UNIVERSITY OF CAMBRIDGE

*Mathematical Tripos Part II*

*Astrophysics Tripos Part II*

# Computational Projects

# 2023-24

*CATAM*

# ***Mathematical Tripos Part II***
# ***Astrophysics Tripos Part II***

# **Computational Projects**

July 2023

*Edited by the Computational Projects Assessors Committee*

*Course Director: Prof J.R. Taylor*

*Assistant Course Director: Dr M. Spivack*

**Computer-Aided Teaching of All Mathematics**

**Faculty of Mathematics**

**University of Cambridge**

# Contents

You may choose freely from the projects above, independently of whether you are studying the examinable courses with which your chosen projects are connected. For up-to-date information on the maximum credit for the Computational Projects in Part II of the Mathematical Tripos, and the total number of units required to achieve that maximum, please consult both the Undergraduate Schedules of the Mathematical Tripos and § 2.1 of the Introduction. Please also see § 2.1 of the Introduction for more information on how credit is awarded.

# Introduction

## 1 General

Please read the whole of this introductory chapter before beginning work on the projects. **It contains important information that you should know as you plan your approach to the course.**

### 1.1 Introduction

The course is a continuation of the Part IB Computational Projects course. The aim is to continue your study of the techniques of solving problems in mathematics using computational methods.

As in Part IB, **the course is examined entirely through the submission of project reports**; there are no questions on the course in the written examination papers. The definitive source for up-to-date information on the examination credit for the course is the Faculty of Mathematics Schedules booklet for the academic year 2023-24. At the time of writing (July 2023) the booklet for the academic year 2022-23 states that

> *No questions on the Computational Projects are set on the written examination papers, credit for examination purposes being gained by the submission of reports. The maximum credit obtainable is 150 marks and there are no alpha or beta quality marks. Credit obtained is added directly to the credit gained on the written papers. The maximum contribution to the final merit mark is thus 150, which is the same as the maximum for a 16-lecture course. The Computational Projects are considered to be a single piece of work within the Mathematical Tripos.*

### 1.2 The nature of CATAM projects

CATAM projects are intended to be exercises in independent investigation somewhat like those a mathematician might be asked to undertake in the 'real world'. They are well regarded by external examiners, employers and researchers (and you might view them as a useful item of your *curriculum vitae*).

The questions posed in the projects are more open-ended than standard Tripos questions: **there is not always a single 'correct' response**, and often the method of investigation is not fully specified. **This is deliberate.** Such an approach allows you both to demonstrate your ability to use your own judgement in such matters, and also to produce mathematically intelligent, relevant responses to imprecise questions. You will also gain credit for posing, and responding to, further questions of your own that are suggested by your initial observations. You are allowed and encouraged to use published literature (but it must be referenced, see also §5) to substantiate your arguments, or support your methodology.

### 1.3 Timetable

You should work at your own speed on the projects contained in this booklet, which cover a wide range of mathematical topics.

A lecture covering administrative aspects of the course is given towards the start of the Michaelmas Term. You may also wish to take advantage of programming tutorials available on-line if you did not attempt the Computational Projects course in Part IB.

Your write-ups must be submitted by the second week of the Easter Term (see §6.2 below). Please also note that **you must be available** in the last week of Easter term in case you are called either for a routine *Viva Voce Examination*, or for an *Examination Interview* or an *Investigative Meeting* if unfair means are suspected (see §5.2 below).

### 1.3.1 Planning your work

- You are strongly advised to complete all your computing work by the end of the Easter vacation if at all possible, since the submission deadline is early in the Easter Term.
- **Do not leave writing up your projects until the last minute.** When you are writing up it is highly likely that you will either discover mistakes in your programming and/or want to refine your code. This will take time. If you wish to maximise your marks, the process of programming and writing-up is likely to be **iterative**, ideally with at least a week or so between iterations.
- It is a good idea to write up each project as you go along, rather than to write all the programs first and only then to write up the reports; each year several students make this mistake and lose credit in consequence (in particular note that a program listing without a write-up, or vice versa, gains no credit). You can, indeed should, review your write-ups in the final week before the relevant submission date.

## 1.4 Programming language[s]

As was the case last year, the Faculty of Mathematics is supporting MATLAB for Part II. During your time in Cambridge the University will provide you with a free copy of MATLAB for your computer. Alternatively you can use the version of MATLAB that is available on the Managed Cluster Service (MCS) that is available at a number of UIS and institutional sites around the Collegiate University.

### 1.4.1 Your copy of MATLAB

All undergraduate students at the University are entitled to download and install MATLAB on their own computer that is running Windows, MacOS or Linux; your copy should be used for non-commercial University use only. The files for download, and installation instructions, are available at

`http://www.maths.cam.ac.uk/undergrad/catam/software/matlabinstall/matlab-personal.htm`.

This link is Raven protected. Several versions of MATLAB may be available; if you are downloading MATLAB for the first time it is recommended that you choose the latest version.

