Software Tools for UNIX/Linux Systems

Part 10: Visualization- Using Matplotlib

C.Hasse

TECHNISCHE UNIVERSITÄT DARMSTADT

STFS

From: matplotlib.org

Agenda

1. **Introduction**

2. **Basics**

3. **Example**

4. **What should you have learned**

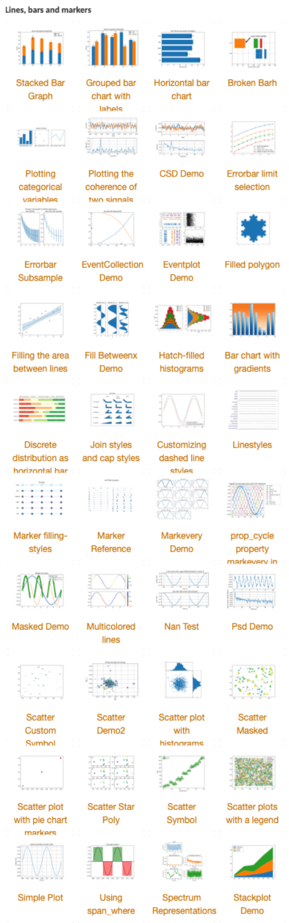
What is matplotlib?

Matplotlib

- Open source python library
- Very popular in scientific community
- A lot of possibilities
- High level of customization possible
- Mastermind behind:
  - John D. Hunter
- Started 2002
- License: Python Software Foundation (PSF) license
- Issue: Documentation sometimes redundant –> confusing

matplotlib

From: matplotlib.org



Why do we use matplotlib?

Competitors:

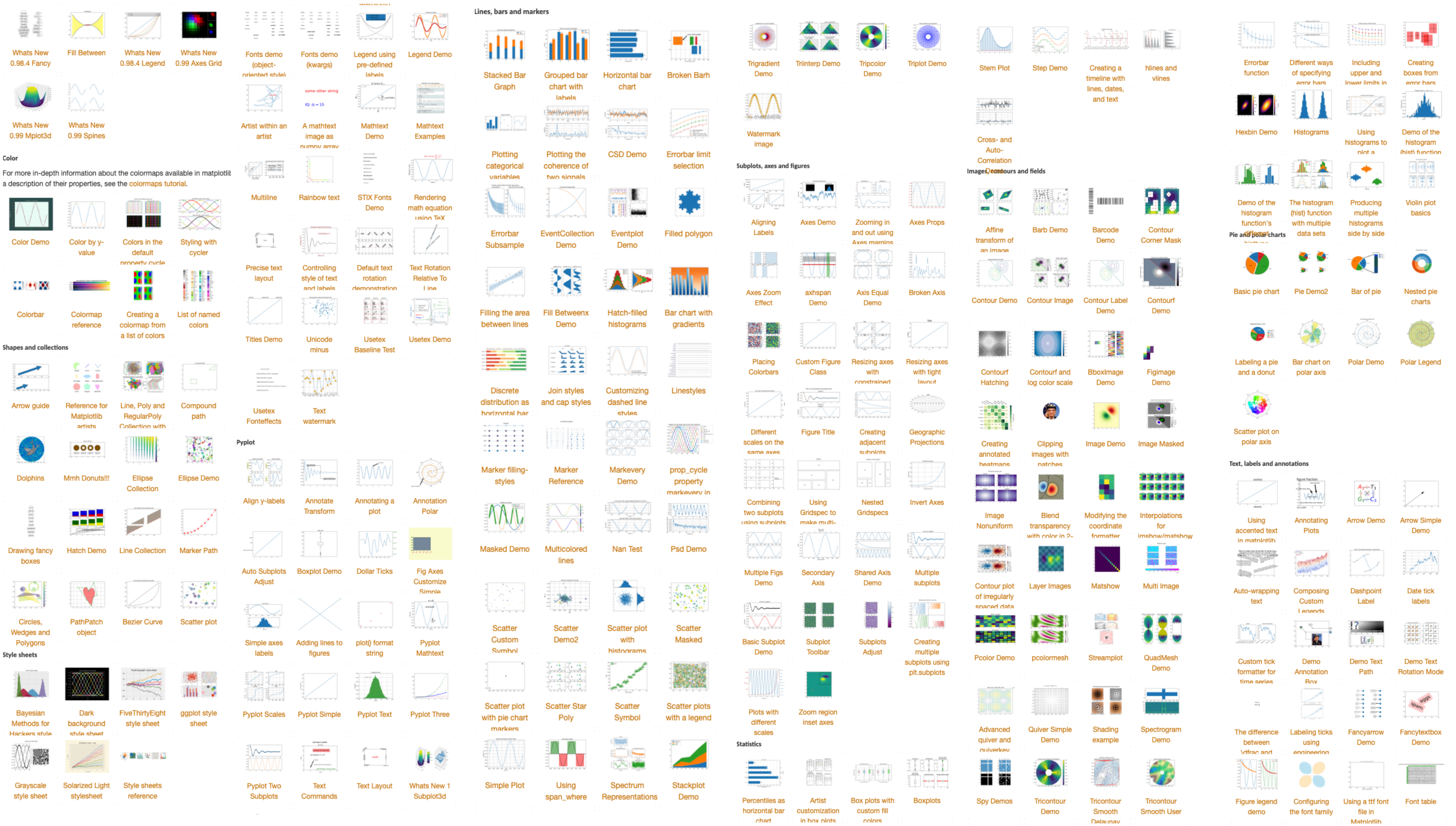
- Matlab
- gnuplot (very common)
- tecplot
- graphpad-prism
- mathematica
- labplot

