

# MAS212 Scientific Computing and Simulation

Dr. Sam Dolan

School of Mathematics and Statistics,  
University of Sheffield

Autumn 2017

<http://sam-dolan.staff.shef.ac.uk/mas212/>

G18 Hicks Building  
s.dolan@sheffield.ac.uk

# MAS212 Scientific Computing and Simulation



**Example:** The Barnsley Fern. Generated by repeated iteration of affine transformations of the form

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix}.$$

## Course Summary (2017)

- 10 credit module in first semester.
- 10 lectures + 10 lab classes + 10 office hours
- **Assessment:** 3 assignments (90%) + 2 class tests (10%)
- Course website:  
<http://sam-dolan.staff.shef.ac.uk/mas212>
- Pre-requisite: MAS115.
- *Not a 'soft option'.*

# Module description

## Aims

- To further develop the students' programming skill in the context of scientific computing;
- To further develop the students' independent investigation skills;
- To introduce the knowledge of scientific computing;
- To further develop the skills of data analysis.

## Outcomes

- To be able to use Python to investigate mathematical problems numerically.
- To learn basic numerical methods for solving ordinary differential equations and linear algebraic equations.
- To be able to implement basic numerical methods using Python.
- To be able to analyse the basic properties of the methods.

# Syllabus (from 2016)

- **Week 1:** The Python language. Revision: variables; data types; arithmetic; list construction, comprehension, indexing, slicing & manipulation; for & while loops; control flow (if-elif-else; strings; string formatting).  
Introduction to Jupyter Notebook: Tab completion, getting help and %magic commands (e.g. %timeit).
- **Week 2:** Functions. Modules & scripts. Built-in modules (math, cmath, random, decimal, datetime, io, os). Simple file I/O and string processing. Debugging and testing. Workflow.
- **Week 3:** Introduction to numpy. Arrays (initialization, slicing). Basic linear algebra. Efficiency.  
Introduction to matplotlib. Simple plotting.  
Examples: (1) Estimating  $\pi$  by Monte Carlo integration; (2) the logistic map.
- **Week 4:** Introduction to scipy. Solving differential equations with odeint. Initial conditions. Time-domain plots. Phase plots. Critical points and limit cycles.  
Examples: (1) Logistic equation; (2) Damped harmonic oscillator; (3) van der Pol oscillator; (4) Predator-prey equations.

# Syllabus (from 2016)

- **Week 5:** Animation with `matplotlib.animation.FuncAnimation`. Saving an animation. Examples: (1) The logistic map (again); (2) Driven damped oscillator and resonance.
- **Week 6:** Elementary numerical methods: Runge-Kutta and Adams-Bashforth methods. Implementation for initial value problems.
- **Week 7:** Error, order and stability of numerical methods.
- **Week 8:** Fitting models to data. `scipy.optimize.curve_fit`. Least-squares method and linear algebra.
- **Week 9:** Linear algebra. Gaussian elimination; iterative methods; convergence; condition number.
- **Week 10:** Plotting in 3D and visualisation. Data analysis with `pandas`. Other languages.

# Books

- There are many books on Python, and on scientific computing.
- I recommend:  
    *“Learning Scientific Programming with Python”*,  
    Christian Hill (Cambridge University Press, 2015).  
    ISBN 978-1-107-42822-5.
- Copies available in Information Commons

# Assessment guide: Assignments

- Three assignments (90%):

		Set	Due(*)
(1)	Assignment #1	Week 1	Week 4
(2)	Assignment #2	Week 4	Week 8
(3)	Assignment #3	Week 8	Week 12

- (\*) All dates provisional.
- In Assignment #1 you will submit one `.py` file
- In Assignments #2 you will submit code and write a report in LaTeX.
- In Assignment #3 you will submit code and a presentation.



