University of Dundee

MA32009

MA40001

Philip Murray

2024-01-09

Table of contents

Preface

Welcome to MA40001

My name is Philip Murray and I am the module lead.

How to contact me?

- email: pmurray@dundee.ac.uk
- office: G11, Fulton Building
- Teams: PM me

Installation

- quarto
- Visual studio Code
- latex
- python

Uni machines

From Apps Anywhere, open:

- VS Code with Quarto XXX
- *Anaconda*
- Install the Quarto extension (Extensions tab on left-hand side)
- Install the Python extension
- Show and Run Commands -> Python: Select Interpreter

Quarto

- Download the test file *asdf.qmd*.
- Open the file in Visual Studio Code

Python

From the terminal within Visual Studio Code:

quarto install tinytex

py -m install numpy py -m install matplotlib py -m pip install jupyter

Lecture notes

You can find lecture notes for the module on this page. If you would like a pdf this can be easily generated by clicking on the pdf link of the webpage. I will occasionally edit/update the notes as we proceed through lectures. If you spot any errors, typos or omissions please Raise an Issue

Python codes

I have provided Python codes for most of the figures in the notes (you can unfold code section by clicking 'Code'). Note that the Python code does not appear in the pdf.

Many of you have taken the Introduction to Programming module at Level 2 and have therefore some experience using Python. I strongly encourage you to use the provided codes as a tool to play around with numerical solutions of the various models that we will be working on. The codes should run as standalone Python codes.

Assessment

- Presentation
- Interim report
- Poster
- Final report

Plan

Table 1: Projected delivery

Week	Up to Section	Tutorial sheet	Assessment
1	1.4	1	
2	1.6	1	Quiz 1
3	2.2	2	
4	2.3.7	2	Quiz 2 (Up to 2.2.1)
5	3.3	3	
6	3.5	3	Test 1 (Chapters 1 and 2)
7	4.1	4	
8	4	4	Quiz 3
9	4	5	
10	5	5	Test 2
11	5		

References

Part I

Mathematical/Scientific communication

1 Scientific/mathematical writing

The assessment of your project will involve:

- verbal presentation (~10 minute slide presentation)
- written reports (interim report and final report)
- poster
- viva (aural exam)

The main aims are to develop:

- mathematical skills,
- independent investigation skills
- scientific/mathematical communication skills.

1.1 Common elements across different assessment points

1.1.1 Narrative

A common theme with the different assessment points is the need for a *narrative* around your project:

- can you describe the project in one sentence
- can you outline why the topic is important?
- what is the background to the project?
- what methods/results/techniques have you developed?
- can you summarise your findings?

Once you have settled on a narrative then the different assessment points can be thought of as variations on presentation of your project narrative.

1.1.2 Pitching the content appropriately

1.1.3 Equations

Are equations presented accurately? Are mathematical objects accurately defined? Has sufficient background detail been presented so that the arguments can be reasonably followed?

1.1.3.1 Typesetting equations

To typeset formulae is actually quite difficult. Mathematics uses a variety of symbols and several different alphabets: Roman, Greek, Hebrew are the most common. In addition formulae are often more similar to graphics than to text. There are numerators and denominators which in turn can have fractions etc, as for instance:

$$f := \frac{1}{1+\frac{1}{1+\frac{1}{1+x}}}.$$

To make this formula look good requires either an advanced typesetting program or a lot of effort. Most common typesetting programmes come with some sort of equation editor, but very few can handle such a problem. The most powerful mathematical typesetting program, which is also the format used for almost all mathematical literature is LaTeX. We will learn about LaTeX later.