### 1.4.2 Programming guides and manual[s]

The Faculty of Mathematics has produced a booklet *Learning to use MATLAB for CATAM project work*, that provides a step-by-step introduction to MATLAB suitable for beginners. This is available on-line at `http://www.maths.cam.ac.uk/undergrad/catam/MATLAB/manual/booklet.pdf`

However, this short guide can only cover a small subset of the MATLAB language. There are many other guides available on the net and in book form that cover MATLAB in far more depth. In addition:

- MATLAB has its own extensive built-in help and documentation.
- The suppliers of MATLAB, *The MathWorks*, provide MATLAB Onramp, an interactive tutorial on the basics which does not require MATLAB installation: see

  `http://uk.mathworks.com/support/learn-with-matlab-tutorials.html`

- *The MathWorks* also provide the introductory guide *Getting Started with MATLAB*. You can access this by 'left-clicking' on the `Getting Started` link at the top of a MATLAB *'Command Window'*. Alternatively there is an on-line version available at

  `http://uk.mathworks.com/help/matlab/getting-started-with-matlab.html`

- Further, *The MathWorks* provide links to a whole a raft of other tutorials; see

  `https://uk.mathworks.com/support/learn-with-matlab-tutorials.html`

  In addition their *MATLAB* documentation page gives more details on maths, graphics, object-oriented programming etc.; see

  `http://uk.mathworks.com/help/matlab/index.html`

- There is a plethora of books on MATLAB. For instance:
  (a) *Numerical Computing with MATLAB* by Cleve Moler (SIAM, Second Edition, 2008, ISBN 978-0-898716-60-3). This book can be downloaded for free from

  `http://uk.mathworks.com/moler/chapters.html`

  (b) *MATLAB Guide* by D.J. Higham & N.J. Higham (SIAM, Second Edition, 2005, ISBN 0-89871-578-4).

  You may be spoilt for choice: Google returns about 100,000,000 hits for the search 'MATLAB introduction', and about 11,000,000 hits for the search 'MATLAB introduction tutorial'.

- The Engineering Department has a webpage that lists a number of helpful articles; see

  `http://www.eng.cam.ac.uk/help/tpl/programs/matlab.html`

### 1.4.3 To Matlab, or not to Matlab

Use of Matlab is recommended,[1] but *you are free to write your programs in any computing language whatsoever.* Python, Julia,[2] R,[3] C, C++, Mathematica,[4] Maple[5] and Haskell have been used by several students in the past, and Excel has been used for plotting graphs of computed results. The choice is your own, provided your system can produce results and program listings for inclusion in your report.[6]

However, you should bear in mind the following points.

- The Faculty does *not* promise to help you with programming problems if you use a language other than Matlab.
- Not all languages have the breadth of mathematical routines that come with the Matlab package. You may discover either that you have to find reliable replacements, or that you have to write your own versions of mathematical library routines that are pre-supplied in Matlab (this can involve a fair amount of effort). To this end you may find reference books, such as *Numerical Recipes* by W. H. Press *et al.* (CUP), useful. You may use equivalent routines to those in Matlab from such works so long as you acknowledge them, and reference them, in your write-ups.
- If you choose a high-level programming language that can perform advanced mathematical operations automatically, then you should check whether use of such commands is permitted in a particular project. As a rule of thumb, do not use a built-in function if there is no equivalent Matlab routine that has been approved for use in the project description, or if use of the built-in function would make the programming considerably easier than intended. For example, use of a command to test whether an integer is prime would not be allowed in a project which required you to write a program to find prime numbers. The *CATAM Helpline* (see §4 below) can give clarification in specific cases.
- Subject to the aforementioned limited exceptions, *you must write your own computer programs.* Downloading computer code, e.g. from the internet, that you are asked to write yourself counts as plagiarism (see §5).

### 1.4.4 Computer Algebra Systems

Some projects require the use of a Computer Algebra System (CAS). At present none is specifically recommended but possible choices include the Symbolic Math Toolbox in Matlab, Mathematica and Maple.

[1] Except where an alternative is explicitly stated, e.g. see footnotes 3 and 5.

[2] Julia is a high-level open source language well suited to numerical computation. An Introduction to Julia for CATAM under ongoing development is available from `https://sje30.github.io/catam-julia/`.

[3] R is a programming language and software environment for statistical and numerical computing, as well as visualisation. It is the recommended language for some Part II projects. R is available for free download for the Linux, MacOS and Windows operating systems from `http://www.r-project.org/`.

[4] Mathematica is a software package that supports symbolic computations and arbitrary precision numerical calculations, as well as visualisation. At the time of writing Mathematica is also available for free to mathematics students, but the agreement is subject to renewal. You can download versions of Mathematica for the Linux, MacOS and Windows operating systems from

`https://www.maths.cam.ac.uk/computing/software/mathematica/`

[5] Maple is a mathematics software package that supports symbolic computations and arbitrary precision numerical calculations, as well as visualisation. It is the recommended language for some Part II projects.

[6] There is no need to consult the *CATAM Helpline* as to your choice of language.

Mathematica and Maple are also sensible choices for several projects other than the ones for which a CAS is actually required, and you should feel free to use them for *any* of the projects, but you should be aware of a few points:

- For intensive numerical calculations Maple should be told to use the hardware floating-point unit (see help on `evalhf`).
- If you choose to use Maple, Mathematica, or any other CAS to do a project for which a CAS is not specifically required, you should bear in mind that you may not be allowed to use some of the built-in functions (see §1.4.3).

