MATH5702M Statistical Computing

Matthew Aldridge

2026-09-28

Table of contents

About MATH5702

Organisation of MATH5702

This module is **MATH5702M Statistical Computing**.

This module lasts for 11 weeks from 29 September to 10 December 2026. The exam will take place sometime between 11 and 22 January 2027.

The module leader, the lecturer, and the main author of these notes is Dr Matthew Aldridge. (You can call me "Matt", "Matthew", or "Dr Aldridge", pronounced "*old*-ridge".) My email address is m.aldridge@leeds.ac.uk, although I much prefer questions in person at office hours (see below) rather than by email.

The HTML webpage is the best way to view the course material. There is also a **PDF version**, although I have been much less careful about the presentation of this material, and it does not include the problem sheets.

Lectures

The main way you will learn new material for this module is by attending lectures. There are three lectures per week:

- Tuesdays at 1500 in 28 University Road, G.23
- Thursdays at 1100 in 28 University Road, G.23
- Thursday at 1400 in Roger Stevens LT17.

28 University Road can be tricky to find if you've not been there before: here is a Google Maps link; here is a video; here is a map I drew in Powerpoint.

I recommend taking your own notes during the lecture. I will put brief summary notes from the lectures on this website, but they will not reflect all the details I say out loud and write on the whiteboard. Lectures will go through material quite quickly and the material may be quite difficult, so it's likely you'll want to spend time reading through your notes after the lecture. Lectures should be recorded on the lecture capture system; I find it very difficult to read the whiteboard in these videos, but if you unavoidably miss a lecture, for example due to illness, you may find they are better than nothing.

In Weeks 3, 5, 7, 9 and 11, the Thursday @ 11 lecture will operate as a problems class – see more on this below.

Attendance at lectures in compulsory. You should record your attendance using the UniLeeds app and the QR code on the wall in the 15 minutes before the lecture or the 15 minutes after the lecture (but not during the lecture).

Device policy in lectures:

- Please do not use phones in the lecture, unless taking part in a poll/survey or some other activity.
- Please only use a tablet/iPad in lectures if using it to make notes on the lecture.
- **New rule for 2026:** *Please do not use laptops during the lecture*
  - If you need a laptop for accessibility reasons, this is OK, but please let me know (by email or before/after a lecture).
  - If you are the world's fastest LaTeX-typer and you make notes on the lecture on your laptop, this is OK, but please let me know (by email or before/after a lecture).
  - I will hold a poll on this after two weeks – if you all hate the policy, I'll change it back.

(In the *problems classes* use of devices is fine – even encouraged.)

Problem sheets and problem classes

Mathematics and statistics are "doing" subjects! To help you learn material for the module and to help you prepare for the exam, I will provide 5 unassessed problem sheets. These are for you to work through in your own time to help you learn; they are *not* formally assessed. You are welcome to discuss work on the problem sheets with colleagues and friends, although my recommendation would be to write-up your "last, best" attempt neatly by yourself.

There will be an optional opportunity to submit one or two questions from the problem sheet to me in advance of the problems class for some brief informal feedback on your work. See the problem sheets for details.

You should work through each problem sheet in preparation for the problems class in the Thursday @ 11 lecture of Week 3, 5, 7, 9 and 11. In the problems class, you should be ready to explain your answers to questions you managed to solve, discuss your progress on questions you partially solved, and ask for help on questions you got stuck on.

You can also ask for extra help or feedback at office hours (see below).

Coursework

There will be one piece of assessed coursework, which will make up 20% of your module mark. You can read more about the coursework here.

The coursework will be in the form of a worksheet. The worksheet will have some questions, mostly computational but also mathematical, and you will have to write a report containing your answers and computations.

The assessed coursework will be introduced in the **computer practical** sessions later in the semester.

The deadline for the coursework will be the last-but-two day of the Autumn term, *Wednesday 9 December* at 1400. Feedback and marks will be returned on Monday 11 January 2027, the first day of the Spring term.