Like grammar for a language there exist also certain conventions about how to write formulae. Here is a (far from exhaustive) list of the most import conventions:

- Treat the formula like text. If the formula is at the end of a sentence there has to be a full stop at the end of the formula. If another formula follows use a comma or semicolon. This how we count
$$1+0=1, 1+1=2.$$
- use Roman (typically lower case) letters in *italic* style for all variables: x, y, z, a, b, c, both if we refer to them in the text as well as in formulae.
- use Greek letters are used for angles, and e.g. differential forms;
- vectors are typeset in bold, $\mathbf{a}$, or using an arrow, $\vec{a}$;
- typeset functions in roman, $\sin(x)$ rather than $sin(x)$;
- typeset matrices using capital roman letters, e.g. M;
- represent number systems and also certain vector spaces in a style where certain lines are double: $\mathbb{R}$, $\mathbb{Q}$, $\mathbb{C}$, …;
- use curly brackets for sets, e.g. $\{1,2,3\} = \{2,1,3\}$, and regular brackets for an ordered list $(1,2,3) \neq (2,1,3)$;
- denote a range by three dots: $i = 1, 2, ..., n$, (no bracket required);

- use brackets only where necessary, note that multiplication/division takes precedence over addition/subtraction; E.g
$$a + (b \cdot c)$$
does not need the brackets, but
$$(a + b) \cdot c$$
does. Any fraction replaces a bracket, so $\frac{5}{a+b}$ does not need the brackets.
- use a separate line for any formula that uses more than one line.

> **Common mistakes**
>
> Common mistakes are (in addition to violating the above conventions):
>
> - there are typos in the equations
> - the presented system is not mathematically self-consistent (e.g. a system of ODEs is missing initial conditions)
> - variables and parameters are not defined
> - using different font styles and sizes to represent the same quantity;
> - not using the same symbol (or font) in the text and separate formulae;
> - not using spaces between symbols, and/or wrong symbols, e.g axb instead of $a \times b$;
> - forgetting brackets or placing too many brackets.

1.1.4 Figures

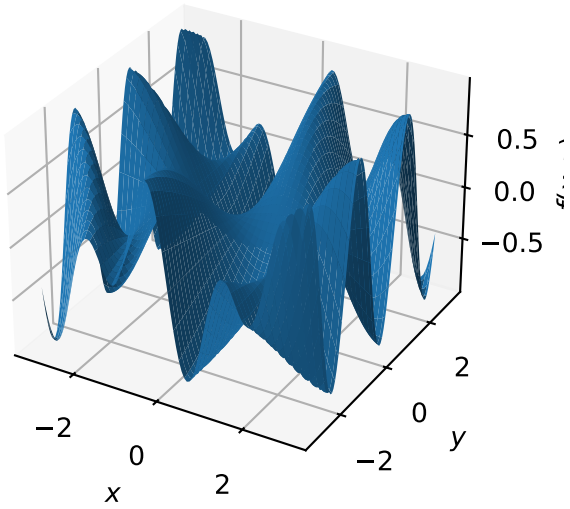
A formula is often the best and most concise way to communicate a relation or function to a mathematically trained audience. However, there are many cases where even a mathematician has difficulties to understand (in the short time available in a presentation) what a function represents. For functions of one and two variables there is always the option to show a plot of the function. Here is an example of a function in two variables:

$$f(x, y) = \cos(x) \sin(xy) \tag{1.1}$$

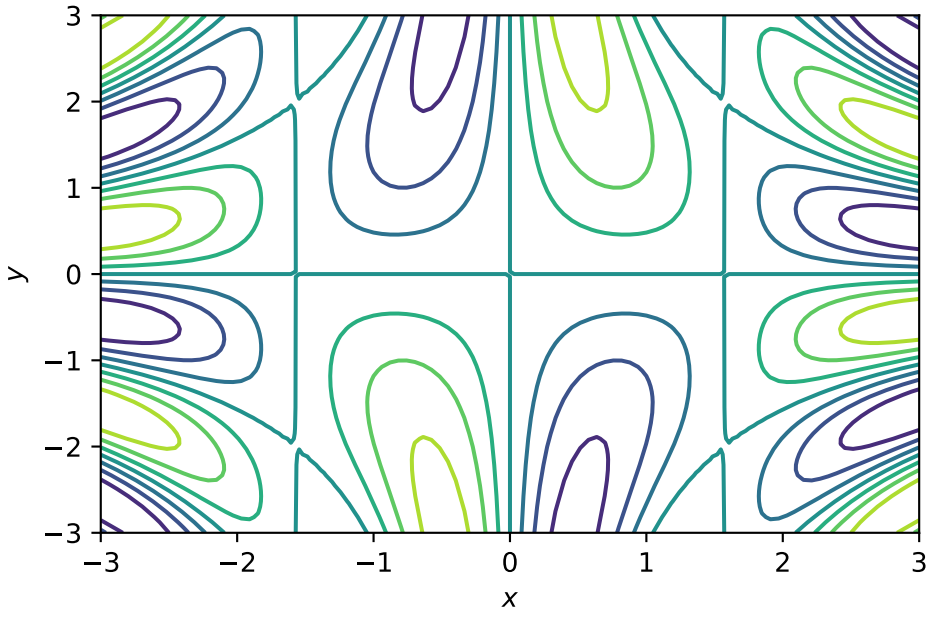
Although this is a comparatively simple function it takes some time to figure out what properties the function has, that is for instance: * Is the function periodic? * How many maxima/minima does it have? * How does it behave for $x, y \to \infty$?

