



# GlobalLogic<sup>®</sup>

## Python in Science and Engineering

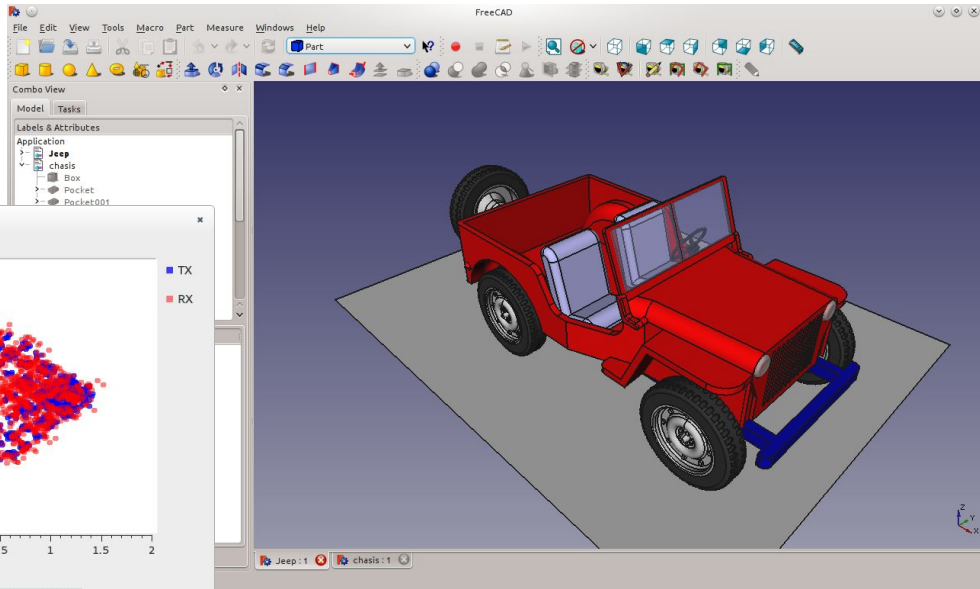
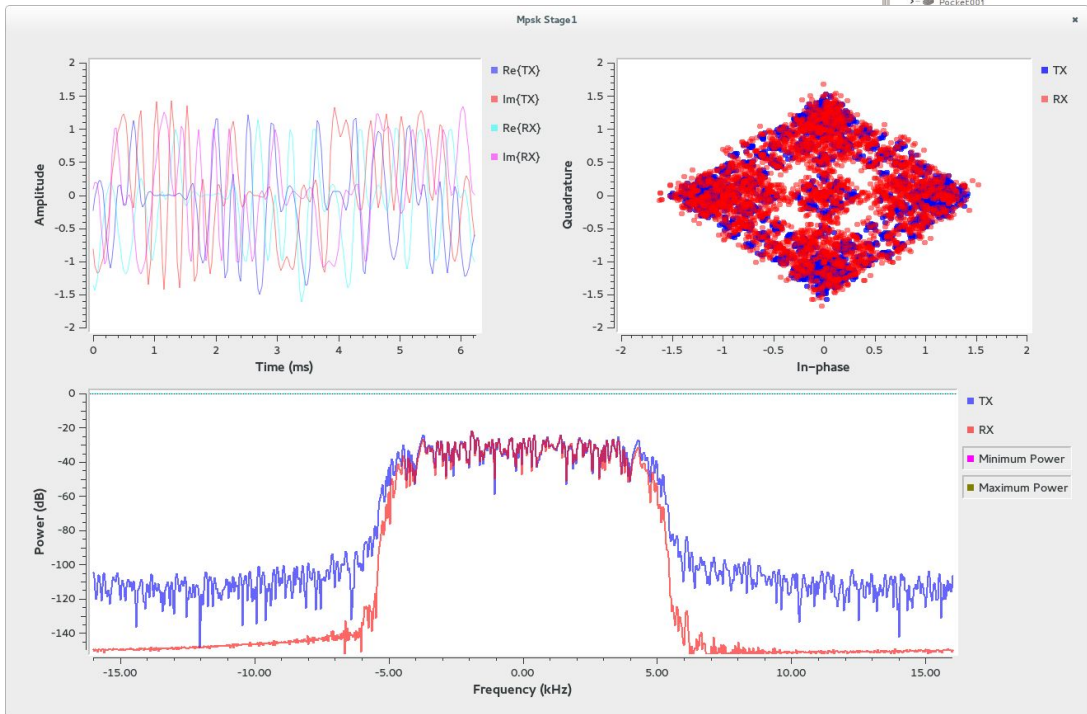
Vasyl Vovk, Consultant, Globallogic

## Agenda

1. Topic overview;
2. Super powerful Calculator;
3. Python Libraries for Science and Engineering:
  - 3.1. Sympy;
  - 3.2. Numpy + Matplotlib;
  - 3.3. SciPy;
  - 3.4. ObsPy;
  - 3.5. AstroPy;
  - 3.6. PyEphem;
  - 3.7. PyRAF.

# Topic overview

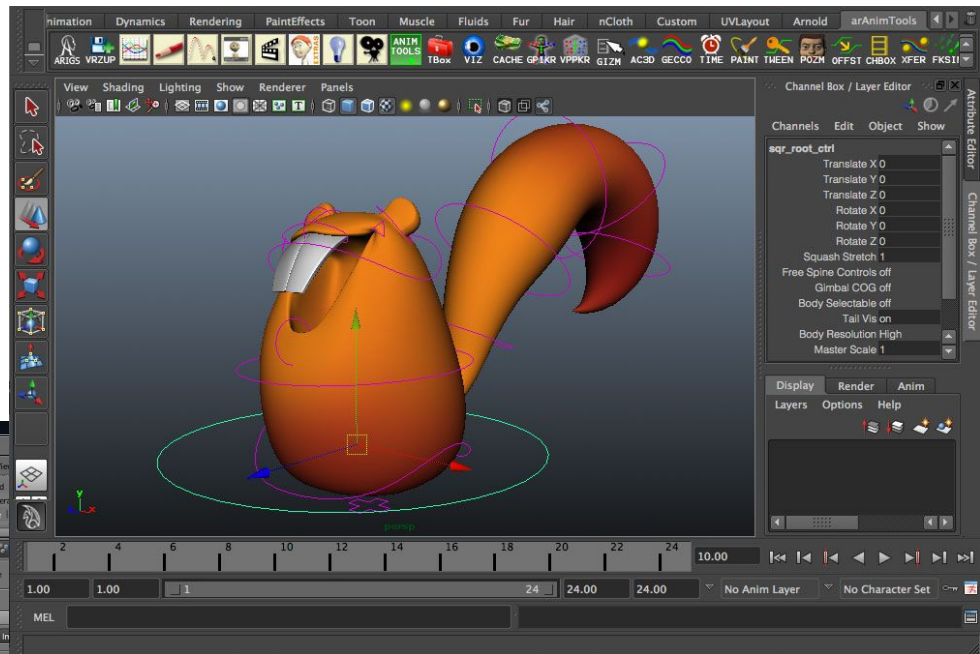
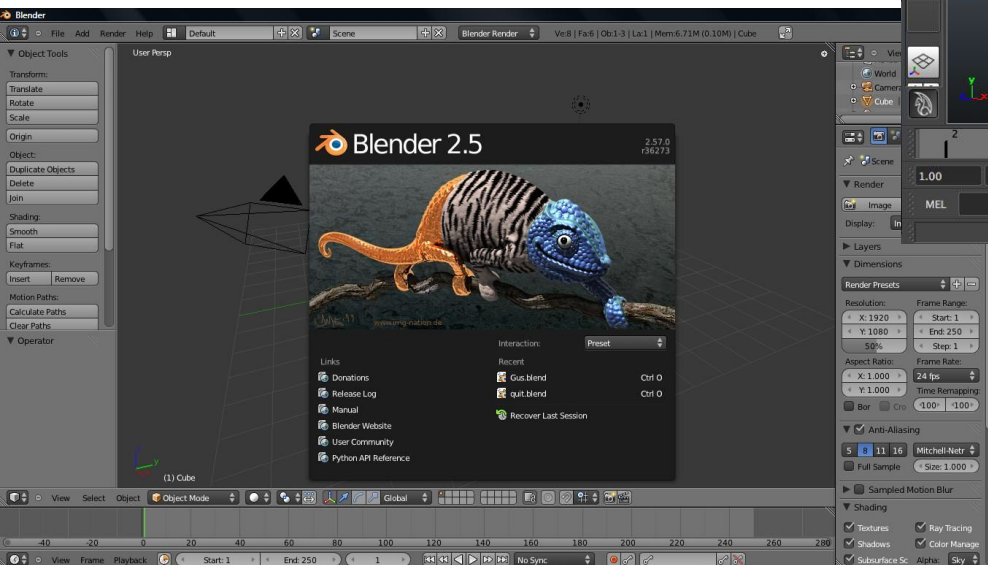
# GNU Radio



