Chapter 1
Modelling using Ordinary Differential Equations

We will begin by using models that solve *ordinary differential equations*. You will have come across ordinary differential equations before in your maths class within *the natural scientists toolkit* and may remember that one way to solve them is by using a procedure known as integration. A simple example of an ordinary differential equation is as follows:

$$\frac{dy}{dt} = -\lambda y$$

with initial condition that y at time $t = 0$ is equal to N_0. In order to solve this differential equation we rearrange so that the y's are on one side and the t's are on the other and then integrate. Doing this in steps find:

$$\begin{aligned}
\frac{dy}{dt} &= -\lambda y \\
&\therefore \\
\frac{1}{y}dy &= -\lambda dt \\
&\therefore \\
\int \frac{1}{y}dy &= -\lambda \int dt \\
&\therefore \\
\ln y &= -\lambda t + C \\
&\therefore \\
\ln N_0 &= C \\
&\therefore \\
\ln y - \ln N_0 &= = -\lambda t \\
&\therefore \\
y &= N_0 \exp(-\lambda t)
\end{aligned}$$

which you may recognise as the equation for radioactive decay.

However, not all integrals have a solution that can be calculated theoretically and for this reason we can also use *approximate methods* to solve them. A simple method is the forward Euler method, which you will have hear before in the Natural Scientists toolkit. But there are more complicated algorithms that have a high degree of accuracy. We will not go into the details of the methods, but those interested are encouraged to look other textbooks (e.g. Hoffman, 1992).

1.1 Lecture: Tools of the trade

This week need to learn some common terms that will come up each week and learn about a pattern of working that you may not be familiar with, but we hope you will pick up fairly quickly.

We need to become familiar with logging into a remote computer (sometimes called a server) on which we will do our modelling / computational work and from which we will download our results from. The specific things we need to do this week are as follows (see Table 1.1).

Term	Brief description
Terminal	On an Apple Mac we can log in using SSH and SFTP through the terminal. Open up the terminal by searching for it in Spotlight Search.
Command line	On a Windows Machine we can log in using SSH and SFTP through the CMD.exe. Open up the CMD.exe by typing cmd in the search.
Linux commands	The commands you type into a terminal screen to get the server computer to perform tasks
Computer code	A series of instructions that tells a computer what to do.
GitHub	An online place where computer code is stored and can be downloaded.
Git	The name given to the way computer code is versioned (i.e. like track changes for code)
Docker (optional)	This is a way to run a computing environment inside a container. You will be able to install Docker and download a Docker image to run locally on your computer, which will allow you to run the models on your computer, should you wish to.

Table 1.1: Things to become familiar with in the first week.

1.1.1 Types of 'Environmental Models'

This list could be more or less endless. Instead of trying to list them all we will list the models that we are going to use in this course:

- Models that solve ordinary differential equations:
  - These usually arise in situations where there is an initial state, and we are trying to calculate the state at some later time (or perhaps some earlier time).
  - Examples in this course are: (1) *modelling the motion of the planets around the sun*; (2) *modelling the evolution of aerosols and cloud*

droplets and water vapour in the air. We could also extend this to chemical processes that depend on time, for example modelling *air quality*.

- Empirical models
  - These models are usually loosely based on theory, but have parameters that best fit measurements.
  - An example we will use is the *Gaussian Plume Model*, which is based on an analytical solution to how air flows away from a chimney stack; however, it has parameters that cannot be calculated from first principles, which must be measured. Gaussian Plume models are used in air quality modelling for EIA.
- Simplified models with reduced dimensionality.
  - It is often useful to simplify the real world. If you try to model everything the computations become too time consuming.
  - One way of simplifying is to reduce the number of dimensions in your model. An example used in atmospheric modelling is to use a *single column model* which can model how rain falls through the air in the vertical. Such a model only has one spatial dimension (the vertical coordinate).
- Fluid Dynamical Models
  - Modelling the motion of fluids is an important part of environmental science. We can use these methods to model the atmosphere (a low density 'fluid') or the oceans, or even atmospheres on other planets.
  - Weather forecasting uses these techniques.
  - In this course we will use several fluid dynamical models to better understand important effects in the real world.

The above list is by no means complete, and we will learn more as we go along. Most importantly we will have fun!

1.1.2 Terminal and CMD

When on campus you can log into the server based at the University using either Terminal (Mac / ChromeOS) or CMD (windows).

To log in with SSH you open up a terminal or CMD window and type:

```
ssh -Y <username>@130.88.66.57
```

followed by your password.

To log in with SFTP you open up a terminal or CMD window and type:

```
sftp <username>@130.88.66.57
```

followed by your password.

Note, that in place of <username> you should insert your actual username (given to you in the course). When typing in your password you will not see it appear on screen. This is a security feature (passwords are not shown in case someone is watching over your shoulder).