See surface and contour plots in Figure 1.1. A plot can often show these properties quicker but usually in a less accurate way. The advantage of a contour plot is that it is often easier to see the locations of maxima and minima but it is not so easy to see how high the extrema are.

Other types of plots are useful for different purposes:



(a) 3D plot of Equation 1.1



(b)

Figure 1.1

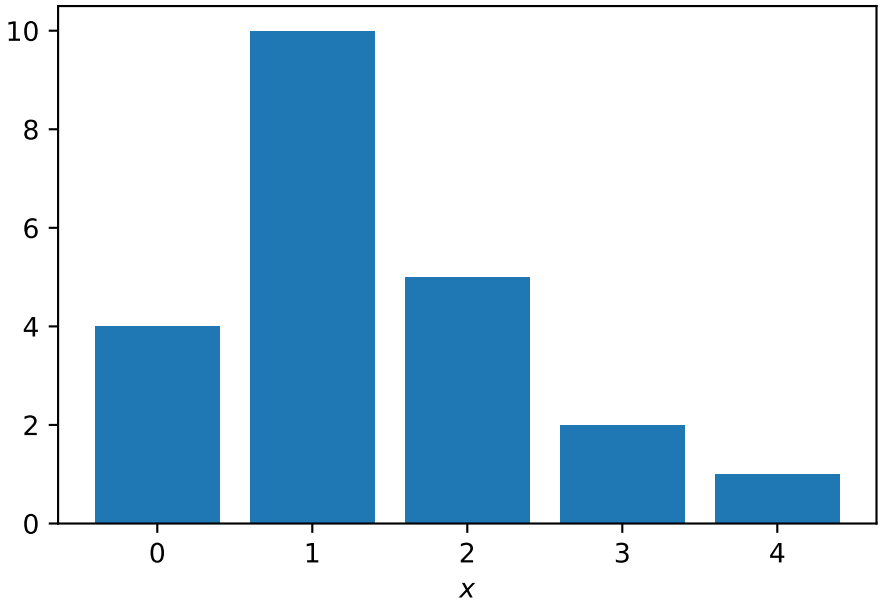


Figure 1.2: A bar chart.

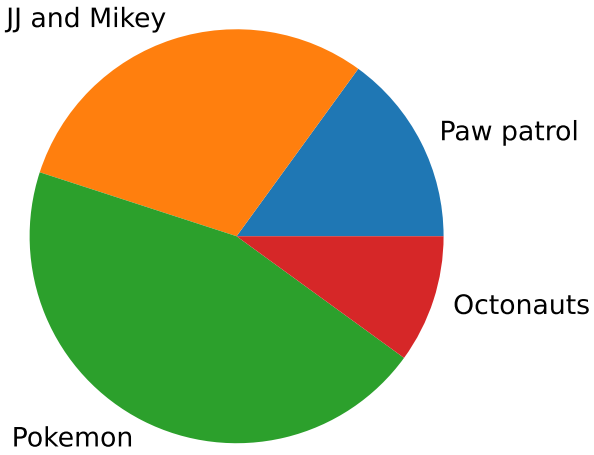


Figure 1.3: A pie chart.

