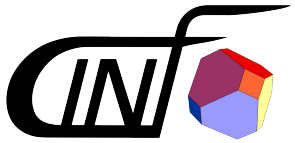
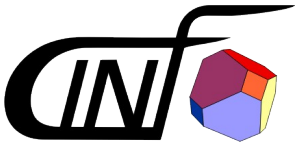


Center for Individual Nanoparticle Functionality

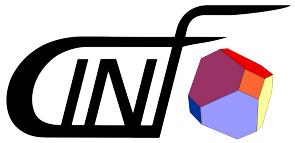
- The scientific Python ecosystem
- matplotlib
- scipy
- Examples
- ipython notebook

Python part 2 – matplotlib & scipy

The program



- Move through each of the topics in the outline
- For each topic there will be:
  - A presentation by me
  - A “type along” session, where we type the new things together
  - A few exerciser (the exercises are meant mainly for typing repetition, so most are simple)

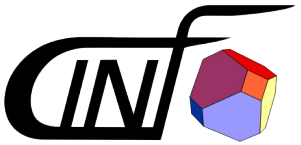


Center for Individual Nanoparticle Functionality

- The scientific Python ecosystem
- matplotlib
- scipy
- Examples
- ipython notebook

Python part 2 – matplotlib & scipy

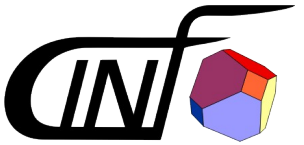
The ecosystem for scientific computing in Python



- **numpy** – arrays and math functions
  - polynomials, sin, exp, etc.
- **scipy** – algorithms for arrays
  - fitting, smoothing, searching, FFT, etc..
- **matplotlib** – plotting
- **scitkits** – SciPy toolkits
  - scikit-learn: Machine learning
  - scikit-parse: Sparse matrices
- ...

The core

Many more extra packages



- **sympy** symbolic manipulation
- **lmfit** for better controlled least squares fit
- **pandas** for data munging
- **seaborne** for quick correlation plotting
- more on https://wiki.python.org/moin/NumericAndScientific

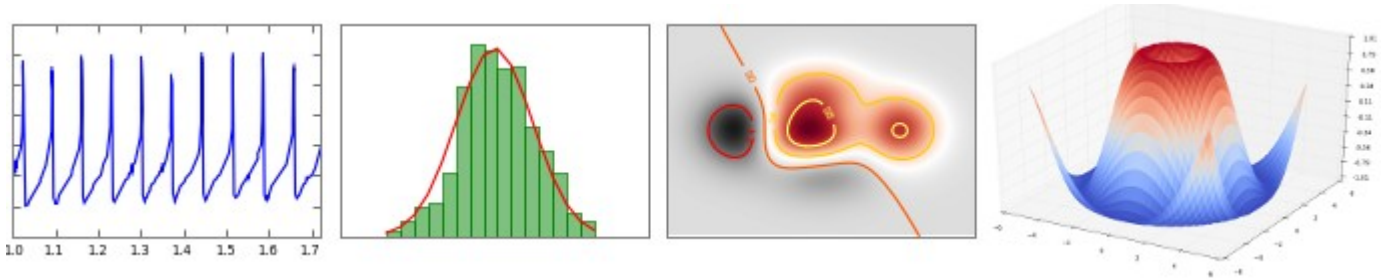
Center for Individual Nanoparticle Functionality

- The scientific Python ecosystem
- **matplotlib**
- scipy
- Examples
- ipython notebook

Python part 2 – matplotlib & scipy

matplotlib

- “matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. matplotlib can be used in python scripts, the python and ipython shell (ala MATLAB®* or Mathematica®†), web application servers, and six graphical user interface toolkits.”