FreeCAD

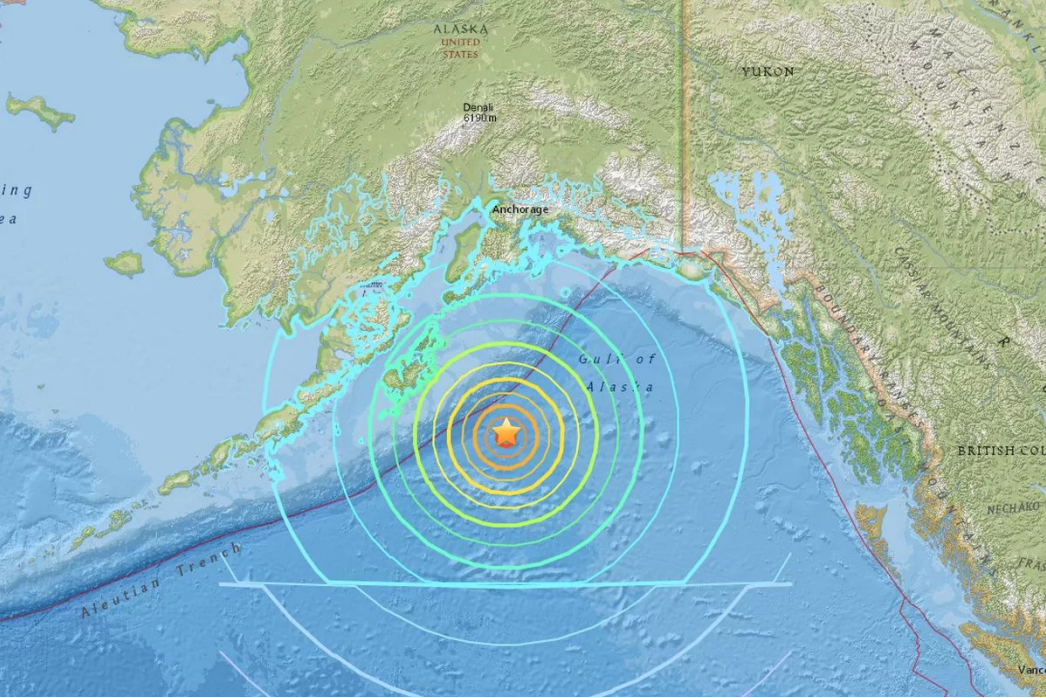
# Computer Graphics

# Blender

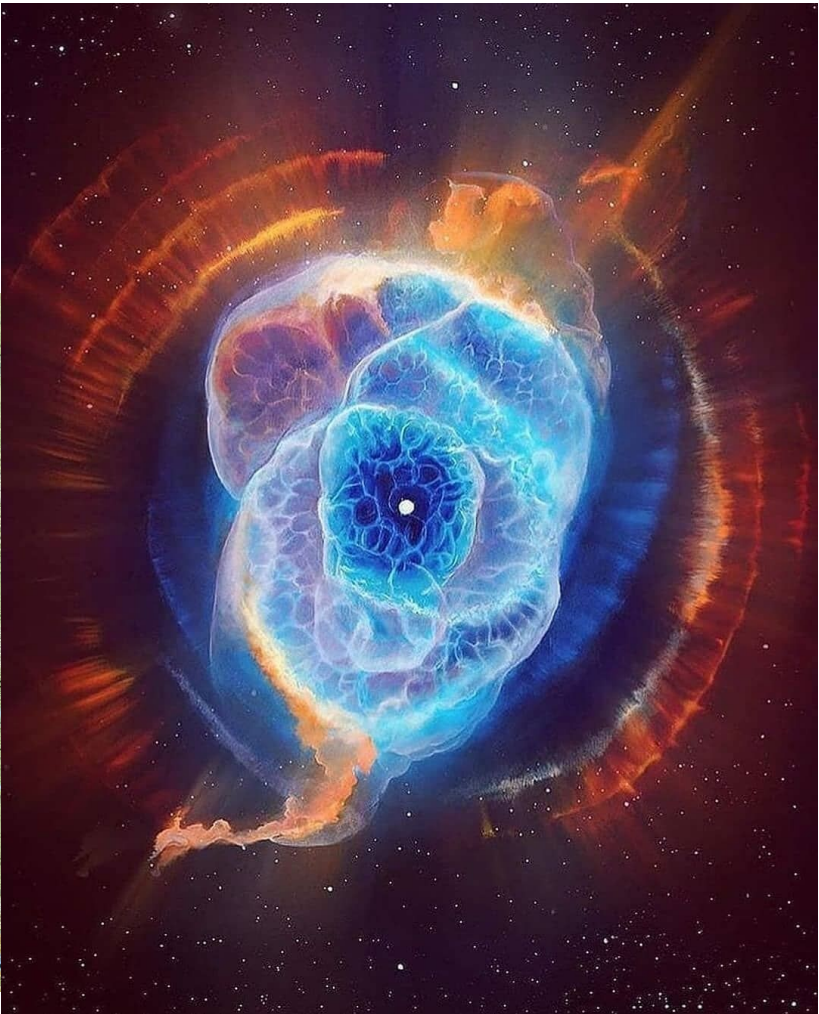


# Autodesk Maya

# Science



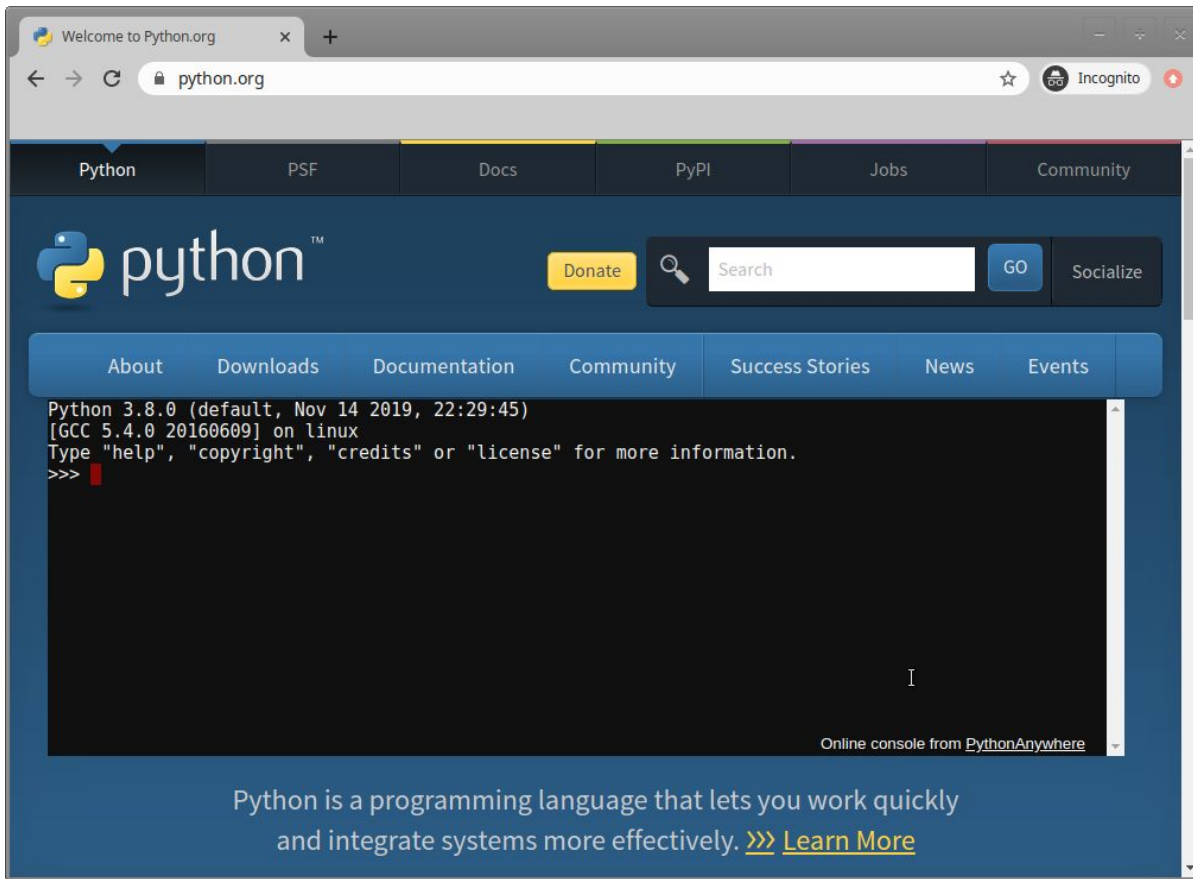
Confidential



# Python CLI as Calculator

# Python CLI as Calculator

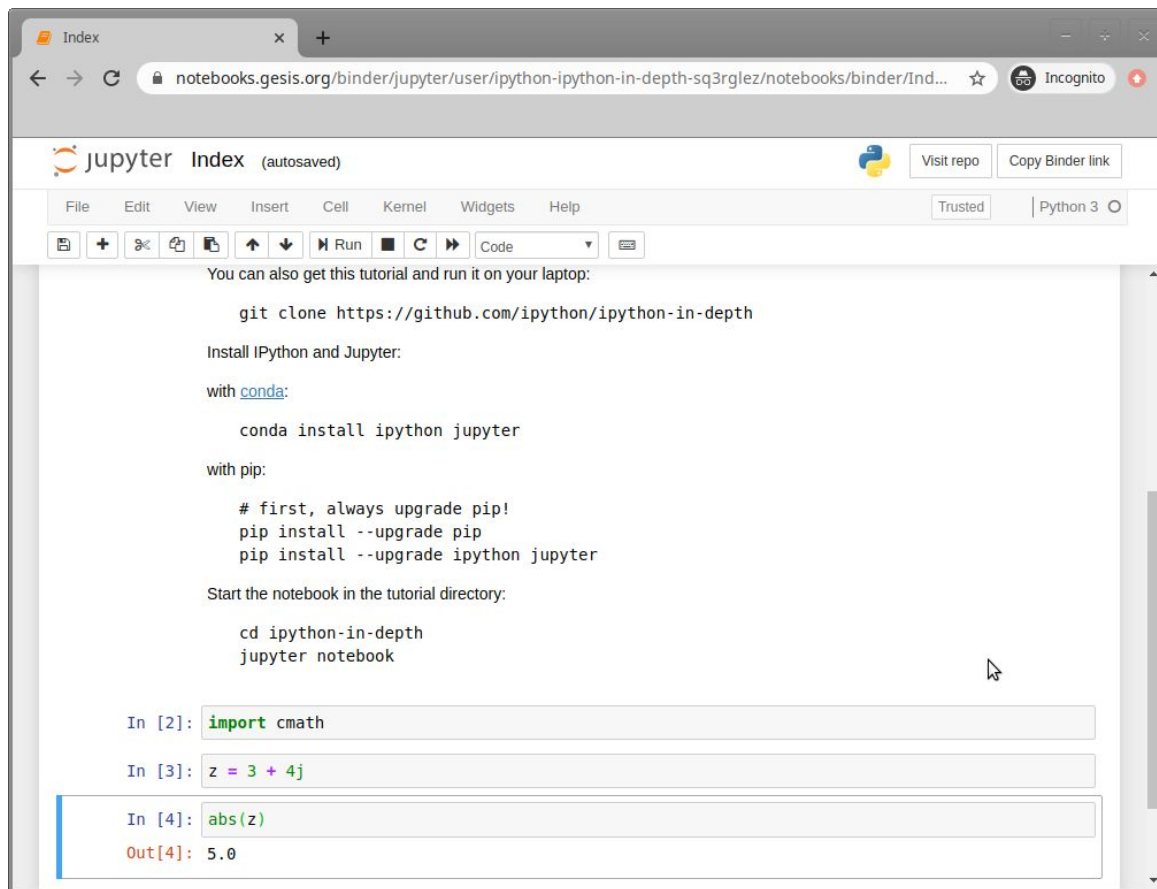
<https://python.org>





# Python CLI as Calculator

<https://jupyter.org/>



The screenshot shows a Jupyter Notebook interface in a web browser. The browser address bar shows the URL: `notebooks.gesis.org/binder/jupyter/user/ipython-ipython-in-depth-sq3rglez/notebooks/binder/Ind...`. The notebook title is "Index (autosaved)". The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, running, and code execution. The main content area contains a tutorial on installing IPython and Jupyter, followed by a code cell demonstrating complex number arithmetic.