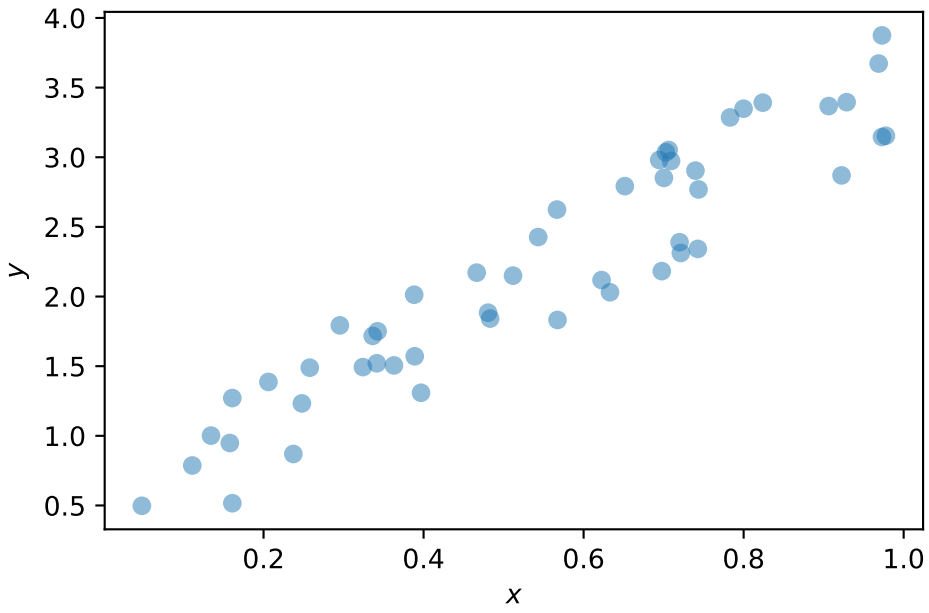


Figure 1.4: A scatter plot.

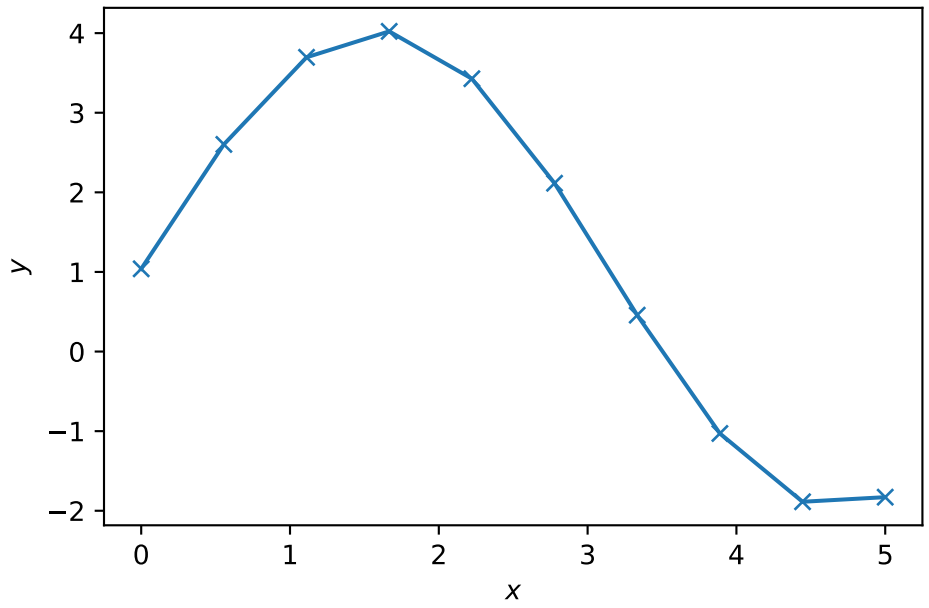


Figure 1.5: y is plotted against x.