Advantages:

- high level of customization
- open source
  - can be shared
  - can be installed everywhere
  - platform independent
- community support
- close to matlabsyntax

- Disadvantages:
- documentation is an issue
  - (projects derived out of this for automatic documentation)
- animation of plots is rather complicated
- many ways to get to the solution

What is possible?



… and many more….!!!!

From: matplotlib.org

How to get started

Installation:

- Python/Conda installation
- (Create environment)
- Execute:
  - pip install matplotlib
- It is recommended to use pandas for data acquisition:
  - pip install pandas
- Get an IDE (e.g. spyder, PyCharm…)
- Use jupyter-notebook
  - pip install jupyter

From: https://pandas.pydata.org/

From: https://docs.spyder-ide.org/

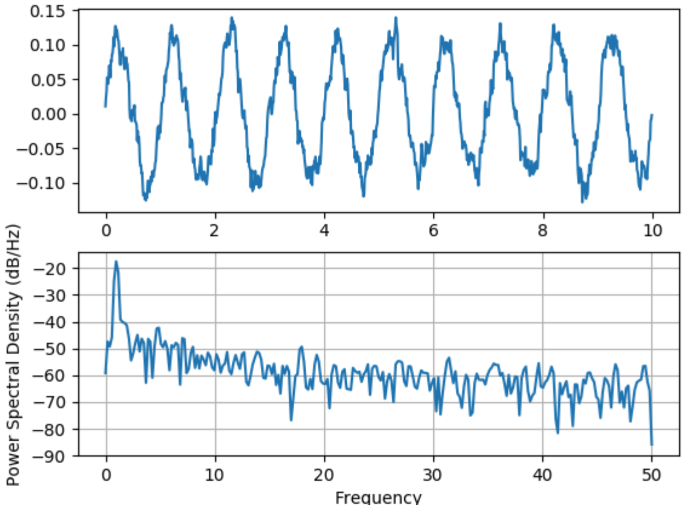
From: https://de.wikipedia.org/wiki/PyCharm