You can also get this tutorial and run it on your laptop:

```
git clone https://github.com/ipython/ipython-in-depth
```

Install IPython and Jupyter:

with [conda](#):

```
conda install ipython jupyter
```

with pip:

```
# first, always upgrade pip!  
pip install --upgrade pip  
pip install --upgrade ipython jupyter
```

Start the notebook in the tutorial directory:

```
cd ipython-in-depth  
jupyter notebook
```

In [2]: `import cmath`

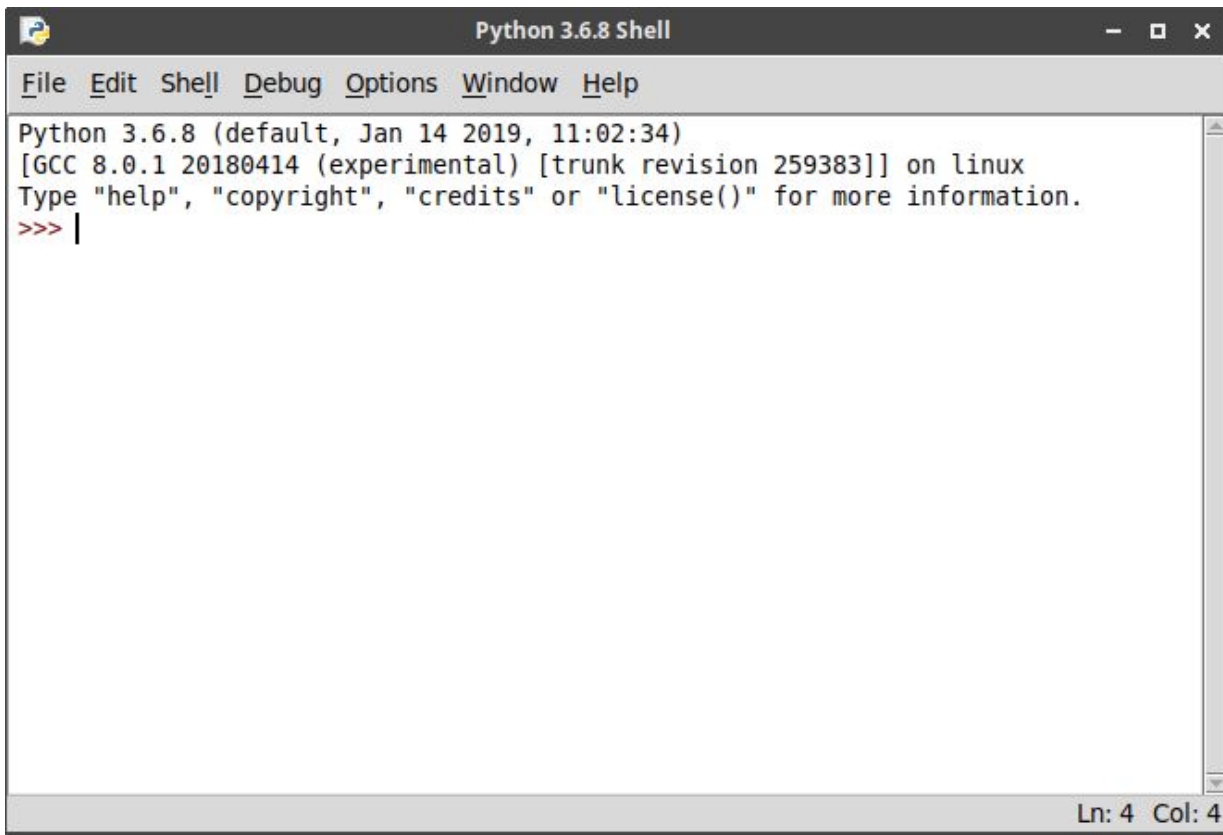
In [3]: `z = 3 + 4j`

In [4]: `abs(z)`

Out[4]: `5.0`

## Python CLI as Calculator

Python idle

A screenshot of a Python 3.6.8 Shell window. The window title is "Python 3.6.8 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area displays the following information: "Python 3.6.8 (default, Jan 14 2019, 11:02:34)", "[GCC 8.0.1 20180414 (experimental) [trunk revision 259383]] on linux", and "Type 'help', 'copyright', 'credits' or 'license()' for more information.". Below this, the prompt ">>> |" is shown, indicating the shell is ready for input. The status bar at the bottom right shows "Ln: 4 Col: 4".

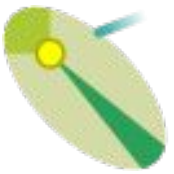
```
Python 3.6.8 (default, Jan 14 2019, 11:02:34)
[GCC 8.0.1 20180414 (experimental) [trunk revision 259383]] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> |
```