- bar chart
- pie chart
- scatter plot
- line chart

> **common mistakes**
>
> - hanging figures (i.e. figures that not placed in context (e.g. connected to the text))
> - figures without axis labels
> - coloured graphs without colour scale/legend
> - labels/ tick marks are too small
> - type of graph unsuitable for the data shown

Some rules -of-thumb:

- check that any text in axis labels/legends is approximately the same size as font in the main text

1.1.5 References

All text in your report must be either original or attributed to the originating author(s). A word-for-word quotation should be indicated with quotation symbols. Such as: The whatever effect causes "extremely aggressive phenotypes" to dominate (Smith et al. 2010). Extensive quotes of more than one sentence should generally be set apart as a separate, often-indented paragraph: According to Tolkien (1954):

"One Ring to rule them all, One Ring to find them, One Ring to bring them all and in the darkness bind them."

It is generally best to limit quotations to no more than 1-3 sentences. Instead, one should paraphrase and/or summarise cited work. Moreover, the majority of the content of a report should be in the authors original words. It should be the authors original logical argument, with citations used to document and justify key assertions and facts. Hence, quotations should be kept to a minimum.

Students must take great care when quoting or paraphrasing text to cite the original source. To do otherwise is *plagiarism*, which undermines the integrity of science, and has serious consequences for the authors as well. (e.g., failing grades, expulsion from a degree program, loss of an academic position, being shunned by the scientific community, etc.)

University of Dundee definition of plagiarism

Plagiarism is the unacknowledged use of another's work as if it were one's own. Examples are:

- the inclusion of more than a single phrase from another's work without the use of quotation marks and acknowledgement of the source;
- summarising another's work by changing a few words or altering the order of presentation without acknowledgement;
- copying another's work;
- the use of another's ideas without acknowledgement.

NB: if you wish to reference your own work, it is important to acknowledge yourself as the source and provide the appropriate reference.

Wikipedia is an excellent resource for learning new topics. However, it is not an acceptable source for citation in a report or publication for various reasons: * the content of wikipedia changes continuously and what we refer to today might not be there tomorrow. , * it is generally not written by experts and often contains errors.

Instead you should try and try and cite primary sources (e.g. research papers or textbooks) as these have permanent bibliographic identifiers (dois).

1.2 Written reports

The overall structure of a report is (generally) as follows:

1. Title page: This should give the title of the report, the author list, and the date.
2. Abstract: provide a short synopsis (~10 sentence) of the project.
3. Introduction: This introduces the overall context and importance of the problem you are addressing. It should give:
    - A basic paragraph or two on background of the problem, its significance, and motivation for the paper. It should make us want to continue reading.
    - In a research-grade paper, you would include information as to why preceding work by others has been insufficient. What work, findings, or improved methodology are required?
    - A summary of any hypotheses that you will develop and justify throughout the paper

- A basic outline of the remainder of the paper, including a note on the methodology (in mathematics, these would be your modelling, analytical, and numerical techniques).

4. Actual content: There should be one or more sections that logically progress your argument and analysis. For example:

   - Formulation of the problem
   - Methods used to solve the problem
   - Results

5. Discussion and Conclusions: Wrap up with a summary of your major results, the significance of your conclusions (including any additional analysis of the results to lay out this significance), and an outlook what else could be done/ or where improvements are required.

> **i** Some common errors in the stucture
>
> - Insufficient development of background (the author assumes that the reader knows more than they do)
> - The aims of the project are not clearly stated
> - The importance of the topic is insufficiently described (e.g. why study this problem?)
> - the discussion is too short

7. References: This section contains the full list of publications that you cited in your report.

8. Appendices (optional): Lengthy and/or tedious calculations or details that are too distracting to the overall flow of your paper, and yet are necessary to fully document your work, should be placed in appendices. You must also provide evidence for any codes that you have developed.

- check that the written text stands alone without the figures (i.e. the figures are helping the reader to understand a point that is being made in the main text.)
- use figure captions to provide techical information about the figure (e.g. what variables are plotted, reference equations in the main text, reference tables of parameter values or details that are essential to reproduce the figure)

1.3 Verbal presentation

2 Version control

3 Getting started with Quarto

3.1

Github is a developer platform is a developer platform that allows developers to create, store, manage and share their code. It is version-controlled, meaning that when you *push* files to the github cloud then you will have access to all previously *pushed* version of those files.