## 2 Project Reports

### 2.1 Project write-ups: examination credit

For each project, 40% of the marks available are awarded for writing programs that work and for producing correct graphs, tables of results and so on. A further 50% of the marks are awarded for answering mathematical questions in the project and for making appropriate mathematical observations about your results.

The final 10% of marks are awarded for the overall 'excellence' of the write-up. Half of these 'excellence' marks may be awarded for presentation, that is for producing good clear output (graphs, tables, etc.) which is easy to understand and interpret, and for the mathematical clarity of your report.

The assessors may penalise a write-up that contains an excessive quantity of irrelevant material (see below). In such cases, the 'excellence' mark may be reduced and could even become negative, as low as -10%.

Unless the project specifies a way in which an algorithm should be implemented, marks are, in general, not awarded for programming style, good or bad. Conversely, if your output is poorly presented — for example, if your graphs are too small to be readable or are not annotated — then you may lose marks.

**No marks** are given for the submission of program code without a report, or vice versa.

The marks for each project are scaled so that a possible maximum of 150 marks are available for the Part II Computational Projects course. No quality marks (i.e. $\alpha$s or $\beta$s) are awarded. The maximum contribution to the final merit mark is thus 150 and the same as the maximum for a 16-lecture course.

#### 2.1.1 Examination credit: algorithm applied to the mark awarded for each project

Each project has a unit allocation. The mark awarded for each project is weighted according to the unit allocation, with each project unit equating to a maximum of 5 Tripos marks. The weighted marks for each project are summed to obtain a candidate's total Tripos mark.

To obtain maximum credit, you should submit projects with unit allocations that sum to 30 units. If you submit $N$ units, where $N > 30$ (i.e. if you submit more then the maximum number of units), then the following algorithm applies:

If your weakest project is $M$ units with $M > N - 30$ then the mark on that project will be rescaled by $[M - (N - 30)]/M$. If $M \leqslant N - 30$ then that project will be discarded entirely, a revised $N$ will be calculated, and the algorithm will be applied recursively.

This algorithm ensures that you can score no more than the overall maximum available, i.e. 150 Tripos marks, by reducing the mark only on your weakest project. There is no expectation that you submit more than 30 units; this algorithm is simply a way to calculate your mark if you do.[7]

A fractional total Tripos mark resulting from the weighting/scaling process is rounded up or down to the nearest integer, with an exact half being rounded up.

## 2.2 Project write-ups: advice

Your record of the work done on each project should contain all the results asked for and your comments on these results, together with any graphs or tables asked for, clearly labelled and referred to in the report. However, it is important to remember that the project is set as a piece of mathematics, rather than an exercise in computer programming; thus the most important aspect of the write-up is the **mathematical content**. For instance:

- Your comments on the results of the programs should go *beyond* a rehearsal of the program output and show an understanding of the mathematical and, if relevant, physical points involved. The write-up should demonstrate that you have noticed the most important features of your results, and understood the relevant mathematical background.
- When discussing the computational method you have used, you should distinguish between points of interest in the algorithm itself, and details of your own particular implementation. Discussion of the latter is usually unnecessary, but if there is some reason for including it, please set it aside in your report under a special heading: it is rare for the assessors to be interested in the details of how your programs work.
- Your comments should be pertinent and concise. Brief notes are perfectly satisfactory — provided that you cover the salient points, and make your meaning precise and unambiguous — indeed, students who keep their comments concise can get better marks. An over-long report may well lead an assessor to the conclusion that the candidate is unsure of the essentials of a project and is using quantity in an attempt to hide the lack of quality. Do not copy out chunks of the text of the projects themselves: you may assume that the assessor is familiar with the background to each project and all the relevant equations.
- Similarly you should not reproduce large chunks of your lecture notes; you will not gain credit for doing so (and indeed may lose credit as detailed in §2.1). However, you will be expected to reference results from theory, and show that you understand how they relate to your results. If you quote a theoretical result from a textbook, or from your notes, or from the WWW, you should give both a brief justification of the result and a *full reference*.[8] If you are actually asked to *prove* a result, you should do so concisely.
- Graphs will sometimes be required, for instance to reveal some qualitative features of your results. Such graphs, *including labels, annotations, etc.*, need to be computer-generated

[7] Needless to say, if you submit fewer than 30 units no upwards scaling applies.
[8] See also the paragraph on *Citations* in §5

(see also § 2.3). Further, while it may be easier to include only one graph per page, it is often desirable (e.g. to aid comparison) to include *two or more* graphs on a page. Also, do not forget to clearly label the axes of graphs or other plots, and provide any other annotation necessary to interpret what is displayed. Similarly, the rows and columns of any tables produced should be clearly labelled.

- You should take care to ensure that the assessor sees evidence that **your programs do indeed perform the tasks** you claim they do. In most cases, this can be achieved by including a sample output from the program. If a question asks you to write a program to perform a task but doesn't specify explicitly that you should use it on any particular data, you should provide some 'test' data to run it on and include sample output in your write-up. Similarly, if a project asks you to 'print' or 'display' a numerical result, you should demonstrate that your program does indeed do this by including the output.
- **Above all, make sure you comment where the manual specifically asks you to.** It also helps the assessors if you answer the questions in the order that they appear in the manual and, if applicable, **number your answers** using the same numbering scheme as that used by the project. Make clear which outputs, tables and graphs correspond to which questions and programs.

