)
print(collapsed.shape)
```

```
(480, 640)
```

```
plt.set_cmap('gray')
plt.imshow(collapsed, cmap='gray')
plt.axis('off')
plt.show()
```

Uneven samples

```
# nonuniform subsampling
uneven = collapsed[::4,::2]
print(uneven.shape)
```

```
(120, 320)
```

```
plt.imshow(uneven)
plt.axis('off')
plt.show()
```

Adjusting aspect ratio

```
plt.imshow(uneven, aspect=2.0)
plt.axis('off')
plt.show()
```

Adjusting extent

```
plt.imshow(uneven, cmap='gray',
           extent=(0,640,0,480))
plt.axis('off')
plt.show()
```

Let's practice!

INTRODUCTION TO DATA VISUALIZATION IN PYTHON