Computing for Geometry and Number Theory

Lecture 1

Dr John Armstrong

King's College London

August 22, 2020

Who?

Who?

- Dr John Armstrong

Who?

- Dr John Armstrong
- PhD "Almost-Kähler Geometry"

Who?

- Dr John Armstrong
- PhD “Almost-Kähler Geometry”
- Yolus risk management system (now ION trading)

Who?

- Dr John Armstrong
- PhD “Almost-Kähler Geometry”
- Yolus risk management system (now ION trading)
- Executive Director Goldman Sachs (operations technology)

Who?

- Dr John Armstrong
- PhD “Almost-Kähler Geometry”
- Yolus risk management system (now ION trading)
- Executive Director Goldman Sachs (operations technology)
- Senior Lecturer in Financial Mathematics, Probability and Statistics

What?

What?

- Weeks 1-4 + 6(ish): Mathematica. (Week 5 off)

What?

- Weeks 1-4 + 6(ish): Mathematica. (Week 5 off)
- Weeks 7(ish)-8: A little Python

What?

- Weeks 1-4 + 6(ish): Mathematica. (Week 5 off)
- Weeks 7(ish)-8: A little Python
- Weeks 9, 10: Project work

What?

- Weeks 1-4 + 6(ish): Mathematica. (Week 5 off)
- Weeks 7(ish)-8: A little Python
- Weeks 9, 10: Project work
- Weeks 11 (11 Dec): Project Presentation

Full Calendar

Week 1	2 Oct	Mathematica
Week 2	9 Oct	
Week 3	16 Oct	
Week 4	23 Oct	
Week 5	30 Oct	Reading Week
Week 6	6 Nov	Mathematica
Week 7	13 Nov	Python
Week 8	20 Nov	
Week 9	27 Nov	Projects
Week 10	4 Dec	
Week 11	11 Dec	Presentatons
f		

Why?

Why?

Why do you think?

Why?

Why do you think?

- You can eliminate drudgery (e.g. boring algebra, integration)
- You can get a computer to finish off a proof for you (e.g. four colour theorem).
- You can use a computer to explore and experiment with ideas
  - Search for patterns and generate hypotheses
  - Test hypotheses
- You can use a computer for visualisation
  - Visualize things you can't yet imagine
  - Help others visualize what you can already imagine
- You can use a computer to develop your understanding. E.g. geometry of perspective.

Why? continued

Why? continued

- It helped us get funding for the CDT!

Why? continued

- It helped us get funding for the CDT!
- You can use a computer to apply your ideas to real problems
    - Elliptic curve cryptography
    - Ricci flow can be used for face recognition
    - Homology theory can be used pattern matching
    - . . .

Why? continued

- It helped us get funding for the CDT!
- You can use a computer to apply your ideas to real problems
  - Elliptic curve cryptography
  - Ricci flow can be used for face recognition
  - Homology theory can be used pattern matching
  - ...
- Whatever you do in the future, computer skills will surely be important
  - A quant
  - A spy
  - A DNA topologist
  - ...

These are the reasons why I am teaching the course and not someone else.

Why? advanced research

- in symplectic geometry, Seidel's proof of homological mirror symmetry for the quartic surface used Singular and Python for some of the computations;
- in algebraic geometry, there is a finite list of deformation classes of Fano 4-folds and the Fanosearch project is hoping to classify them by enumerating their mirror Landau-Ginzburg superpotentials and grouping them according to mutation equivalence: a massive computational task;
- in additive number theory, Helfgott's recent proof of the ternary Goldbach conjecture relied on computer calculations (finite verifications of the generalised Riemann hypothesis) by Platt

Which languages

- Mathematica. Because it is quick, easy and fun to use for mathematicians and you will learn the functional programming paradigm.
- Python. Because it is quick, easy and fun to use for mathematicians and you will learn the procedural and object oriented programming paradigms.