The following are indicative of some points that might be addressed in the report; they are not exhaustive and, of course not all will be appropriate for every project. In particular, some are more relevant to pure mathematical projects, and others to applied ones.

- Does the algorithm or method always work? Have you tested it?
- What is the theoretical running time, or complexity, of the algorithm? Note that this should be measured by the number of simple operations required, expressed in the usual $O\big(f(n)\big)$ or $\Omega\big(f(n)\big)$ notation, where $n$ is some reasonable measure of the size of the input (say the number of vertices of a graph) and $f$ is a reasonably simple function. Examples of simple operations are the addition or multiplication of two numbers, or the checking of the $(p, q)$ entry of a matrix to see if it is non-zero; with this definition finding the scalar product of two vectors of length $n$ takes order $n$ operations. Note that this measure of complexity can differ from the number of MATLAB commands/'operations', e.g. there is a single MATLAB command to find a scalar product of two vectors of length $n$.
- What is the accuracy of the numerical method? Is it particularly appropriate for the problem in question and, if so, why? How did you choose the step-size (if relevant), and how did you confirm that your numerical results are reliably accurate for all calculations performed?
- How do the numerical answers you obtain relate to the mathematical or physical system being modelled? What conjectures or conclusions, if any, can you make from your results about the physical system or abstract mathematical object under consideration?

In summary, it is the candidate's responsibility to determine which points require discussion in the report, to address these points fully but concisely, and to structure the whole so as to present a clear and complete response to the project. It should be possible to read your write-up without reference to the listing of your programs.

### 2.2.1 Project write-ups: advice on length

The word *brief* peppers the last few paragraphs. To emphasise this point, in general **eight sides of A4 of text**, *excluding in-line graphs, tables, etc.*, should be plenty for a clear concise report of a seven or eight unit project.[9] Indeed, the best reports are sometimes shorter than this.

To this total you will of course need to add tables, graphs etc. However, *do **not** include every single piece of output you generate*: include a selection of the output that is a *representative* sample of graphs and tables. It is up to you to choose a selection which demonstrates all the important features but is reasonably concise. Presenting mathematical results in a clear and concise way is an important skill and one that you will be evaluated upon in CATAM. Twenty pages of graphs would be excessive for most projects, even if the graphs were one to a page.[10] Remember that the assessors will be allowed to **deduct** up to 10% of marks for any project containing an excessive quantity of irrelevant material. Typically, such a project might be long-winded, be very poorly structured, or contain long sections of prose that are not pertinent. Moreover, if your answer to the question posed is buried within a lot of irrelevant material then it may not receive credit, even if it is correct.

## 2.3 Project write-ups: technicalities

As emphasised above, elaborate write-ups are not required. You are required to submit your project reports electronically. In particular, you will be asked to submit your write-ups electronically in Portable Document Format (PDF) form, and you should ensure that the submitted file can be printed (in portrait mode on standard A4 paper). Note that many word processors (e.g. LaTeX, *Microsoft Word*, *LibreOffice*) will generate output in PDF form. In addition, there are utility programs to convert output from one form to another, in particular to PDF form (e.g. there are programs that will convert plain text to PDF). Before you make your choice of word processor, you should confirm that you will be able to generate submittable output in PDF form. Please note that a PDF file including pages generated by scanning a hand-written report or other text document is ***not*** acceptable.

In a very few projects, where a *sketch* (or similar) is asked for, a scanned hand-drawing is acceptable. Such exceptions will be noted *explicitly* in the project description.

If it will prove difficult for you to produce electronic write-ups, e.g. because of a disability, then please contact the *CATAM Helpline* as early as possible in the academic year, so that reasonable adjustments can be made for you.

*Choice of Word Processor.* As to the choice of word processor, there is no definitive answer. Many mathematicians use LaTeX (or, if they are of an older generation, TeX), e.g. this document is written in LaTeX. However, please note that although LaTeX is well suited for mathematical typesetting, it is absolutely acceptable to write reports using other word-processing software, e.g. *Microsoft Word* or *LibreOffice*.

- *Microsoft Word* is commercial, but is available free while you are a student at Cambridge: see

  `https://help.uis.cam.ac.uk/service/collaboration/office365`.

- *LibreOffice* can be installed for free for, *inter alia*, the Windows, MacOS and Linux operating systems from

[9] Reports of projects with fewer/more units might be slightly shorter/longer.
[10] Recall that graphs should not as a rule be printed one to a page.

http://www.libreoffice.org/download/download/.

*LaTeX.* If you decide to use LaTeX, you will probably want to install it on your own personal computer. This can be done for free. For recommendations of TeX distributions and associated packages see

- http://www.tug.org/begin.html and
- http://www.tug.org/interest.html.

*Front end.* In addition to a TeX distribution you will also need a front-end (i.e. a 'clever editor'). A comparison of TeX editors is available on WIKIPEDIA; below we list a few of the more popular TeX editors.

*TeXstudio.* For Windows, Mac and Linux users, there is TeXstudio. The proTeXt distribution, based on MiKTeX, includes the TeXstudio front end.

*TeXworks.* Again for Windows, Mac and Linux users, there is TeXworks. The MiKTeX distribution includes TeXworks.