# Python Libraries for Science and Engineering

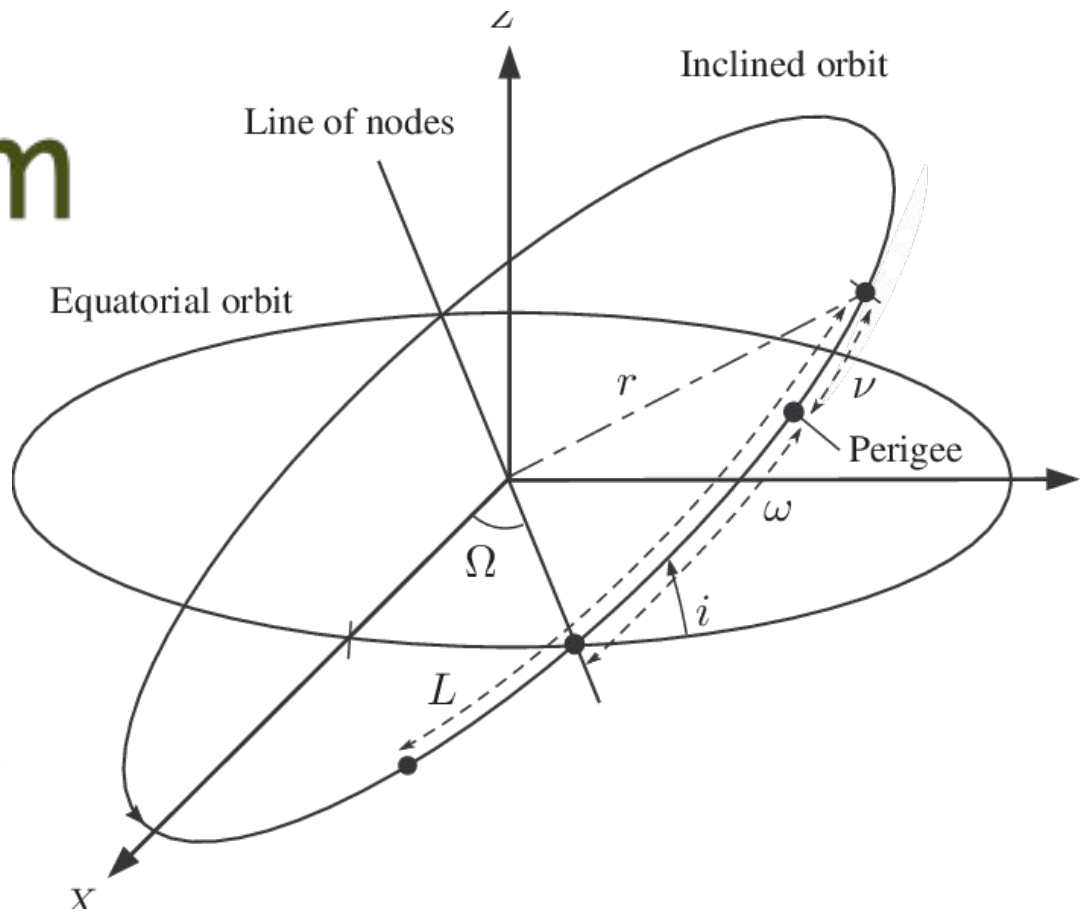
# **I R A F**

*Image Reduction and Analysis Facility*





# PyEphem





# astropy

A Community Python Library for Astronomy

```
import numpy as np
import matplotlib.pyplot as plt
from astropy.io import fits
from astropy.utils.data import download_file
# Take a shot which was taken in Mykolaiv Astrom=nomical Observatory
# from open DataBase UKR Virtual Observatory
fit_url = 'http://nao.db.ukr-vo.org/MBT_F/2013/0213/R2015253K.fits.gz'
image_file = download_file(fit_url, cache=True)

hdu_list = fits.open(image_file)
print(hdu_list.info())

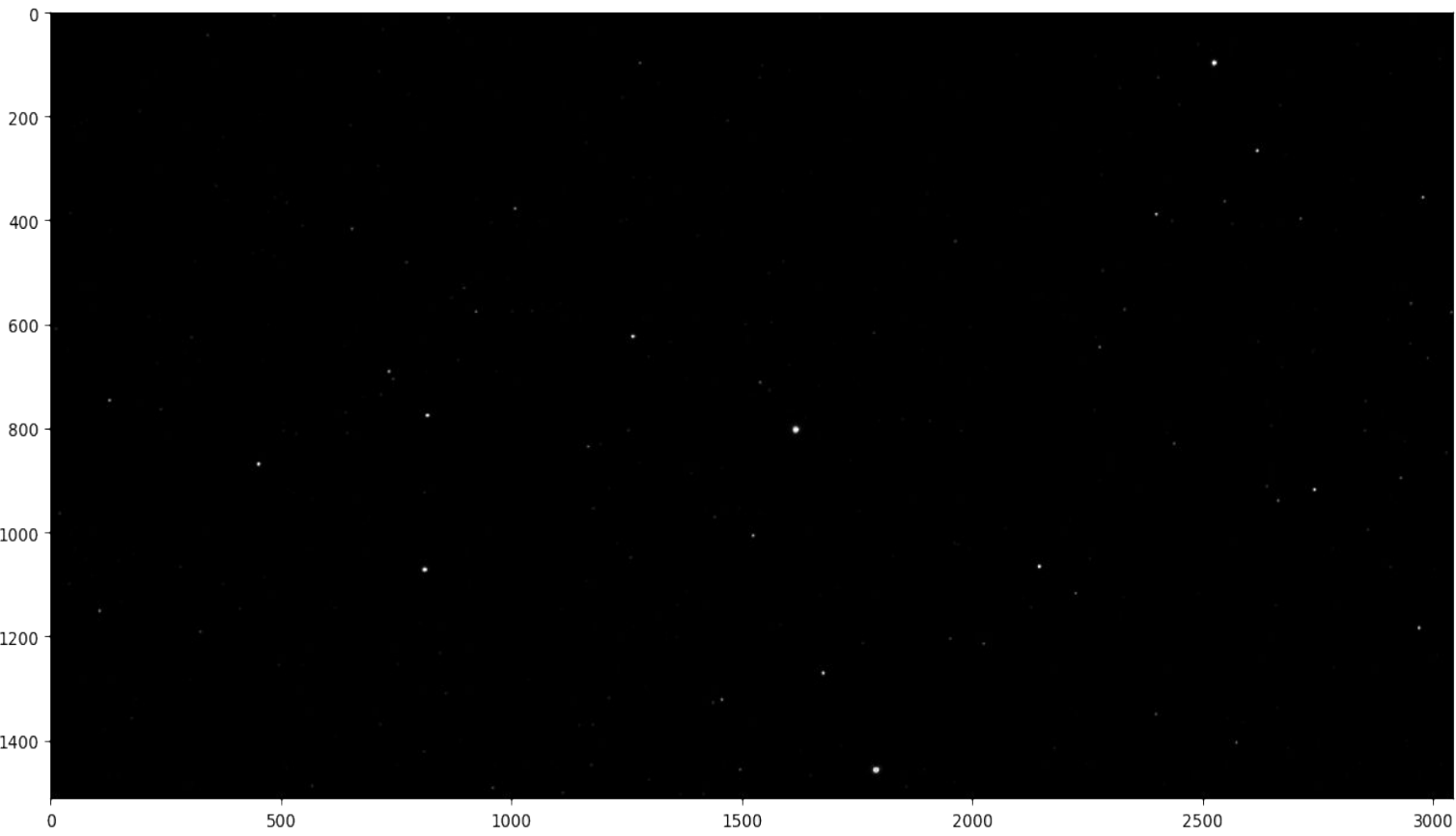
image_data = hdu_list[0].data

print(type(image_data))
print(image_data.shape)

hdu_list.close()

image_data = fits.getdata(image_file)
print(type(image_data))
print(image_data.shape)

plt.imshow(image_data, cmap='gray')
plt.colorbar()
plt.show()
```







# ObsPy

A Python Framework for Seismology

```
from obspy import read

st = read(

'http://examples.obspy.org/RJOB_061005_072159.ehz.new'

)

print(st)

tr = st[0] # assign first and only trace to new
variable
print(tr)
print(tr.stats)
tr.stats.station
tr.stats.gse2.datatype

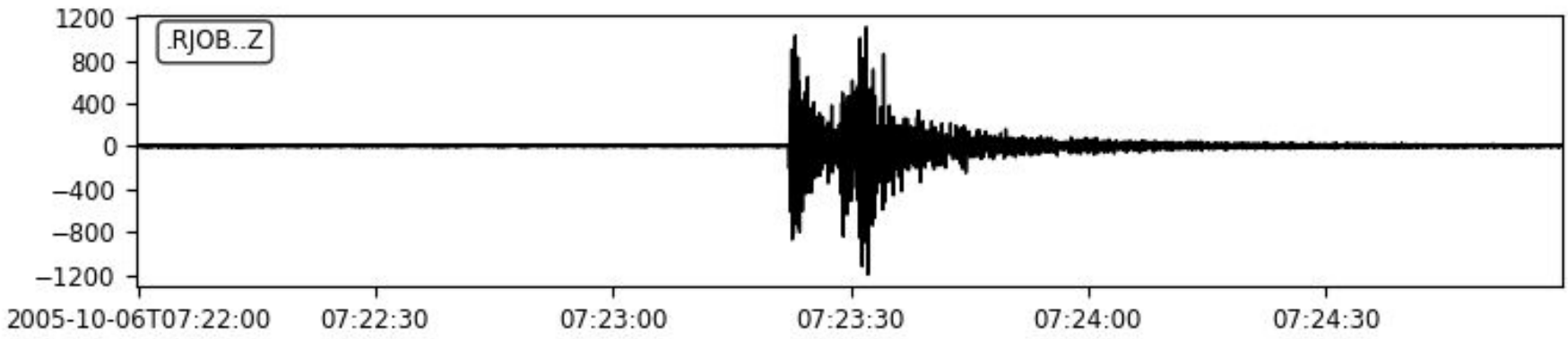
tr.data
tr.data[0:3]
```

```
1 Trace(s) in Stream:
.RJOB..Z | 2005-10-06T07:21:59.850000Z -
2005-10-06T07:24:59.845000Z | 200.0 Hz, 36000 samples
.RJOB..Z | 2005-10-06T07:21:59.850000Z -
2005-10-06T07:24:59.845000Z | 200.0 Hz, 36000 samples
  network:
    station: RJOB
    location:
    channel: Z
  starttime: 2005-10-06T07:21:59.850000Z
  endtime: 2005-10-06T07:24:59.845000Z
  sampling_rate: 200.0
    delta: 0.005
    npts: 36000
    calib: 0.0949
  _format: GSE2
    gse2: AttribDict({'auxid': 'RJOB',
'datatype': 'CM6', 'calper': 1.0, 'instype': '', 'hang':
-1.0, 'vang': -1.0, 'lat': -999.0, 'lon': -999.0,
'coordsys': '', 'elev': -0.999, 'edepth': -0.999})
```

```
st.plot()
```