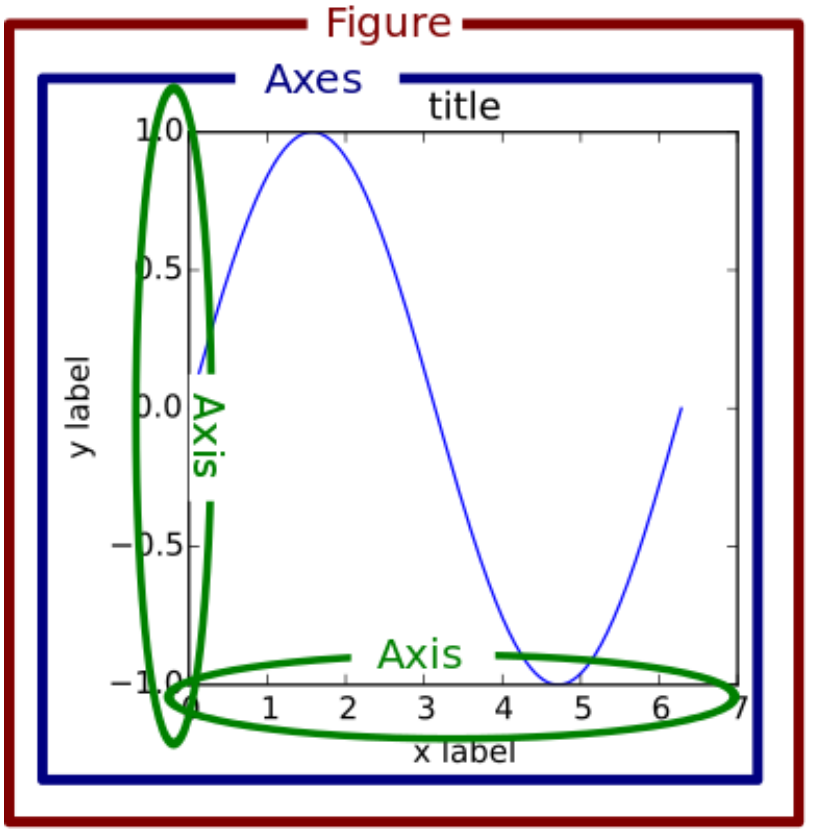


matplotlib

- **matplotlib** is incredibly features rich
- Many different types of plots
- Publication quality figures
- Can be created scripted
  - Which is good for reproducability
  - And bad for manual fidling with e.g. arrows
- http://matplotlib.org/

Anatomy of a figure

- **Figure** is the whole thing
  - Can contain subplots
- **Axes** if the single plot object
- **Axis** used to set limits etc.
- Inside are **Line2D**



http://matplotlib.org/faq/usage_faq.html

matplotlib and matlab

- **matplotlib** was created with inspiration from matlab
- Attempt to provide drop-in-replacement
- The has a few negative consequences from a python point of view:
  - matplotlib count from 1 …
  - Extrensive use of get set instead of properties
  - Two entry points for usage (the two faces of matplotlib)

pyplot

- pyplot is a module witin matplotlib
- Resemble Matlab plotting as closely as possible
- pyplot is a state machine
  - It holds the figure objects for you
  - Has a notion about “the current figure”
  - All action performed with pyplot works on the current figure

pyplot

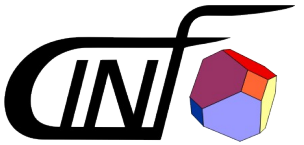
```
from matplotlib.pyplot import pyplot as plt

# plt contains the figure and axes
plt.plot([1, 2, 3], [4, 3, 5], label='First line')
# Creates the legend
plt.legend()

# Titles and labels
plt.title('The title')
plt.xlabel('The xlabel')
plt.ylabel('The ylabel')

plt.show()
```

pyplot only as start (object oriented)



```
from matplotlib import pyplot as plt

figure = plt.figure()
axes = figure.add_subplot(111)

axes.plot([1, 2, 3], [4, 3, 5], label='First line')
axes.legend()
axes.set_title('The title')
axes.set_xlabel('The xlabel')
axes.set_ylabel('The ylabel')

plt.show()
```

Next to one another ...