3.2 Github classroom

3.3 Clone the MA40001 repository

3.4 Edit a file and push modified version to github

3.5 Revert a file using version control

Part II

Markdown

4 Quarto

5 Getting started with Quarto

5.1

Quarto is an open source scientific and technical publishing system. It can be used to make a range of publishable outputs (reports, posters, slides, blogs, webpages dashboards etc.). In this module you will earn to use Quarto to write reports and make slide decks.

It is assumed that you are using VSCode with Quarto from Apps Anywhere on the Uni machines. However, you are also encouraged to install Quarto/VSCode on your personal computer.

Quarto documents are written using Markdown (you may have previously used RMArkdown). In the background, Quarto uses an open source document convertor called Pandoc.

5.2 A first markdown document

To begin with let's make a simple markdown document.

At the top of a file we include a YAML block. This provides configuration information for the document.

```
---
title: "Hello, Quarto"
format: html
---

This is a Quarto document
```

1. Open VisualStudio Code.
2. Copy and paste the above code into a new file and save as a .qmd file in a new directory
3. - Compile the code by clicking the preview on the top right of your VSCode editor
   - Open a terminal and type

```
quarto render
quarto preview
```

4. Have a look at the contents on the directory. You shoule be able to see a .html file.

> **i** Quarto on your personal computer
>
> On your personal computer you may need to install Quarto.
> You can find some guidance on how to do this here.

5.3 Creating a pdf

You will need to submit documents in pdf format. To do this you can edit the YAML information as follows.

```
---
title: "Hello, Quarto"
format: pdf
---

This is a Quarto document
```

Repeat the above compiling steps. You should now be able to see that a pdf file has been created.

> **i** LAtex on your personal computer
>
> To generate a pdf your computer must have a version of latex installed. In the Uni machines this has been set up for you. On your personal machines you may need to install latex and configure Quarto.
> You can find some guidance on how to do this here.

5.4 Adding structure to your Quarto documents

Information on document structure can be found on the Quarto help pages. Below I will highlight some of the key document structures that you will need in your project.

5.4.1 Markdown basics

5.4.2 Section headings

```
---
title: " Quarto sections"
format: html
---

# Main section
This is a Quarto document

## Sub section title
```

5.4.3 Tables

5.4.4 Figures

5.4.5 Internal referencing

5.4.6 Citations and references

5.4.7 Submitting code

6 Mathematical typesetting to Quarto

6.1 The Latex environment

6.2 Grammar

7 Presentations

7.1 Slides

Part III

Python

8 Python code development

9 Introduction

It is strongly encouraged that you use and develop programs over the course of your project. You should discuss programming with your project supervisor. In this section we will develop Python codes to solve a number of example problems.

The focus of the examples: - developing programming skills for problem solving - writing your own programmes - using existing packages

It is usually a very useful exercise to try to implement your own version of an algorithm. However, in many cases there already exist well developed codes that we of course should use!

You can use Python via Apps Anywhere (Open Anaconda and then use the SPyder IDE).

Alternatively, you can run Python codes via your Quarto document. On the Uni machines Quarto has been connected to an installation of Python.

9.1 Some key ideas

9.1.1 Variables and datatypes

```
<class 'int'>
<class 'float'>
<class 'str'>
```

9.1.2 Lists and dictionaries

```
[1, 2, 3, 4, 5]
The second entry in the list is
2
```

9.1.3 Logical statements and control loops

9.1.4 Writing functions

A good rule of thumb is that if you find yourself using the same piece of code three or more times you should write a function. This avoids duplication of code.

Suppose we find ourselves manually computing the sum of positive integers many times, i.e.

$$s_2 = 0 + 1 + 2 = 3$$

and

$$s_3 = 0 + 1 + 2 + 3 = 6.$$

It makes sense to write a function that computes the sum for arbitrary n. Then we call that function when needed.

```
3
7140
```

9.2 Using Python libraries

9.2.1 Matrix computation (numpy)

Numpy is a widely used Python libary. It is a standard way to use arrays in Python. Numpy also contains lots of algorithm (e.g. linear algebra, calculus, mathematical functions, integration, random number generation etc.). You can find a beginner's guide here.