Confidential

2005-10-06T07:21:59.85 - 2005-10-06T07:24:59.845



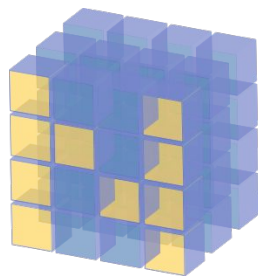


**SciPy**

All needed Numerical Methods for Physics research



**SciPy**



**NumPy**



Physical Consts and  
extended algorithms for  
Physics research

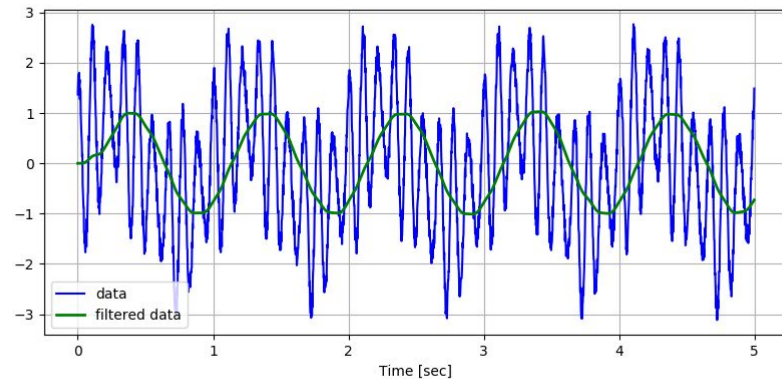
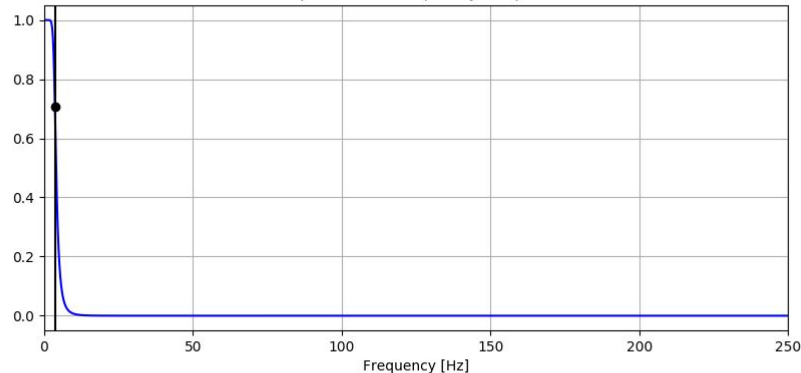
# Butterworth filter

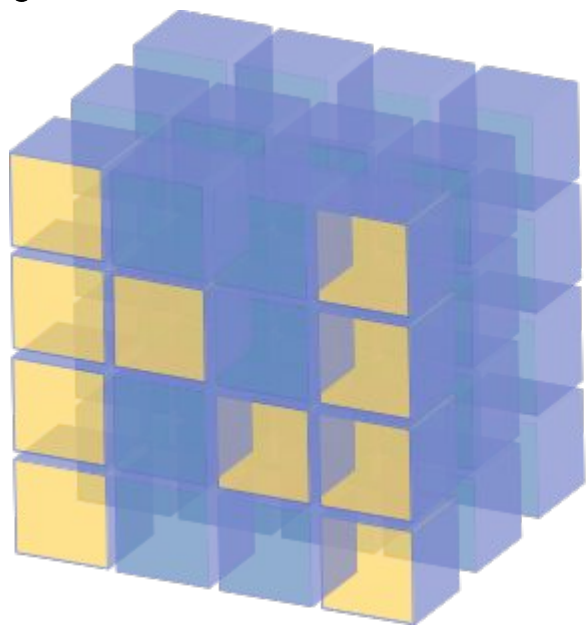
```
import numpy as np
from scipy.signal import butter, lfilter, freqz, firwin
import matplotlib.pyplot as plt
```

```
def butter_lowpass(cutoff, fs, order=5):
    nyq = 0.5 * fs
    normal_cutoff = cutoff / nyq
    b, a = butter(order, normal_cutoff, btype='low',
analog=False)
    return b, a

def butter_lowpass_filter(data, cutoff, fs, order=5):
    b, a = butter_lowpass(cutoff, fs, order=order)
    y = lfilter(b, a, data)
    return y
```