Office hours

I will run an optional "office hours" drop-in session each week for feedback and consultation. You can come along if you want to talk to me about anything on the module, including if you'd like more feedback on your attempts at problem sheet questions. (For extremely short queries, you can approach me before or after lectures, but my response will often be: "Come to my office hours, and we can discuss it there!")

Office hours will happen on **Thursdays from 1200 to 1300** – so directly after the Thursday lecture / problems class – in my office, which is **MRD 9.10n** in "Maths Research Deck" area on the 9th floor of the EC Stoner building. (One way to the Maths Research Deck is via the doors directly opposite the main entrance to the School of Mathematics; you can also get there from Staircase 1 on the Level 10 "red route" through EC Stoner, next to the Maths Satellite.) If you cannot make this time, contact me for an alternative arrangement.

Exam

There will be one exam, which will make up 80% of your module mark.

The exam will be in the January 2027 exam period (11–22 January); the date and time will be announced in December. The exam will be in person and on campus.

The exam will last 2 hours and 30 minutes. The exam will consist of 4 questions, all compulsory. The exam is a "pen and paper" exam, but you will be allowed to use a permitted calculator in the exam.

Content of MATH5702

Necessary background

I will assume you have studied probability and statistics at the undergraduate level. Strong confidence and high proficiency in basic material is more important than very deep knowledge of more complicated topics. Experience suggests that confidence and proficiency at basic statistics is the most important factor in doing well at this module. (Last year, students with a strong background in statistics had a median score of a Distinction, while students with a weak background in statistics had a median score of a Fail.)

For Leeds undergraduates, MATH2701/2715 Statistical Methods is an official prerequisite, although confidence and proficiency in the more basic material of MATH1700/1710/1712 Probability and Statistics is probably more important.

Some knowledge I will assume:

- **Probability:** Basic rules of probability; random variables, both continuous and discrete; "famous" distributions (especially the normal, exponential, and continuous uniform distributions); expectation, variance, covariance, correlation; law of large numbers and central limit theorem.
- **Statistics:** Estimation of parameters; bias and error; sample mean and sample variance; confidence intervals.

This module will also include an material on Markov chains. I won't assume any pre-existing knowledge of this, and I will introduce all new material we need, but students who have studied Markov chains before (for example in the Leeds module MATH2702 Stochastic Processes / MATH2750 Introduction to Markov Processes) may find a couple of lectures here are merely a reminder of things they already know.

A **"Probability and statistics background: self-test"** is available on Minerva. This test consists of 10 short numerical answer or multiple choice questions. I recommend taking this test to assess your background preparation for MATH5702 – but this is not compulsory, and it is not officially marked for credit.

I expect most students who are well prepared for MATH5702 should get full, or nearly full, marks on this test on the first attempt. If you score 6 or less, I recommend spending significant time revising your basic probability and statistics, and (if you are not on MSc Statistics) you may want to discuss with me whether you think MATH5702 is the right module for you.

The lectures will include examples using the **R** program language. The coursework and problem sheets will require use of R. The exam, while just a "pencil and paper" exam, will require understanding and writing short portions of R code. We will assume basic R capability – that you can enter R commands, store R objects using the `<-` assignment, and perform basic arithmetic with numbers and vectors. Other concepts will be introduced as necessary. If you

want to use R on your own device, I recommend downloading (if you have not already) the R programming language and the program RStudio. (These lecture notes were written in R using RStudio.)

Syllabus

We plan to cover the following topics in the module:

- **Monte Carlo estimation:** definition and examples; bias and error; variance reduction techniques: control variates, antithetic variables, importance sampling. [9 lectures]
- **Random number generation:** pseudo-random number generation using linear congruential generators; inverse transform method; rejection sampling [7 lectures]
- **Markov chain Monte Carlo** (MCMC)**:** [7 lectures]
  - Introduction to Markov chains in discrete and continuous space
  - Metropolis–Hastings algorithm: definition; examples; MCMC in practice; MCMC for Bayesian statistics
- **Resampling methods:** Empirical distribution; plug-in estimation; bootstrap statistics; bootstrap estimation [4 lectures]
- Frequently-asked questions [1 lecture]

Together with the 5 problems classes, this makes 33 lectures.