1.1.3 Linux commands

Common operating systems that computers run are MacOS; Windows, iOS, and ChromeOS. The remote computer we are connecting to runs an operating system called Linux. Linux is a common operating system that remote computer run because it supports many users and all the major programming languages. When we connect to a Linux computer using SSH we have to type in commands to tell the computer what to do. It may be unfamiliar to you to interact with a computer by typing in commands, but hopefully you will soon get to grips with it.

Once connected to the server through SSH try typing in some of the commands in Table 1.2, followed by enter. There are too many Linux commands to cover in this course, but we will use more as we go through the course.

What to type (followed by enter)	Brief description
ls	stands for 'list'. This will print the name of any files and directories (folders containing files) to the screen.
pwd	stands for 'print working directory'. This will print to the screen the location of the directory you are in on the remote computer. Forward slashes separate the names of directories.
htop	this will show the status of all the processors on the remote computer. The remote computer will have many processors and is quite powerful. Press 'q' to quit the htop application.
mkdir mydirectory	stands for 'make directory'. This will create a new folder called 'mydirectory' that you can use. Note that new directory names should not have spaces in them or strange characters like apostrophes.
cd mydirectory	stands for 'change directory'. This will change directory to the directory called 'mydirectory'. You will need to have made it first.

Table 1.2: Try typing in the commands when you are logged in via SSH.

1.1.4 Computer code and computer languages

At a low level, computers execute instructions to add, subtract, multiply or divide binary numbers. These instructions are given in machine code, which is hard for humans to understand. If you want to learn the basics of how computers do this I recommend the book called 'But How Do It Know' by J Clark Scott.

For most practical purposes, we need a way of writing instructions that is not too difficult for use to understand and to do that we use computer coding languages such as python, Fortran, c++, Java, and the list goes on.

You have already started to learn python in *the Natural Scientists Toolkit*. Python is what is known as an *Interpreted Language*. Interpreted languages convert each line of code you write to machine code during the running of the code. Python is a great general purpose language and can do nearly anything. However, the fact that each line has to be interpreted before it can be run by the processor means that python can be slow at times. Examples of interpreted languages are: python, MATLAB, BASIC.

The alternative for programs that need to run exceptionally fast is a *compiled language*. Compiled languages take the source code you write and compile the source code into a machine code file (or executable) that can be executed on the computer. An example of such a program is the Met Offices Unified Model, which is written in the Fortran language. Examples of compiled languages are: Fortran, C++, C, Java and many others.

In this course you will use a mixture of interpreted and compiled programs. You wont necessarily be writing your own code, but you will be using these tools so it is useful to know something about them.

1.1.5 GitHub and Git

So that science is reproducible it is extremely important to keeping track of model versions. For instance there may be a small bug fix to some code in a large model, which you think is not really important for some example calculations; however, if you then use the new code to try to reproduce some earlier results you may get completely different results. For these reasons it is important to be able to track the various updates and bug fixes, and be able to return to earlier versions of the code.

In order to do this we use so-called git versioning tools. A git repository is a collection of files that stores all of the code states as they are updated. We do not need to go into the details here, but there will be times during the course that we will use git versioning tools.

Git versioning tools can link to an on-line git repository (a web site that stores a git code repository). A famous on-line git repository is known as GitHub. If you want to be a developer you can get yourself a free account and share your code to others (including yourself). Again, we will not go into the details here, but we will often download code from GitHub each week.

1.1.6 Docker

Note that you only need to bother with Docker if you are working off campus. For some people you may not bother with Docker at all, and only work from on campus—that's fine. But if you'd like the opportunity to practise at home you can install Docker Desktop (if on Windows or Mac); or if on Chromebook you can install Docker by following the instructions on the video for this week.

On windows you should also install git bash. See the video for this week.

On windows you can open up git cmd (after installing git bash) and type:

```
git clone https://github.com/UoM-maul1609/projects-and-teaching
```

this will download the scripts you need to run the modelling environment.

Next install Docker. Once Docker is installed you will open up a CMD / Terminal window and download the modelling environment by typing:

```
docker pull ghcr.io/uom-maul1609/projects-and-teaching-all
```

Next you should change directory to the location of the github repository you have downloaded. From here, to run the modelling environment you can type

```
./scripts/docker-run-win.bat
```

if on windows, or

```
./scripts/docker-run-mac.sh
```

if on mac or

```
./scripts/docker-run-chromeos.sh
```

on ChromeOS

1.1.7 Homework and reading

Before the synchronous activity this week you should do the following:

- Review the notes.
- Open up CMD windows (or terminal on Mac / ChromeOS).
- Log into the server computer using either CMD windows, a terminal (Apple Mac) or terminal (ChromeOS). Type 'ls' and then type 'exit'.

References

Hoffman, J. D., 1992: *"Numerical methods for engineers and scientists"*. "McGraw-Hill".

Latham, J., 1990: Control of global warming? **347**, 339–340.