## Assessment guide: Class tests

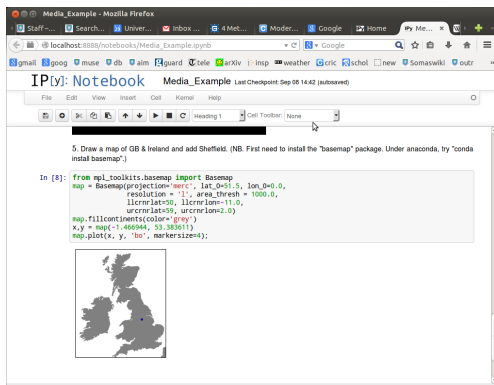
- Two 'open-book' class tests: each 5% of module mark.
- Held in lab class in Weeks 2 and 11.
- Intended as 'formative assessment' (to improve skills).
- Test will use **Jupyter Notebook**.
- Mock tests are available on the course website.
- If you cannot attend your Week 2 lab class, please email me.
- Submit 1st class test by **midnight Sun 8th Oct** via <http://somas-uploads.shef.ac.uk/mas212>

# IPython

- IPython is an interactive shell for the Python programming language (and others)
- <http://en.wikipedia.org/wiki/IPython>
- It has several advantages over the standard interpreter:
  - Tab completion
  - Object introspection and enhanced help
  - `%magic` commands

# Jupyter Notebook

<http://jupyter.org/notebook.html>



Media\_Example - Mozilla Firefox

localhost:8888/notebooks/Media\_Example.ipynb


IP[y]: Notebook Media\_Example Last Checkpoint: Sep 05 14:42 (autosaved)

File Edit View Insert Cell Kernel Help

Heading 1 Cell Toolbar: None

5. Draw a map of GB & Ireland and add Sheffield. (NB. First need to install the "basemap" package. Under anaconda, try "conda install basemap".)

```
In [8]: from mpl_toolkits.basemap import Basemap
map = Basemap(projection='merc', lat_0=51.5, lon_0=0.0,
              resolution='1', area_thresh=1000.0,
              llcrnrlat=50, llcrnrlon=-11.0,
              urcrnrlat=59, urcrnrlon=2.0)
map.fillcontinents(color='grey')
x,y = map(-1.466944, 53.383611)
map.plot(x,y,'bo', markersize=4);
```



Jupyter Notebook is ...

... a web-based interactive computational environment where you can combine code execution, text, mathematics, plots and rich media into a single document.

# Installing Jupyter Notebook

## On your computer:

The simplest way to get Jupyter Notebook is to install the Anaconda distribution of Python 3:

<http://continuum.io/downloads>

Choose Python 3.6 version.

This comes with the most popular libraries for scientific computing.

## On the managed desktop:

The Anaconda distribution should be pre-installed. From the Start Menu, look for the folder Anaconda3 (64-bit)

# Using Jupyter Notebook

- Enter Python code into a cell
- Press Shift-Enter to evaluate a cell
- Some example notebooks are shown on course website  
<http://sam-dolan.staff.shef.ac.uk/mas212>
- Notebooks can be converted to HTML or PDF.
- Notebooks may be shared on the web:  
<http://nbviewer.jupyter.org/>
- For an introduction to Jupyter Notebook see e.g.  
<http://opentechschoo1.github.io/python-data-intro/core/notebook.html>

# Using Jupyter Notebook: Magic functions

**Magic functions** start with `%`. Examples:

- `%matplotlib inline` : include the plots in the workbook (cf. `%pylab inline`).
- `%timeit my_func()` : test the efficiency of your function.
- `%load my_module.py` : read the contents of `my_module.py` into a cell
- `%run my_module.py` : run the module as a script
- `?reversed` : get help on the `reversed` function (e.g.).
- `!` : execute a shell command.

# Spyder

- Spyder is an Integrated Development Environment (IDE) for Python, ...
- ... the **S**cientific **PY**thon **D**evelopment **EnviR**onment.
- It includes
  - A code editor with syntax colouring
  - An IPython console
  - Introspection: tab completion; go-to-definition, etc.
  - Online help
  - Object inspector
  - Debugging features, such as breakpoints
- You are encouraged to use Spyder and/or Jupyter Notebook :
  - The class test will use Jupyter Notebook;
  - Spyder is useful for developing code for the first assignment.

## Checklist for Weeks 1–4

- ( ) Work through the ‘Python basics’ slides & **videos** on MOLE.
- ( ) Get started with Jupyter Notebook in Week 1 lab class
- ( ) Browse the course website & example notebooks.  
<http://sam-dolan.staff.shef.ac.uk/mas212>
- ( ) Try mock class test(s).
- ( ) Week 2 lab class: **Class Test**. Submit by **Sunday 8th Oct**
- ( ) Assignment #1 (Asymptotic series) due **Sunday 23rd Oct**.
  - NB. The **add/drop window** closes at the end of Week.