Book

The following book is strongly recommended for the module:

- J Voss, *An Introduction to Statistical Computing: A simulation-based approach*, Wiley Series in Computational Statistics, Wiley, 2014

The library has electronic access to this book (and two paper copies).

Dr Voss is a lecturer in the School of Mathematics and the University of Leeds, and has taught MATH5702 many times. *An Introduction to Statistical Computing* grew out of his lecture notes for this module, so the book is ideally suited for this module. My lectures will follow this book closely – specifically:

- Monte Carlo estimation: Sections 3.1–3.3
- Random number generation: Sections 1.1–1.4
- Markov chain Monte Carlo: Section 2.3 and Sections 4.1–4.3

- Resampling methods: Section 5.2

For a second look at material, for preparatory reading, for optional extended reading, or for extra exercises, this book comes with my highest recommendation!

Part I

Monte Carlo estimation

1 Introduction to Monte Carlo

A **"Probability and statistics background: self-test"** is available on Minerva. This test consists of 10 short numerical answer or multiple choice questions. I recommend taking this test to assess your background preparation for MATH5702 – but this is not compulsory, and it is not officially marked for credit.

Today, we'll start the first main topic of the module, which is called "Monte Carlo estimation". But first, a bit about the subject as a whole.

1.1 What is statistical computing?

"Statistical computing" – or "computational statistics" – refers to the branch of statistics that involves not attacking statistical problems merely with a pencil and paper, but rather by combining human ingenuity with the immense calculating powers of computers.

One of the big ideas here is **simulation**. Simulation is the idea that we can understand the properties of a random model not by cleverly working out the properties using theory – this is usually impossible for anything but the simplest "toy models" – but rather by running the model many times on a computer. From these many simulations, we can observe and measure things like the typical (or "expected") behaviour, the spread (or "variance") of the behaviour, and other things. This concept of simulation is at the heart of the module MATH5702M Statistical Computing.

In particular, we will look at **Monte Carlo** estimation. Monte Carlo is about estimating a parameter, expectation or probability related to a random variable by taking many samples of that random variable, then computing a relevant sample mean from those samples. We will study Monte Carlo in its standard "basic" form, then look at ways we can make Monte Carlo estimation more accurate (Lectures 1–9).

To run a simulation – for example, when performing Monte Carlo estimation – one needs random numbers with the correct distribution. **Random number generation** (Lectures 10–16) will be an important part of this module. We will look first at how to generate randomness of any sort, and then how to manipulate that randomness into the shape of the distributions we want.

Sometimes, it's not possible to generate perfectly independent samples from exactly the distribution you want. But we can use the output of a process called a "Markov chain" to get "fairly independent" samples from nearly the distribution we want. When we perform Monte Carlo estimation with the output of a Markov chain, this is called **Markov chain Monte Carlo (MCMC)** (Lectures 17–23). MCMC has become a vital part of modern Bayesian statistical analysis.

The final section of the module is about dealing with data. Choosing a random piece of data from a given dataset is a lot like generating a random number from a given distribution, and similar Monte Carlo estimation ideas can be used to find out about that data. We think of a dataset as being a sample from a population, and sampling again from that dataset is known as **resampling** (Lecture 24–27). The most important method of finding out about a population by using resampling from a dataset is called the "bootstrap", and we will study the bootstrap in detail.

MATH5702M Statistical Computing is a *mathematics* module that will concentrate on the *mathematical* ideas that underpin statistical computing. It is not a programming module that will go deeply into the practical issues of the most efficient possible coding of the algorithms we study. But we will want to investigate the behaviour of the methods we learn about and to explore their properties, so will be computer programming to help us do that. We will be using the statistical programming language R, (although one could just as easily have used Python or other similar languages). As my PhD supervisor often told me: "You don't really understand a mathematical algorithm until you've coded it up yourself."