*TeXShop.* Many Mac aficionados use TeXShop. To obtain TeXShop and the TeXLive distribution see http://pages.uoregon.edu/koch/texshop/obtaining.html.

*TeXnicCenter.* TeXnicCenter is another [older] front end for Windows users.

*LyX.* LyX is not strictly a front end, but has been recommended by some previous students. LyX is available from

http://www.lyx.org/.

However, note that LyX uses its own internal file format, which it converts to LaTeX as necessary.

*Learning LaTeX.* A *Brief LaTeX Guide for CATAM* is available for download from

http://www.maths.cam.ac.uk/undergrad/catam/files/Brief-Guide.pdf .

- The LaTeX source file (which may be helpful as a template), and supporting files, are available for download as a zip file from

  http://www.maths.cam.ac.uk/undergrad/catam/files/Guide.zip .

  Mac, Unix and most Windows users should already have an unzip utility. Windows users can download 7-Zip if they have not.

*Other sources of help.* A welter of useful links have been collated by the Engineering Department on their *Text Processing using LaTeX* page; see

http://www.eng.cam.ac.uk/help/tpl/textprocessing/LaTeX_intro.html.

*Layout of the first page.* The first page of your report should include the **project name** and **project number**.

*Your script is marked anonymously.* Hence, your **name or user identifier should not appear anywhere** in the write-up (including any output).

*Further technicalities.* Please do not use red or green for text (although red and/or green lines on plots are acceptable). Please leave a margin at least 2 cm wide at the left, and number each page, table and graph.

*Program listings.* At the end of each report you should include complete **listings** (i.e. printout of source code) of every major program used to generate your results. You do *not* need to include a listing of a program which is essentially a minor revision of another which you have already included. Make sure that your program listings are the *very last* thing in your reports. Please do not mix program output and program listings together; if you do, the program output may not be marked as part of the report.

## 3 Computing Facilities

You may write and run your programs on any computer you wish, whether it belongs to you personally, to your College, or to the University.

When permitted by COVID protocols, you can also use other computing facilities around the University; for further information (including which Colleges are linked to the MCS network) see[11]

`https://help.uis.cam.ac.uk/service/desktop-services/mcs/mcs-sites`

At most MCS locations you can access the MATLAB software and any files you store on the MCS from one location should be accessible from any other MCS location.

If you believe that do not have access to an adequate computer to complete the CATAM projects, you should contact your Director of Studies and/or the CATAM helpline *well in advance* of any project deadlines.

### 3.1 Backups

Whatever computing facilities you use, **make sure you make regular (electronic and paper) backups of your work** in case of disaster! Remember that occasionally systems go down or disks crash or computers are stolen. **Malfunctions of your own equipment or the MCS are not an excuse for late submissions**: leave yourself enough time before the deadline.

Possibly one of the easiest ways to ensure that your work is backed up is to use an online 'cloud' service; many of these services offer some free space. WIKIPEDIA has a fairly comprehensive list at `http://en.wikipedia.org/wiki/Comparison_of_online_backup_services`. In particular note that eligible students have 5TB of OneDrive personal storage space via their University Microsoft account under a University agreement and unlimited storage via Google Drive (see `https://help.uis.cam.ac.uk/individual-storage`).

## 4 Information Sources

There are many ways of getting help on matters relating to CATAM.

*The CATAM Web Page.* The CATAM web page,

`http://www.maths.cam.ac.uk/undergrad/catam/`

contains much useful information relating to CATAM. There are on-line, and up-to-date, copies of the projects, and any data files required by the projects can be downloaded. There is also the booklet *Learning to use MATLAB for CATAM project work*.

[11] Note that the Phoenix Teaching Rooms and the Titan Room are used during term-times for practical classes by other Departments, but a list of these classes is posted at each site at the start of each term so that you can check the availability in advance (see Opening Hours).

*CATAM News and Email.* Any important information about CATAM (e.g. corrections to projects or to other information in the *Manual*) is publicised via *CATAM News*, which can be reached from the CATAM web page. You must read *CATAM News* from time to time (e.g. just before starting a project) to check for these and other important announcements, such as submission dates and procedures.

As well as adding announcements to *CATAM News*, occasionally we will email students using the year lists maintained by the Faculty of Mathematics. You have a responsibility to read email from the Faculty, and if we send an email to one of those lists we will assume that you have read it.

After 1 October 2023 you can check that you are on the appropriate Faculty year list by referring to the `https://lists.cam.ac.uk/mailman/raven` webpage (to view this page you will need to authenticate using Raven if you have not already done so). You should check that the *Maths-II* mailing list is one of your current lists.

If you are not subscribed to the correct mailing list, then this can be corrected by contacting the Faculty Undergraduate Office (email: `undergrad-office@maths.cam.ac.uk`) with a request to be subscribed to the correct list (and, if necessary, unsubscribed from the wrong list).

*The CATAM Helpline.* If you need help (e.g. if you need clarification about the wording of a project, or if you have queries about programming and/or MATLAB), you can email a query to the *CATAM Helpline*: `catam@maths.cam.ac.uk`. Almost all queries may be sent to the *Helpline*, and it is particularly useful to report potential errors in projects. However the *Helpline* cannot answer detailed mathematical questions about particular projects. Indeed if your query directly addresses a question in a project you may receive a standard reply indicating that the *Helpline* cannot add anything more.