Java, C, C++ and C# are the most heavily used languages commercially. Python is third and it's popularity has been growing year on year (up from fifth last year). (See `http://www.tiobe.com/tiobe-index/`)

A more realistic assessment

- ~~Once you know one computer language you know them all~~
- The biggest hurdle to computer programming is writing your first program, but you will surmount this hurdle with ease.
- Once you have learned a couple of different languages you won't find it too hard to learn a new language — at least so long as it is well designed.
- It takes years to master most programming languages, but very little time to be highly productive.

A more realistic assessment

- ~~Once you know one computer language you know them all~~
- The biggest hurdle to computer programming is writing your first program, but you will surmount this hurdle with ease.
- Once you have learned a couple of different languages you won't find it too hard to learn a new language — at least so long as it is well designed.
- It takes years to master most programming languages, but very little time to be highly productive.
- The most difficult part of computer programming is **not** the language:
  - The ideas themselves
  - Fixing bugs
  - Testing
  - Maintenance
  - Manageability
  - Working with a team

How?

- One 2 hour lecture-cum-class per week
- One week off (30 October)
- No exam
- Projects at the end of the semester (11 December)
    - You can choose what language to use for your project.
    - Projects must be completed in teams
    - Each team must give a presentation on their project
    - I will invite academics from throughout the CDT.
- All materials will be made available on this website.

Questions

- That's it for motivation. Do you have any questions?

ys

Getting started with Mathematica

- Type `3^2 + 4^2`
- Press `SHIFT` + `ENTER`.
- Type `Sqrt[ \%]`.
- Type `3 ( 4 + 5 )`.
- This mathematical notation is ambiguous $f(x)$.

Variables

- Type the first line of the following without pressing `ENTER`:

Variables

- Type the first line of the following without pressing `ENTER`:

```
x = 3
y = 4
z = x^2 + y^2
```

Variables

- Type the first line of the following without pressing `ENTER`:

```
x = 3
y = 4
z = x^2 + y^2
```

- Use `ENTER` on it's own to separate lines and `SHIFT` + `ENTER` to run the whole block.
- Note the way colour is used to highlight what is defined and what isn't defined.

The meaning of equals

The meaning of equals

- Now try:

```
x = 5
y = 12
z
```

The meaning of equals

- Now try:

```
x = 5
y = 12
z
```

- Depending upon what you wanted to happen you may consider this to be satisfactory or unsatisfactory.

The meaning of equals

- Now try:

```
x = 5
y = 12
z
```

- Depending upon what you wanted to happen you may consider this to be satisfactory or unsatisfactory.
- Perhaps you prefer:

```
x = 3
y = 4
z := x^2 + y^2
```

```
x = Cos[theta];
y = Sin[theta];
z
```

```
x = Cos[theta];
y = Sin[theta];
z
```

- ; means “stop repeating everything I say”

The meaning of equals

- = means “calculate and assign”
- := means “should be calculated on demand as follows”.

Some more examples

```
7 * 8
7 8 (* Note the space between 7 and 8 *)
I^2
Pi
E
Im[ 3 + 7 I ]
Re[ 3 + 7 I ]
```

In Mathematica, the convention is that variables and functions that begin with a capital letter are defined by the system. Your variables should start with lower case.

Typesetting

- Use `(*` and `*)` for comments midway through code
- Use `Right Click->Insert New Cell->Text` for lengthy comments.
- If you want use `Esc q Esc` to type θ. Or for LaTeXfans `Esc \theta Esc`.
- Don't waste your time making things look pretty unless you want to.
- You can create presentations using Mathematica, but I notice that I haven't...

`N`

- N means compute numerically

`N`

- N means compute numerically
- Compare `Sqrt[2]` with `N[Sqrt[2]]` and `N[Sqrt[2],10]` and `Sqrt[2.0]`

`N`

- N means compute numerically
- Compare `Sqrt[2]` with `N[Sqrt[2]]` and `N[Sqrt[2],10]` and `Sqrt[2.0]`
- When performing numerical calculations it's usually quickest to work with numerical values as soon as possible.