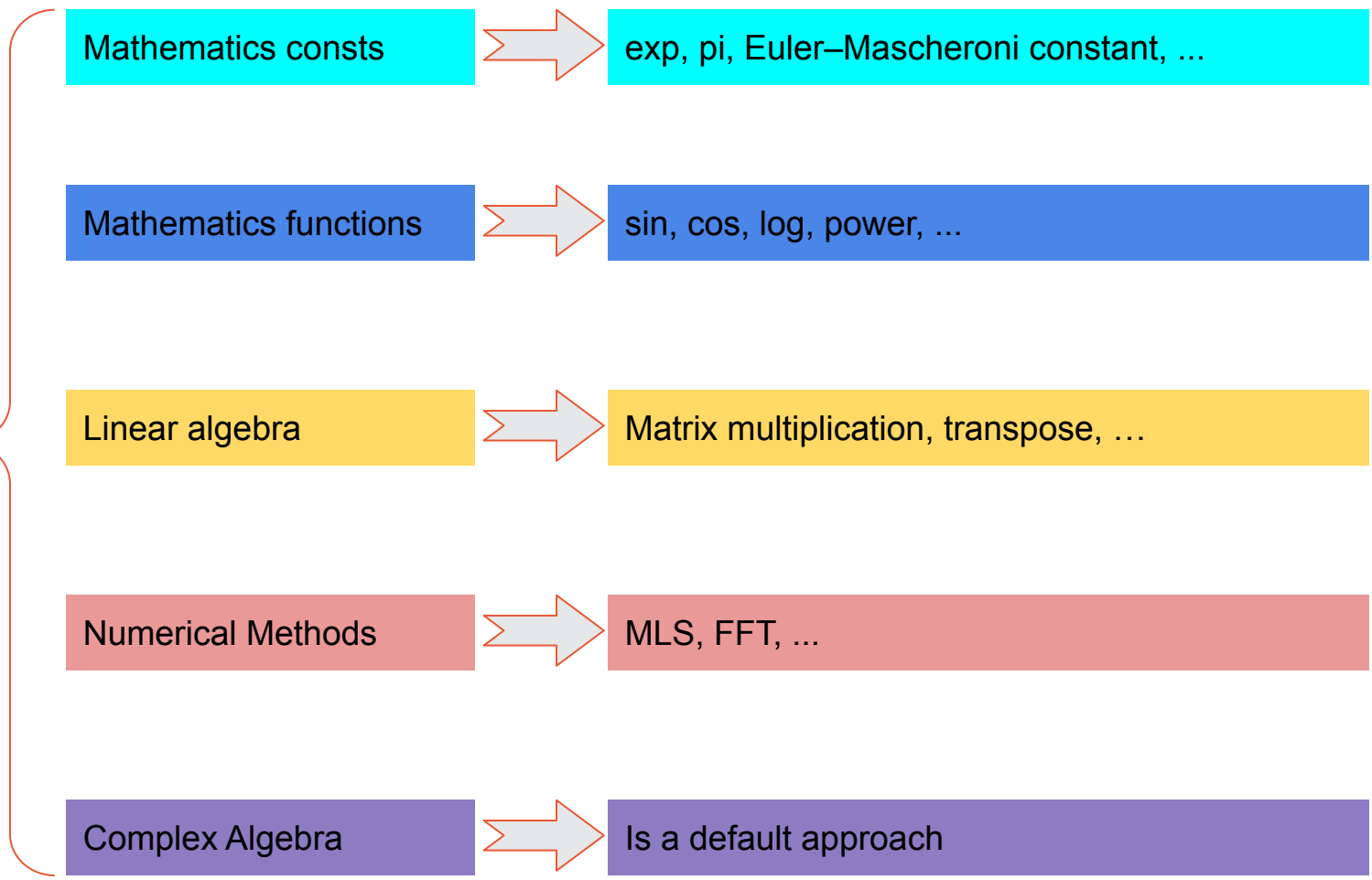
Lowpass Filter Frequency Response





# NumPy

All needed Numerical Methods





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

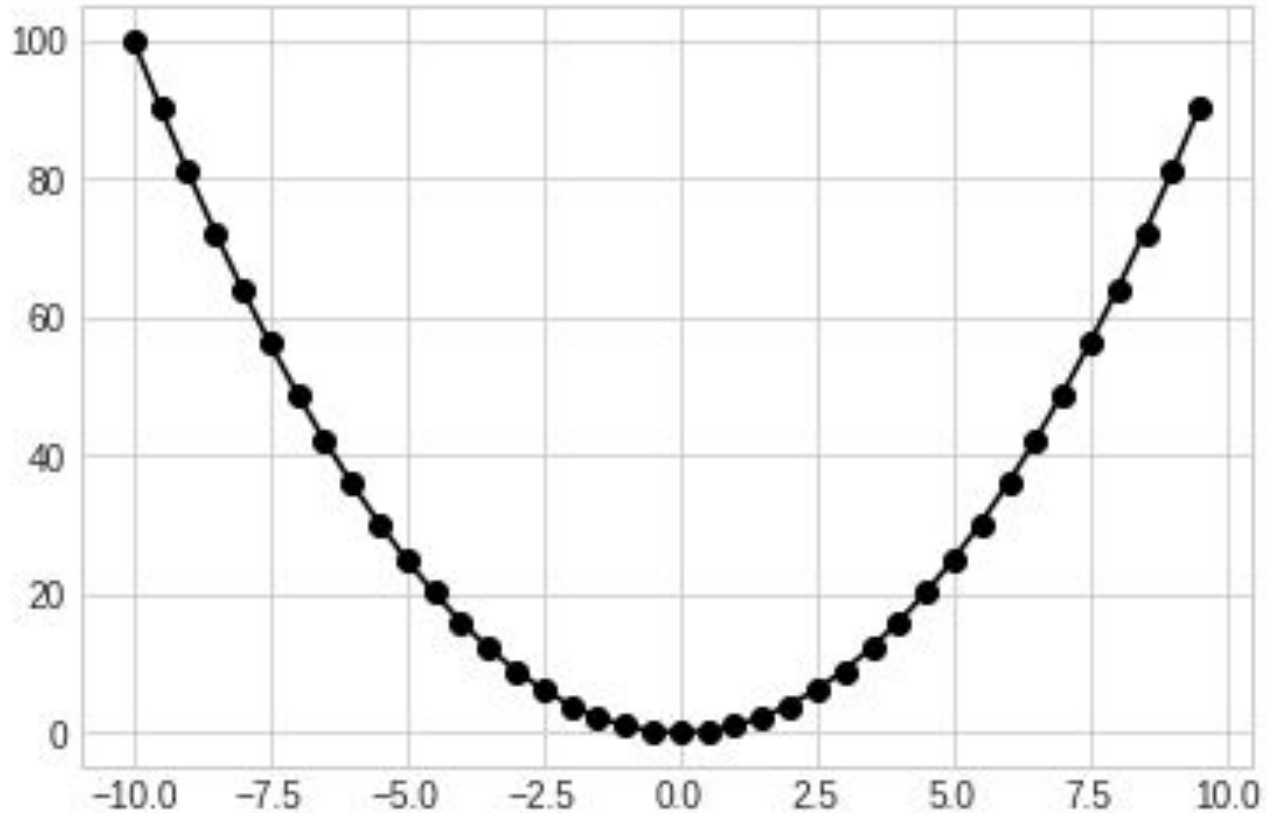
x = np.arange(-10, 10, .5)
y = np.power(x, 2)
plt.plot(x, y, '-ok')
plt.show()

noise = np.random.uniform(low=-10, high=10,
size=len(x))
plt.plot(x, noise, '-ok')
plt.show()