In order to help us manage the emails that we receive,

- please use an email address ending in `cam.ac.uk` (rather than a Gmail, etc. address) both so that we may identify you and also so that your email is not identified as spam;
- please specify, in the subject line of your email, 'Part II' as well as the project number and title or other topic, such as 'MATLAB query', to which your email relates;
- please also **restrict each email to one question or comment** (use multiple emails if you have more than one question or comment).

The *Helpline* is available during Full Term and one week either side. Queries sent outside these dates will be answered subject to personnel availability. We will endeavour (but do not guarantee) to provide a response from the *Helpline* within three working days. However, if the query has to be referred to an assessor, then it may take longer to receive a reply. Please do not send emails to any other address.

*The CATAM FAQ Web Pages.* Before asking the *Helpline* about a particular project, please check the *CATAM FAQ web pages* (accessible from the main CATAM web page). These list questions which students regularly ask, and you may find that your query has already been addressed.

*Advice from Supervisors and Directors of Studies.* The general rule is that advice **must be general in nature**. You should not have supervisions on any work that is yet to be submitted for examination.

## 5 Unfair Means, Plagiarism and Guidelines for Collaboration

The objective of CATAM is for you to learn computational methods, mathematics and written presentation skills. To achieve these objectives, you must work independently on the projects, both on the programming and on the write-ups.

The work that you turn in **must** be your own. This applies equally to the source code and the write-ups, i.e. you must write and test all programs yourself, and all reports must be written *independently.*

Any attempt to gain an unfair advantage, for example by copying computer code, mathematics, or written text, is not acceptable and will be subject to serious sanctions.

If you have any questions about what constitutes unfair means, you should seek advice from the CATAM helpline.

*Citations.* It is, of course, perfectly permissible to use reference books, journals, reference articles on the WWW or other similar material: indeed, you are encouraged to do this. You may quote directly from reference works so long as you acknowledge the source (WWW pages should be acknowledged by a *full* URL). There is no need to quote lengthy proofs in full, but you should at least include your own brief summary of the material, together with a *full* reference (including, if appropriate, the page number) of the proof.

*Programs.* You must write your own computer programs. Downloading computer code, e.g. from the internet, that you are asked to write yourself counts as plagiarism even if cited.

*Acceptable collaboration.* It is recognised that some candidates may occasionally wish to discuss their work with others doing similar projects. This can be educationally beneficial and is accepted provided that it remains within reasonable bounds. Acceptable collaboration may include an *occasional general discussion* of the approach to a project and of the numerical algorithms needed to solve it. Small hints on debugging code (note the *small*), as might be provided by an adviser, are also acceptable.

*Unacceptable collaboration (also known as collusion).* If a general discussion *either* is happening regularly *or* gets to the point where physical or virtual notes are being exchanged (even on the back of an envelope, napkin or stamp), then it has reached the stage of unacceptable collaboration. As an example to clarify the limits of 'acceptable collaboration', if an assessor reading two anonymous write-ups were to see significant similarities in results, answers, mathematical approach or programming which would clearly not be expected from students working independently, then there would appear to be a case that the students have breached the limits. An *Investigative Meeting* would then be arranged (unless such similarities were deemed to be justified in light of the declared lists of discussions, see below). If you are uncertain about what constitutes an *unacceptable collaboration* you should seek advice from the CATAM Helpline.

*Generative AI.* Using generative AI (e.g. ChatGPT, Bing, Bard and similar) to produce some or part of the submitted write-up or source code would not be original work and hence is considered a form of academic misconduct. This interpretation is consistent with *University guidelines*. We use software that is capable of detecting AI-generated content, and where a case of unfair means is suspected, the Examiners may, at their discretion, examine a candidate by means of an Oral Examination.

The following actions are examples of *unfair means*

- copying any other person's program, either automatically or by typing it in from a listing;
- using someone else's program or any part of it as a model, or working from a jointly produced detailed program outline;
- copying or paraphrasing of someone else's report in whole or in part;
- turning in output from a generative AI either in the report or in the source code.

These comments apply just as much to copying from the work of previous Part II students, or another third party (including any code, etc. you find on the internet), as they do to copying from the work of students in your own year. Asking anyone for help that goes past the limits of *acceptable collaboration* as outlined above, and this includes posting questions on the internet (e.g. StackExchange), constitutes *unfair means.*

Further, you should not allow any present or future Part II student access to the work you have undertaken for your own CATAM projects, even after you have submitted your write-ups. If you knowingly give another student access to your CATAM work you are in breach of these guidelines and may be charged with assisting another candidate to make use of unfair means.

## 5.1 Further information about policies regarding plagiarism and other forms of unfair means

*University-wide Statement on Plagiarism.* You should familiarise yourself with the University's *Statement on Plagiarism*.

There is a link to this statement from the University's *Good academic practice and plagiarism* website

`http://www.plagiarism.admin.cam.ac.uk/`,

which also features links to other useful resources, information and guidance.

*Faculty Guidelines on Plagiarism.* You should also be familiar with the Faculty of Mathematics Guidelines on Plagiarism. These guidelines, which include advice on quoting, paraphrasing, referencing, general indebtedness, and the use of web sources, are posted on the Faculty's website at

`http://www.maths.cam.ac.uk/facultyboard/plagiarism/`.