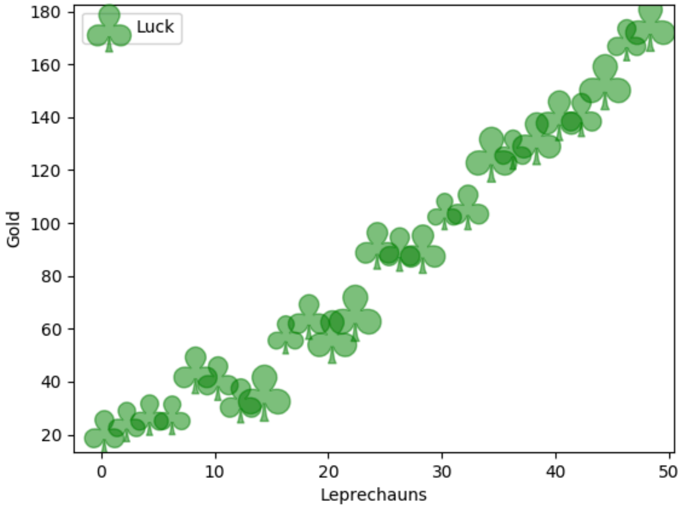
From:https://jupyter.org/

Basics

Matplotlib.pyplot as plt:

- Pyplot is the module of matplotlib that we will focus on most
- Important functions:
  - **plt.plot**(*args, scalex=True, scaley=True, data=None, **kwargs)
  - **plt.scatter**(x, y, s=None, c=None, marker=None, cmap=None, norm=None, vmin=None, vmax=None, alpha=None, linewidths=None, verts=None, edgecolors=None, *, plotnonfinite=False, data=None, **kwargs)





From: matplotlib.org

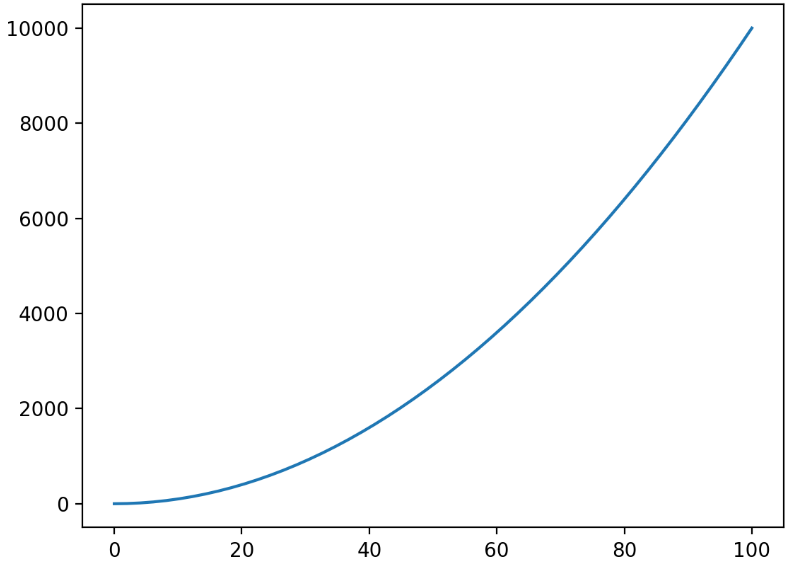
Basics: Simple example

Basics:

- Import libraries
- Use simple plot() function
- Show the plot

```
#import important libraries
import matplotlib.pyplot as plt
import numpy as np

#create data
x = np.linspace(0, 2, 100)
#actual plotting
plt.plot(x,x**2)
plt.show()
```