1.2 What is Monte Carlo estimation?

Let X be a random variable. We recall the **expectation** $\mathbb{E}X$ of X. If X is discrete with probability mass function (PMF) p, then the expectation of X is

$$\mathbb{E}X = \sum_x x\, p(x);$$

while if X is continuous with probability density function (PDF) f, then the expectation is

$$\mathbb{E}X = \int_{-\infty}^{+\infty} x\, f(x)\, \mathrm{d}x.$$

More generally, the expectation of a function ϕ of X is

$$\mathbb{E}\,\phi(X) = \begin{cases} \displaystyle\sum_x \phi(x)\, p(x) & \text{for } X \text{ discrete} \\ \displaystyle\int_{-\infty}^{+\infty} \phi(x)\, f(x)\, \mathrm{d}x & \text{for } X \text{ continuous.} \end{cases}$$

(This matches with the "plain" expectation when $\phi(x) = x$.)

But how do we actually *calculate* an expectation like one of these? If X is discrete and can only take a small, finite number of values, then we can simply add up the sum $\sum_x \phi(x)\, p(x)$. But otherwise, we just have to hope that ϕ and p or f are sufficiently "nice" that we can manage to work out the sum/integral using a pencil and paper (and our brain). But while this is often possible in the sort of "toy example" one comes across in maths or statistics lectures, this is very rare in "real life" problems.

Monte Carlo estimation is the idea that we can get an approximate answer for $\mathbb{E}X$ or $\mathbb{E}\,\phi(X)$ if we have access to lots of samples from X. If we have access to $X_1, X_2 \ldots, X_n$, independent and identically distributed (IID) samples with the same distribution as X, then we already know that the mean

$$\overline{X} = \frac{1}{n}(X_1 + X_2 + \cdots + X_n) = \frac{1}{n}\sum_{i=1}^{n} X_i$$

can be used to estimate the expectation $\mathbb{E}X$. We know that $\overline{X}$ is usually close to the expectation $\mathbb{E}X$, at least if if the number of samples n is large; this is justified by the "law of large numbers", which says that $\overline{X} \to \mathbb{E}X$ as $n \to \infty$.

Similarly, we can use

$$\frac{1}{n}(\phi(X_1) + \phi(X_2) + \cdots + \phi(X_n)) = \frac{1}{n}\sum_{i=1}^{n} \phi(X_i)$$

to estimate $\mathbb{E}\,\phi(X)$. The law of large numbers again says that this estimate tends to the correct value $\mathbb{E}\,\phi(X)$ as $n \to \infty$.

In this module we will write that $X_1, X_2, \ldots, X_n$ is a "**random sample** from X" to mean that $X_1, X_2, \ldots, X_n$ are IID with the same distribution as X.

Definition 1.1. Let X be a random variable, ϕ a function, and write $\theta = \mathbb{E}\,\phi(X)$. Then the **Monte Carlo estimator** $\hat{\theta}_n^{\mathrm{MC}}$ of θ is

$$\hat{\theta}_n^{\mathrm{MC}} = \frac{1}{n}\sum_{i=1}^{n} \phi(X_i),$$

where $X_1, X_2, \ldots, X_n$ are a random sample from X.

While general ideas for estimating using simulation go back a long time, the modern theory of Monte Carlo estimation was developed by the physicists Stanislaw Ulam and John von Neumann. Ulam (who was Polish) and von Neumann (who was Hungarian) moved to the US in the early 1940s to work on the Manhattan project to build the atomic bomb (as made famous by the film *Oppenheimer*). Later in the 1940s, they worked together in the Los Alamos National Laboratory continuing their research on nuclear physics generally and nuclear weapons more specifically, where they used simulations on early computers to help them numerically solve difficult mathematical and physical problems.

The name "Monte Carlo" was chosen because the use of randomness to solve such problems reminded them of gamblers in the casinos of Monte Carlo, Monaco. Ulam and von Neumann also worked closely with another colleague Nicholas Metropolis, whose work we will study later in this module.

1.3 Examples

Let's see some simple examples of Monte Carlo estimation using R.