In order to preserve the academic integrity of the Computational Projects component of the Mathematical Tripos, the following procedures have been adopted.

*Declarations.* To certify that you have *read and understood* these guidelines, you will be asked to sign an electronic declaration. Further instructions will be given during Michaelmas Term.

In order to certify that you have *observed* these guidelines, you will be required to sign an electronic submission form provided when you submit your write-ups, and you are advised to read it carefully; it will be similar to that reproduced (subject to revision) as Appendix A. You must list on the form *anybody* (students, supervisors and Directors of Studies alike) with whom you have exchanged information (e.g. by talking to them, or by electronic means) about the projects at any more than a *trivial* level: *any* discussions that affected your approach to the projects to *any* extent must be listed. Failure to include on your submission form any discussion you may have had *is* a breach of these guidelines.

However, declared exchanges are perfectly allowable so long as they fall within the limits of 'acceptable collaboration' as defined above, and you should feel no qualms about listing them. For instance, as long as you have refrained from discussing in any detail your programs or write-ups with others after starting work on them, then the limits have probably not been breached.

The assessors will not have knowledge of your declaration until after all your projects have been marked. However, your declaration may affect your CATAM marks if the assessors believe that discussions have gone beyond the limits of what is acceptable. If so, or if there is a suspicion that your have breached any of the other guidelines, you will be summoned to an *Investigative Meeting* (see §5.2). Ultimately, your case could be brought to the University courts and serious penalties could result (see *Sanctions* below).

*Plagiarism detection.* **The programs and reports submitted will be checked carefully both to ensure that they are your own work, and to ensure the results that you hand in have been produced by your own programs.**

*Checks on submitted program code.* The Faculty of Mathematics uses (and has used for many years) specialised software, including that of external service providers, which automatically checks whether your programs either have been copied or have unacceptable overlaps (e.g. the software can spot changes of notation). All programs submitted are screened.

The code that you submit, and the code that your predecessors submitted, is kept in *anonymised* form to check against code submitted in subsequent years.

*Checks on electronically submitted reports.* In addition, the Faculty of Mathematics will screen your electronically submitted reports using the *Turnitin UK* text-matching software. Further information will be sent to you before the submission date. The electronic declaration which you will be asked to complete at the start of the Michaelmas term will, *inter alia*, cover the use of *Turnitin UK*.

Your electronically submitted write-ups will be kept in *anonymised* form to check against write-ups submitted in subsequent years.

*Sanctions.* If plagiarism, collusion or any other method of unfair means is suspected in the Computational Projects, normally the Chair of Examiners will convene an *Investigative Meeting* (see §5.2). If the Chair of Examiners deems that unfair means were used, the case may be brought to the University courts. According to the Statues and Ordinances of the University [12]

> suspected cases of the use of unfair means (of which plagiarism is one form) will be investigated and may be brought to one of the University courts or disciplinary panels. The University courts and disciplinary panels have wide powers to discipline those found to have used unfair means in an examination, including depriving such persons of membership of the University, and deprivation of a degree.

**The Faculty of Mathematics wishes to make it clear that any breach of these guidelines will be treated very seriously.**

However, we also wish to emphasise that the great majority of candidates have, in the past, had no difficulty in keeping to these guidelines. Unfortunately there have been a small number

[12] From `https://www.admin.cam.ac.uk/univ/so/`.

of cases in recent years where *some individuals have been penalised by the loss of significant numbers of marks, indeed sufficient to drop a class.* If you find the guidelines unclear in any way you should seek advice from the *CATAM Helpline.* These policies and practices have been put in to place so that you can be sure that the hard work you put into CATAM will be fairly rewarded.

### 5.2 Oral examinations

*Viva Voce Examinations.* A number of candidates may be selected, either randomly or formulaically, for a *Viva Voce Examination* after submission of either the core or the additional projects. This is a matter of routine, and therefore a summons to a *Viva Voce Examination* should not be taken to indicate that there is anything amiss. You will be asked some straightforward questions on your project work, and may be asked to elaborate on the extent of discussions you may have had with other students. So long as you can demonstrate that your write-ups are indeed your own, your answers will not alter your project marks.

*Examination Interviews.* Additionally, the Chair of Examiners may summon a particular candidate or particular candidates for interview on any aspect of the written work of the candidate or candidates not produced in an examination room which in the opinion of the Examiners requires elucidation. If plagiarism or other unfair means is suspected, an Investigative Meeting will be convened (see below).

*Investigative Meetings.* When plagiarism, collusion or other unfair means are suspected the Chair of Examiners may summon a candidate to an *Investigative Meeting*[13]. If this happens, you have the right to be accompanied by your Tutor (or another representative at your request). The reasons for the meeting, together with copies of supporting evidence and other relevant documentation, will be given to your Tutor (or other representative). One possible outcome is that the case is brought to the University courts where *serious penalties can be imposed* (see *Sanctions* above).

*Timing. Viva Voce Examinations*, *Examination Interviews* and *Investigative Meetings* are a formal part of the Tripos examination, and if you are summoned then you must attend. These will usually take place during the last week of Easter Full Term. *Viva Voce Examinations* are likely to take place on the Monday of the last week (i.e. Monday 10th June 2024), while *Examination Interviews* and *Investigative Meetings* may take place any time that week. If you are required to attend a *Viva Voce Examination*, an *Examination Interview* and or an *Investigative Meeting* you will be informed in writing just after the end of the written examinations. **You must be available** in the last week of Easter Full Term in case you are summoned.