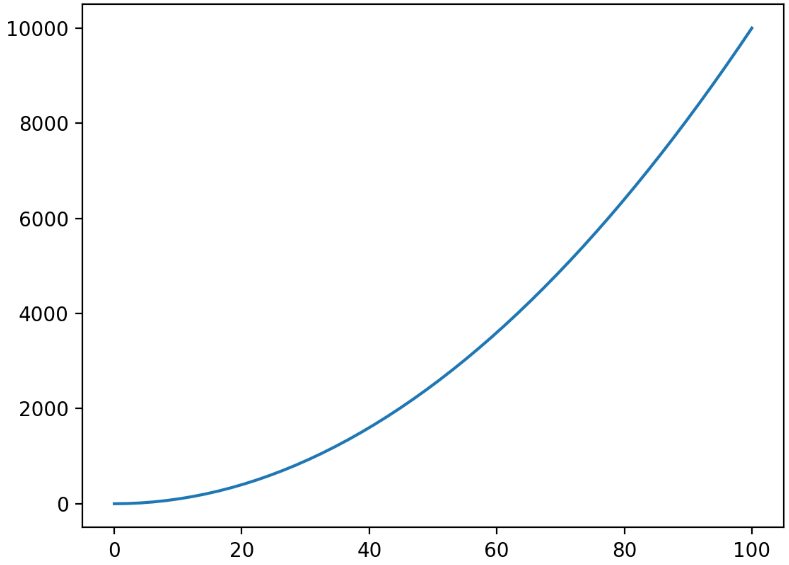
Basics: Simple example

Basics:

- ▶ Import libraries
- ▶ Use subplots()
- ▶ Show the plot

```
#import important libraries
import matplotlib.pyplot as plt
import numpy as np

#create data
x = np.linspace(0, 2, 100)
# create figure
fig, ax = plt.subplots()
#actual plotting
ax.plot(x,x**2)
plt.show()
```



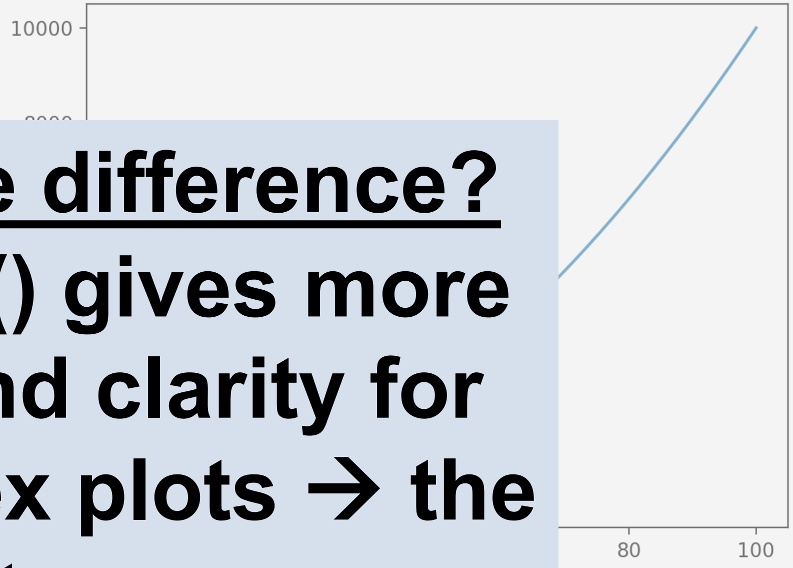
Basics: Simple example

Basics:

- Import libraries
- Use subplots()
- Show the plot

```
#import importan
import matplotlib.
import numpy as

#create data
x = np.linspace(0
# create figure
fig, ax = plt.subplots()
#actual plotting
ax.plot(x,x**2)
plt.show()
```



Where is the difference?

plt.subplots() gives more flexibility and clarity for more complex plots → the way to go

Real life example

Real life example:

- Download data from moodle
- Install matplotlib, pandas, jupyter-notebook
- Open notebook:
  - Go to folder containing data and notbook
  - Execute:
    - “jupyter-notebook linuxTutoriumNotebook”

From: https://media.giphy.com/media/3ohjUVvzSNsKmzUum4/200w_d.gif

From: https://media.giphy.com/media/yR4xZagT71AAM/giphy.mp4

Literature

- https://realpython.com/python-matplotlib-guide/

- https://matplotlib.org/Matplotlib.pdf