9.3 Arrays

Numpy provides tools for calculating many mathematical operations.

```
sin (3.14) is:
0.0015926529164868282
1.2246467991473532e-16
```

We can also use numpy to define and manipulate arrays. In the example below we use python lists to define two 1D arrays.

```
[1 2 3 4 5 6]
The sum of a and b is:
[ 8 10 12 14 16 18]
The first entry in a is


1
```

We can also use numpy for higher dimensional arrays

Calculate the determinant of the 2x2 matrix

$$A = \begin{pmatrix} 4 & 3 \\ 2 & 1 \end{pmatrix}.$$

```
The matrix is:
[[4. 3.]
 [2. 1.]]
The determinant is:
-2.0
```

1. Compute the trace of the matrix A.
2. Compute the determinant of the 3×3 matrix

$$B = \begin{pmatrix} 4 & 3 & 2 \\ 2 & 1 & 4 \\ 3 & 2 & 1 \end{pmatrix}.$$

3. Compute the eigenvalues of B.

9.3.1 Plotting (matplotlib)

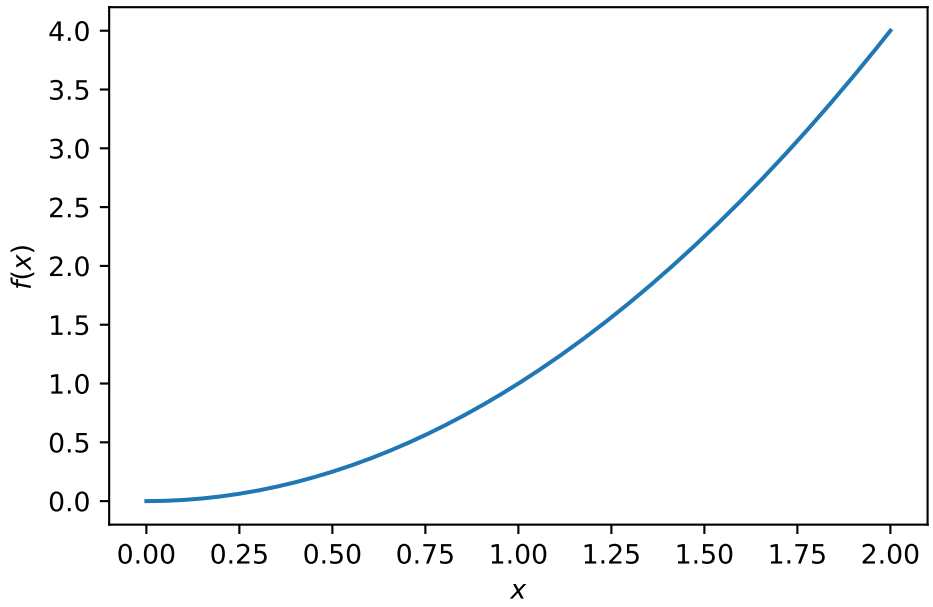
Matplotlib is a pyton library for plotting.

Let's plot some well-known functions.

Suppose we wish to plot the function

$$f(x) = x^2, \quad x \in [0, 2].$$

```
Text(0, 0.5, '$f(x)$')
```



Exercises:

1. Change the domain to
$$x \in [0, 4]$$
.
2. Plot the function
$$f(x) = \sin(x)$$
.
3. Create two subplots.
4. Make the font on the labels larger
5. Add a legend to the figures.

9.3.2 Symbolic computation in Python (sympy)

9.3.3 Numerically solving differential equations (sci-py)

ODEs

PDEs

9.3.4 Optimisation (skikit-learn)

9.3.5 Data analysis (pandas)

9.3.6 Machine learning (tensorflow)

9.4 Writing your own scripts

At some point (either over the course of your project or later) you will likely encounter a problem that cannot be solved using existing code libraries, i.e. you will need to write your own progammes. To prepare for this day, it is a good idea to practice your code development skills on problems where the solution are already known.

9.5 Image analysis (skikit-image)