Example 1.1. Let's suppose we've forgotten the expectation of the exponential distribution $X \sim \text{Exp}(2)$ with rate 2. In this simple case, we could work out the answer using the PDF $f(x) = 2\text{e}^{-2x}$ as

$$\mathbb{E}X = \int_0^\infty x \, 2\text{e}^{-2x} \, \text{d}x$$

and, without too much difficulty, get the answer $\frac{1}{2}$. But instead, let's do this the Monte Carlo way.

In R, we can use the `rexp()` function to get IID samples from the exponential distribution: the full syntax is `rexp(n, rate)`, which gives `n` samples from an exponential distribution with rate `rate`. The following code takes the mean of $n = 100$ samples from the exponential distribution.

```
n <- 100
samples <- rexp(n, 2)
MCest <- (1 / n) * sum(samples)
MCest
```

```
[1] 0.5192779
```

So our Monte Carlo estimate is 0.5193, to 4 decimal places.

That's fairly close to the correct answer of $\frac{1}{2}$. But we should (hopefully) be able to get a more accurate estimation if we use more samples. We could also simplify the third line of our code by using the `mean()` function.

```
n <- 1e6
samples <- rexp(n, 2)
MCest <- mean(samples)
MCest
```

```
[1] 0.4997209
```

In the second line, `1e6` is R code for the scientific notation 1×10^6, or a million. I just picked this as "a big number, but where my code still only took a few seconds to run."

Our new Monte Carlo estimate is 0.4997, which is (probably) much closer to the true value of $\frac{1}{2}$.

By the way: all R code "chunks" displayed in the notes should work perfectly if you copy-and-paste them into RStudio. (Indeed, when I compile these lecture notes in RStudio, all the R code gets run on my computer – so I'm certain in must work correctly!) If you hover over a code chunk, a little "clipboard" icon should appear in the top-right, and clicking on that will copy it so you can paste it into RStudio. I strongly encourage playing about with the code as a good way to learn this material and explore further!

Example 1.2. Let's try another example. Let $X \sim \mathrm{N}(1, 2^2)$ be a normal distribution with mean 1 and standard deviation 2. Suppose we want to find out $\mathbb{E}(\sin X)$ (for some reason). While it *might* be possible to somehow calculate the integral

$$\mathbb{E}(\sin X) = \int_{-\infty}^{+\infty} (\sin x) \, \frac{1}{\sqrt{2\pi \times 2^2}} \exp\left(-\frac{(x-1)^2}{2 \times 2^2} \right) \, \mathrm{d}x,$$

that looks extremely difficult to me.

Instead, a Monte Carlo estimation of $\mathbb{E}(\sin X)$ is very straightforward: we just take the mean of the sine of a bunch of normally distributed random numbers. That is we get a random samples $X_1, X_2, \ldots, X_n$ from X; then take the mean of the values

$$\sin(X_1), \sin(X_2), \ldots, \sin(X_n).$$

(We must remember, though, when using the `rnorm()` function to generate normally distributed random variates, that the third argument is the *standard deviation*, here 2, *not* the variance, here $2^2 = 4$.)

```
n <- 1e6
samples <- rnorm(n, 1, 2)
MCest <- mean(sin(samples))
MCest
```

```
[1] 0.1144493
```

Our Monte Carlo estimate is 0.11445.

Next time: *We look at more examples of things we can estimate using the Monte Carlo method.*

Summary:

- Statistical computing is about solving statistical problems by combining human ingenuity with computing power.
- The Monte Carlo estimate of $\mathbb{E}\,\phi(X)$ is

$$\hat{\theta}_n^{\mathrm{MC}} = \frac{1}{n}\sum_{i=1}^{n}\phi(X_i),$$

  where $X_1, \ldots, X_n$ are IID random samples from X.
- Monte Carlo estimation typically gets more accurate as the number of samples n gets bigger.

Read more: Voss, *An Introduction to Statistical Computing*, Section 3.1 and Subsection 3.2.1.