`N`

- N means compute numerically
- Compare `Sqrt[2]` with `N[Sqrt[2]]` and `N[Sqrt[2],10]` and `Sqrt[2.0]`
- When performing numerical calculations it's usually quickest to work with numerical values as soon as possible. You might prefer `Sqrt[2] // N`

★ What are the first million digits of π?

`N`

- N means compute numerically
- Compare `Sqrt[2]` with `N[Sqrt[2]]` and `N[Sqrt[2],10]` and `Sqrt[2.0]`
- When performing numerical calculations it's usually quickest to work with numerical values as soon as possible. You might prefer `Sqrt[2] // N`

★ What are the first million digits of π?

- Moral: what you find unbearably tedious a computer may find trivial. Empathy is a bad way to estimate the performance of software.

`Simplify`

- Use the `Simplify` function to get Mathematica to simplify the expression $cos(\theta)^2 + sin(\theta)^2$.
- Use `Simplify` with the postfix format and decide if you like it.

Calculus

You can use the function `D` to perform differentiation.

```
D[ Tan[theta], theta]
D[ f[x], x]
D[ x^x, x]
D[ x^2 Cos[y], x, y]
```

Integration

★ Use the Mathematica help system to work out how to compute the following integrals

$$\int \cos(x)\mathrm{d}x$$

$$\int_0^{\pi} \sin(x)\mathrm{d}x$$

$$\int_{-\infty}^{\infty} e^{-\frac{x^2}{2}}$$

The next exercise is about Mathematica's strengths and weaknesses:

★ Find a function that you know how to integrate but that Mathematica can't integrate.

Lists, vectors, matrices

- Use curly brackets for lists e.g. `{x, -Infinity, Infinity}`

Lists, vectors, matrices

- Use curly brackets for lists e.g. `{x, -Infinity, Infinity}`
- Use a list to represent a vector e.g. `e1 = {1, 0, 0}`

Lists, vectors, matrices

- Use curly brackets for lists e.g. `{x, -Infinity, Infinity}`
- Use a list to represent a vector e.g. `e1 = {1, 0, 0}`
- Use a list of lists to represent a matrix e.g. `{{1,2},{3,4}}`
- Use `.` to multiply matrices, multiply matrices and vectors or compute the dot product.
- Use `MatrixForm` to print matrices prettily.
- Use `Transpose` to create column vectors if desired.

Functions

```
solveQuadratic[a_,b_,c_]:=
    (-b + Sqrt[b^2 - 4 a c ])/(2a)
solveQuadratic[1,2,1]
```

- Note the underscores
- It's usually best to use `:=` when defining functions
- Note the use of parentheses when writing such a complex expression
- Note the way Mathematica colours things in as you type. This can be very helpful.

★ Enhance solve Quadratic so that it returns a list containing both roots of the quadratic. Don't worry about duplicates.

★ Write a Mathematica function called `rotationMatrix`. It should take one parameter θ and return the matrix:

$$\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$$

Use your function to prove the standard formula for $\sin(\theta + \phi)$. Show that Pythagoras's theorem from the fact that $\theta \mapsto \text{rotationMatrix}(\theta)$ is a homomorphism.

Moral

Functions:

- Reduce typing
- Prevent typographical errors
- Make your code much clearer
- Enable reuse of code
- Slogan: “Once and only once”.

More complex functions

```
solveQuadratic[a_,b_,c_]:= Module[ {discriminant, value1, value2},
    discriminant = b ^2 - 4 a c;
    value1 = (-b + discriminant)/(2a);
    value2 = (-b - discriminant)/(2a);
    {value1, value2 }
    ]
```

- We have defined “local variables”
- Note the way Mathematica colours things in
- You can use Module whenever you want temporary variables, not just in function definitions.
- The functions `Block` and `Module` are almost interchangeable. `Block` is faster. `Module` is “safer”.