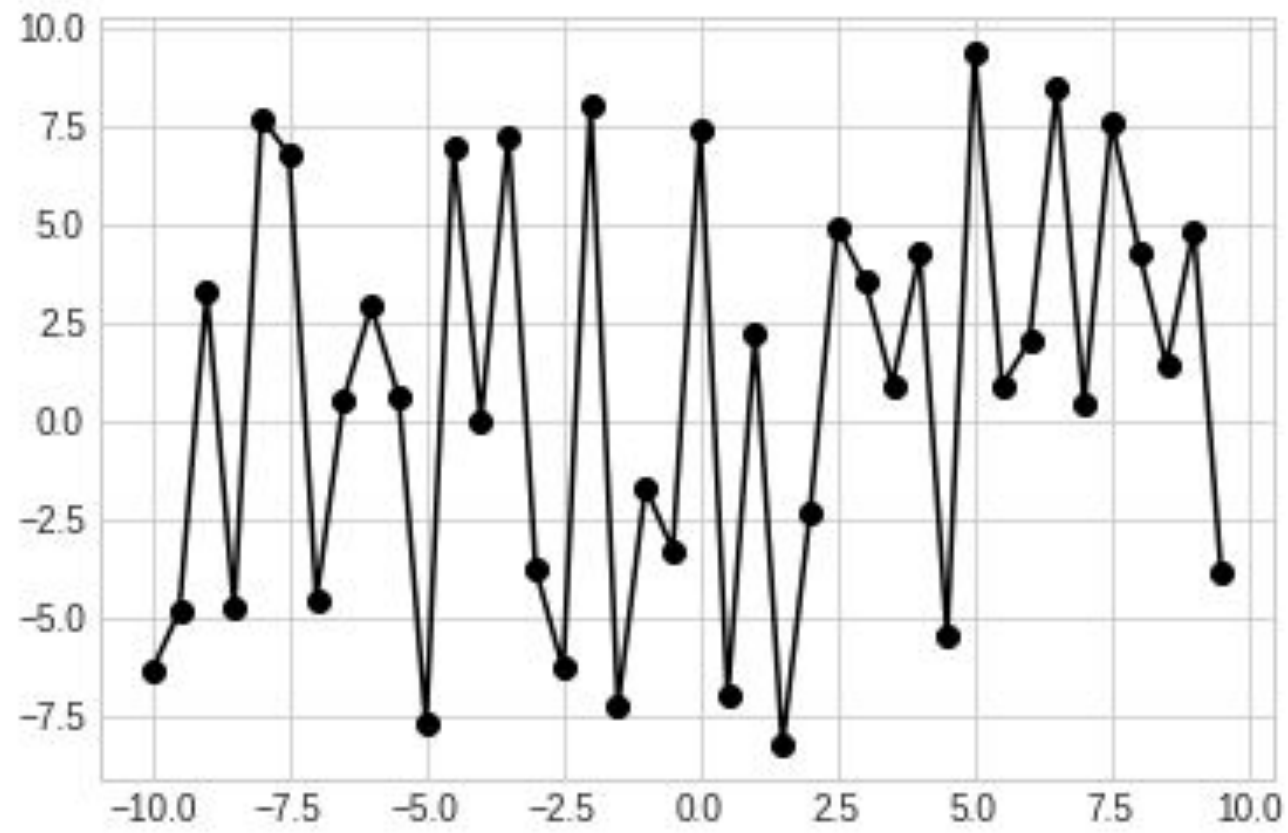
y_real = y + noise
plt.plot(x, y_real, '-ok')
plt.show()

A = np.vstack([x**2, x, np.ones_like(x)]).T
results = np.linalg.lstsq(A, y_real,
rcond=None)
print(results[0])
```

```
x = np.arange(-10, 10, .5)  
y = np.power(x, 2)  
plt.plot(x, y, '-ok')  
plt.show()
```



```
noise =  
np.random.uniform( low=-10,  
high=10, size=len(x))  
plt.plot(x, noise, '-ok')  
plt.show()
```

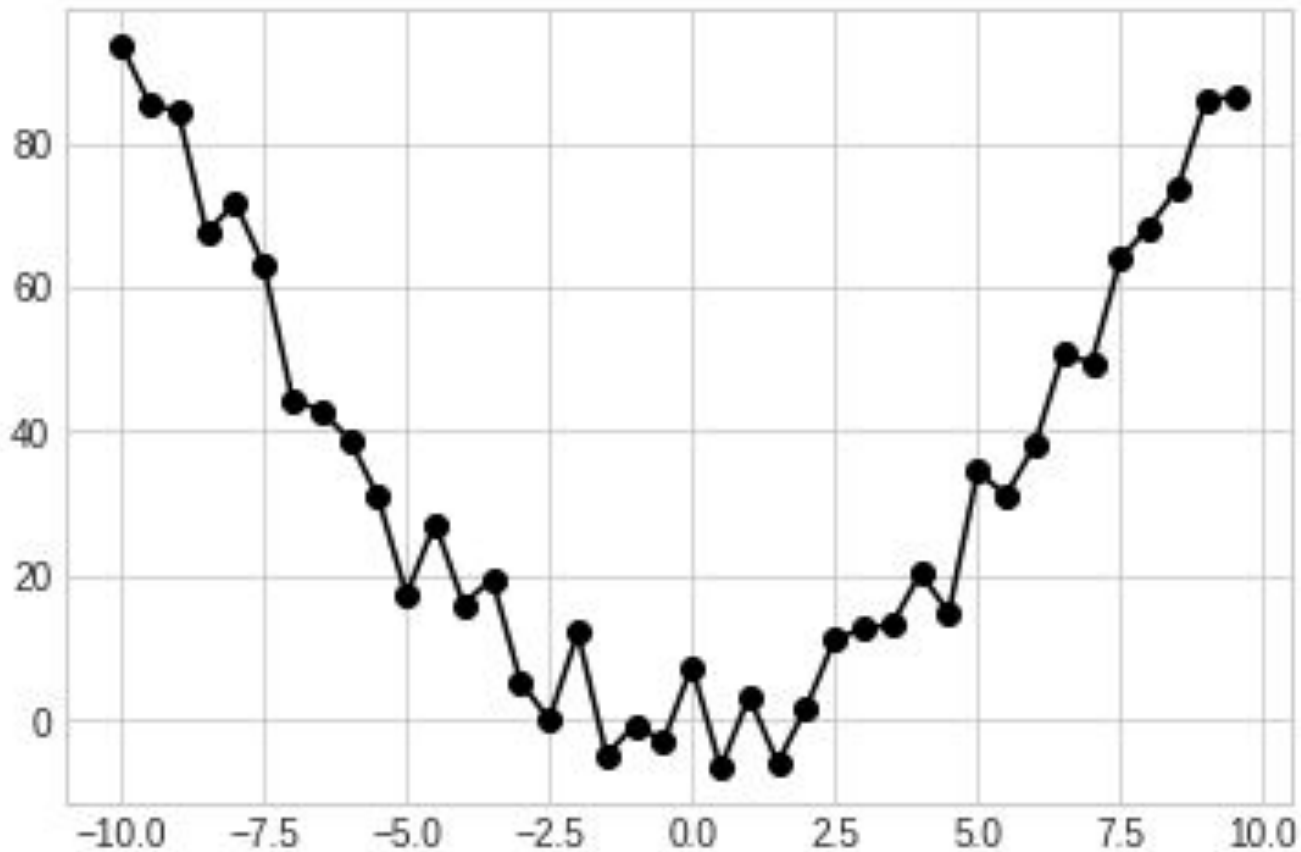


```
y_real = y + noise  
plt.plot(x, y_real, '-ok')  
plt.show()
```

```
A = np.vstack([x**2, x,  
np.ones_like(x)]).T  
results = np.linalg.lstsq(A,  
y_real, rcond=None)  
print(results[0])
```

Result:

```
(array([  
1.0097034, # a * x^2  
0.20207776, # b * x  
0.47732674 # c  
)
```





# SymPy

```
from sympy import *
init_printing()
```

```
x, y = symbols('x y')
f = (x**2 + y**2)
pprint(f)
```

```
F = Integral(f, x, y)
pprint(F)
pprint(F.doit())
```

```
pprint(diff(f, x))
```

```
lambda_ = Symbol('lambda')
res = sqrt(6 * Sum(1/lambda_**2,
(lambda_, 1, oo)))
pprint(res)
pprint(res.doit())
```

$$x^2 + y^2$$

function "f"

$$\iint (x^2 + y^2) dx dy$$

integral of "f"

$$\frac{x^3 \cdot y}{3} + \frac{x \cdot y^3}{3}$$

integration result of "f"

$$2 \cdot x$$

differentiation of "f" by "x"

$$\sqrt{6} \cdot \sum_{\lambda=1}^{\infty} \frac{1}{\lambda^2}$$

infinite summ

$$\pi$$

result of summ

# Questions?