pyplot

```
from matplotlib.pyplot \
    import pyplot as plt




# plt contains figure and axes
plt.plot([1, 2, 3], [4, 3, 5],
         label='First line')


plt.legend()

# Titles and labels
plt.title('The title')
plt.xlabel('The xlabel')
plt.ylabel('The ylabel')

plt.show()
```

object oriented

```
from matplotlib \
    import pyplot as plt

# Create figure and axis
figure = plt.figure()
axes = figure.add_subplot(111)


axes.plot([1, 2, 3], [4, 3, 5],
          label='First line')


axes.legend()

# Titles and labesl
axes.set_title('The title')
axes.set_xlabel('The xlabel')
axes.set_ylabel('The ylabel')

plt.show()
```

On the two faces

- Some more advanced functionality is only available in object orinted style
- We already know that everything is an object
- Pull them out and use them

- **Recommend to use the object oriented style**
- But will not outright say that pyplot style is a bad idea
- Examples online exist in both

Docs

- Pyplot reference:

  http://matplotlib.org/api/pyplot_summary.html

- Object oriented:

  Axes: http://matplotlib.org/api/axes_api.html

  Fugure: http://matplotlib.org/api/figure_api.html

Subplots

- Ability to create several axes
- Use three integers e.g. 121
  - num_rows, num_columns, num_figure
- Written as just on integer
- Or string
- plt.subplot(211) # pyplot
- figure.add_subplot(211) # object orient
- Both pyplot and figure has tight_layout() which is usefull

matplotlib – Type along

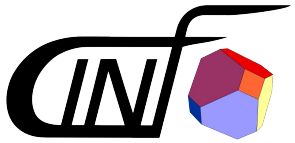
Type along session in the terminal:
matplotlib

Types - Exercises

- With both examples, experiment with:
  - modifying line style and color
  - line width

- Make a figure with two axes (sub plots) in it

  You will probably find **tight_layout()** useful



Center for Individual Nanoparticle Functionality

- The scientific Python ecosystem
- matplotlib
- scipy
- Examples
- ipython notebook

Python part 2 – matplotlib & scipy

scipy

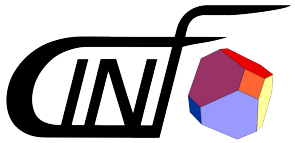
- “Fundamental library for scientific computing“
- Basically algorithms to work on arrays
  - Special functions
  - Integration
  - Optimization
  - Interpolation
  - Fourier transform
  - Linear algebra
  - Statistics
  - ...

scipy docs

- Get the version: scipy.__version__
- http://docs.scipy.org/doc/

scipy

- Is fairly extensive
- To find an algorithm
  - Start with google
  - Then go to the documentation to figure out what it does
- Ex: polyfit



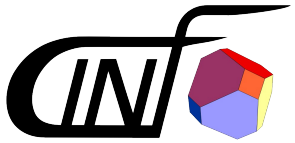
Center for Individual Nanoparticle Functionality

- The scientific Python ecosystem
- matplotlib
- scipy
- **Examples**
- ipython notebook

Python part 2 – matplotlib & scipy

Examples

- Databases and general fitting
  - Welcome back Robert
- Specific fitting and cutting
- Organzing code and making it look good

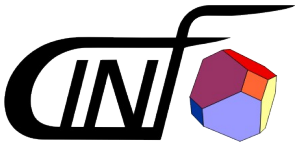


Center for Individual Nanoparticle Functionality

- The scientific Python ecosystem
- matplotlib
- scipy
- Examples
- ipython notebook

Python part 2 – matplotlib & scipy

The iPython notebook



- Is yet another way to run Python code
- In a browser (but with local files)
- Runs of a local server
- Provides a mathematica like “notebook” interface
- Very popular, lots of on-line documentation

iPython notebook – Type along

Type along session in the terminal:
iPython notebook

Center for Individual Nanoparticle Functionality

- **Summary**

Python part 2 – matplotlib & scipy

Summary

- Matplotlib is powerful
  - Publication quality figures
  - Fast plotting on-the-fly
  - But has to faces
- Scipy has a lot of extra algorithms
  - Go look for them
- The iPython notebook is a fun new way to use Python
  - Have a look at it

THE END