Workbooks

- Open a new workbook
- What is the value of `z` in this workbook?
- Since this quickly becomes irritating you might want to:
    - Open the Options Inspector (`CTRL+SHIFT+ O`)
    - Change the scope from "Selection" to "Global Preferences"
    - Expand `Cell Options -> Evaluation Options`, and change the `CellContext` setting to `Notebook`

Clearing variables

- Type `ClearAll[x,y,z]` to get rid of the definitions for these variables.
- Sometimes you might want to clear everything and start again. I then
  - Select: `Evaluation->Quit Kernel->Local`.
  - Select: `Evaluation->Evaluate Notebook`.
- As you will see from the options, you can evaluate just parts of the document too.

The sensible way of solving equations

```
Solve[ x^2 + 2 x + 2 == 0, {x}]
Solve[ {2 x + 3 y == -1, 2 x - 4 y + 1 == -1}, {x,y}]
Solve[ 2 x + 3 y == -1 && 2 x - 4 y + 1 == -1, {x,y}]
Solve[ 2 x + 3 y == -1 || 2 x - 4 y + 1 == -1, {x,y}]
Reduce[ x^2 + 2 x + 2 == 0, {x}]
```

- Notice the `==` signs. We've now met `=`, `:=` and `==`.
- Notice the `&&` this means “and”
- Notice the `||` this means “or”

You can easily specify the domain of the variables if you want:

`Solve[ x^2 + 1 == 0, {x}, Reals]`

Exercises

★ What is the general formula for the roots of a quartic equation?

★ Use the `Solve` command over the Integers to show that 2 is irrational.

★ Use the `Solve` command to find all Pythagorean triples. The main challenge is understanding the output. `Reduce` gives a more comprehensible answer.

Plots

The basic command to plot a function is `Plot`.

```
Plot[ Sin[x], {x, -10, 10}]
```

But you can tinker with the output:

```
Plot[Sin[x], {x, -10, 10}, AspectRatio -> Automatic,
 PlotStyle -> {Orange, Dashed, Thick}]
```

★ Use `ContourPlot` to plot a hyperbola $x^2 - y^2 = 1$

★ Use `ParametricPlot` to plot the same hyperbola.

★ Use `ContourPlot` to show how the hyperbola $x^2 - y^2 = a$ depends on the parameter a. Use `Plot3D` to examine the surface defined by $x^2 - y^2 = a$.

★ Use `ContourPlot` to plot two touching circles. Do this in two ways: by passing two equations to contour plot, by thinking up a function whose zero set consists of two touching circles.

- ★ Use `ImplicitPlot3D` to plot a sphere.
- ★ Use `ParametericPlot3D` to plot a Möbius strip.

Interesting surfaces [Hard]

★ Plot a torus using `ContourPlot3D`

★ Plot a genus 2 surface using the approach of your choice.

★ Plot a Klein bottle using the approach of your choice. You may want to use the option `Opacity` and you may want to switch off the Mesh so that you can understand your picture. Use `PlotStyle->Opacity[0.5]` to create a translucent parametric plot and `ContourStyle->Opacity[0.5]` to create a translucent contour plot.

Morse theory

★ Find a map from the unit square to a vertical torus - i.e. a torus oriented so that you would be looking through it if you held it level with your eye. Use ParametericPlot3D to plot this torus. This defines a "height" function on the unit square. Plot the contours of this height function on the square.

★ A "critical point" of a function on the plane is a point where the gradient is zero. What are the critical points in your contour plot? How do they relate to the the 3D picture?

Morse theory continued

★ A non-degenerate critical point of f is a critical point where the 2×2 matrix of partial derivatives:

$$\frac{\partial^2 f}{\partial x_i \partial x_j}$$

does not vanish. Here x_1, x_2 are coordinates on the plane. Recall the classification of conic sections (up to linear transformation of the plane). Give without proof a classification of non-degenerate critical points up to deformation.