2 Uses of Monte Carlo

Summary:

- The indicator $\mathbb{I}_A(x)$ function of a set A is 1 if $x \in A$ or 0 if $x \notin A$.
- We can estimate a probability $\mathbb{P}(X \in A)$ by using the Monte Carlo estimate for $\mathbb{E}\,\mathbb{I}_A(X)$.
- We can estimate an integral $\int h(x)\,\mathrm{d}x$ by using a Monte Carlo estimate with $\phi(x)\,f(x) = h(x)$.

Read more: Voss, *An Introduction to Statistical Computing*, Section 3.1 and Subsection 3.2.1.

3 Monte Carlo error I: theory

Summary:

- The Monte Carlo estimator is unbiased.
- The Monte Carlo estimator has mean-square error $\mathrm{Var}(\phi(X))/n$, so the root-mean-square error scales like $1/\sqrt{n}$.
- The mean-square error can be estimated by S^2/n, where S^2 is the sample variance of the $\phi(X_i)$.

Read more: Voss, *An Introduction to Statistical Computing*, Subsection 3.2.2.

4 Monte Carlo error II: practice

Summary:

- We can approximate confidence intervals for a Monte Carlo estimate by using a normal approximation.
- To get the root-mean-square error below ϵ we need $n = \mathrm{Var}(\phi(X))/\epsilon^2$ samples.
- We can use a two-step process, where a small "pilot" Monte Carlo estimation allows us to work out how many samples we will need for the big "real" estimation.

Read more: Voss, *An Introduction to Statistical Computing*, Subsections 3.2.2–3.2.4.

5 Control variate

Summary:

- Variance reduction techniques attempt to improve on Monte Carlo estimation making the variance smaller.
- If we know $\eta = \mathbb{E}\,\psi(X)$, then the control variate Monte Carlo estimate is

$$\widehat{\theta}_n^{\mathrm{CV}} = \frac{1}{n}\sum_{i=1}^{n}\left(\phi(X_i) - \psi(X_i)\right) + \eta.$$

- The mean-square error of the control variate Monte Carlo estimate is

$$\mathrm{MSE}\left(\widehat{\theta}_n^{\mathrm{MC}}\right) = \frac{1}{n}\,\mathrm{Var}\left(\phi(X) - \psi(X)\right).$$

Read more: Voss, *An Introduction to Statistical Computing*, Subsection 3.3.3.

6 Antithetic variables I

Summary:

- Estimation is helped by combining individual estimates that are negatively correlated.
- For antithetic variables Monte Carlo estimation, we take pairs of non-independent variables (X, X'), to get the estimator

$$\hat{\theta}_n^{\mathrm{AV}} = \frac{1}{n} \sum_{i=1}^{n/2} (\phi(X_i) + \phi(X_i')).$$

Read more: Voss, *An Introduction to Statistical Computing*, Subsection 3.3.2.

On **Problem Sheet 1**, you should now be able to answer all questions. You should work through this problem sheet in advance of the problems class on *Thursday 15 October* @ 11.

Problem Sheet 1

This is Problem Sheet 1, which covers material from Lectures 1 to 6. You should work through all the questions on this problem sheet in advance of the problems class, which takes place in the lecture of **Thursday 15 October** @ 11.

This problem sheet is to help you practice material from the module and to help you check your learning. It is *not* for formal assessment and does not count towards your module mark.

However, if, optionally, you would like some brief informal feedback on **Questions 4, 6 and 8** (marked), I am happy to provide some. If you want some brief feedback, you should submit your work electronically through Gradescope via the module's Minerva page by **1400 on Tuesday 13 October**. (If you hand-write solutions on paper, the easiest way to scan-and-submit that work is using the Gradescope app on your phone.) I will return some brief comments on your those two questions by the problems class on Thursday 15 October. Because this informal feedback, and not part of the official assessment, I cannot accept late work for any reason – but I am always happy to discuss any of your work on any question in my office hours.

Many of these questions will require use of the R programming language (for example, by using the program RStudio).

Full solutions will be released on Friday 16 October.

A "Probability and statistics background: self-test" is available on Minerva. This test consists of 10 short numerical answer or multiple choice questions. I recommend taking this test to assess your background preparation for MATH5702 – but this is not compulsory, and it is not officially marked for credit.