## 6 Submission and Assessment

In order to gain examination credit for the work that you do on this course, you must write reports on each of the projects that you have done. As emphasised earlier it is the quality (not quantity) of your written report which is the most important factor in determining the marks that you will be awarded.

[13] For more information see
https://www.plagiarism.admin.cam.ac.uk/files/investigative_2016.pdf.

### 6.1 Submission form

When you submit your project reports you will be required to complete and upload the submission form provided, detailing which projects you have attempted and listing all discussions you have had concerning CATAM (see §5, *Unfair Means, Plagiarism and Guidelines for Collaboration*, and Appendix A). Further details, including the definitive submission form, will be made available when the arrangements for electronic submission of reports and programs (see below) are announced.

### 6.2 Submission of written work

In order to gain examination credit, you must:

- submit electronic copies of your reports and programs (see §6.3);
- complete and submit your submission form listing each project for which you wish to gain credit.

Further details about submission arrangements will be announced via *CATAM News* and email closer to the time.

The submission deadline is

**Wednesday 1st May 2024, 4pm.**

After this time, projects may be submitted only under exceptional circumstances. If an extension is likely to be needed, a letter of application and explanation is required from your Director of Studies. The application should be sent to the CATAM Director by the submission date as detailed above.[14]

- Applications must demonstrate that there has been an unexpected development in the student's circumstances.
- Extensions are not normally granted past the Friday of submission week.

A student who is dissatisfied with the CATAM Director's decision, can request within 7 days of the decision, or by the submission date (extended or otherwise), whichever is earlier, that the Chair of the Faculty review the decision.

The Computational Projects Assessors Committee reserves the right to **reduce** the marks awarded for any projects (including reports and source code) which are submitted late.

### 6.3 Electronic submission

You will be required to submit electronically copies of both your reports and your program source files. Electronic submission enables the Faculty to run automatic checks on the independence of your work, and also allows your programs to be inspected in depth (and if necessary run) by the assessors.

[14] Alternatively, the University's procedure can be invoked via the Examination Access and Mitigation Committee; see the *Guidance Notes for Dissertation and Coursework extensions*.

As regards your programs, electronic submission applies whether you have done your work on your own computer, on the MCS, or elsewhere, and is regardless of which programming language you have chosen.

Details of the procedure will be given in advance of the submission deadlines via *CATAM News* and email.

**However please note that you will need to know your UIS password in order to submit copies of your report and program source files.**

If you cannot remember your UIS password you will need to follow that instructions provided by the University Information Service.[15] Note that if you need a Password Reset Token then this may take some time to obtain, so check that you know your UIS password well before submission day.

## 6.4 Saving and sharing electronic files

After the submission deadline the electronic files will be taken offline and you will not be able to download your submitted work from the submission site. We recommend that you keep electronic copies of your work.

Since the manuals will be taken off-line after the close of submission, you might also like to save a copy of the projects you have attempted.

**It is critical that you do not make your reports or source code available to any present or future students. This includes posting to publically accessible repositories such as github.**

Please note that all material that you submit electronically is kept in *anonymised* form to check against write-ups and program code submitted in subsequent years.

## 6.5 Returning from intermission

If a student is returning from intermission that began in an academic year during which they submitted some or all of the CATAM projects, then in certain circumstances it is possible to carry forward some or all of their CATAM marks from that year. Action is required by the Director of Studies. Hence, before attempting any further CATAM work, the student should discuss the options available with their Director of Studies and decide on their intended strategy.

The following general policies have been approved by the Faculty Board. If there are exceptional circumstances in which these seem inappropriate, the Director of Studies should discuss these with the CATAM Director: catam-director@maths.cam.ac.uk.

In the unlikely event that a Part II student submits some CATAM projects in the Easter Term, intermits, and is then allowed to repeat the entire year starting in Michaelmas Term, they should normally be expected to start CATAM afresh as a logical part of repeating the year.

On the other hand, if a Part II student submits some CATAM projects in the Easter Term, then intermits, and then returns at the start of either the Lent Term or the Easter Term, then any marks on projects submitted should be carried over. In addition, the student may submit as many new projects as they wish in the Easter Term of the year they return. If the total number of units submitted is greater than 30, then they will receive credit for their best 30 units, as defined by the standard algorithm. The Director of Studies must notify the Undergraduate Office and the CATAM Director about any marks that are to be carried forward.

[15] See `https://password.csx.cam.ac.uk/forgotten-passwd`.

## A Appendix: Example Submission Form

**PART II MATHEMATICAL TRIPOS 2023-24**

| **Computational Projects 2023** |
|---|
| |

**COMPUTATIONAL PROJECTS**

**STATEMENT OF PROJECTS SUBMITTED FOR EXAMINATION CREDIT**

**Please observe these points when submitting your CATAM projects:**

1. Your name, College or CRSid User Identifier **must not** appear anywhere in the submitted work.
2. Complete this declaration form and submit it electronically with your reports.
3. The Moodle submission site will close at 4pm on submission day and it is likely to be slow immediately prior to the deadline. Please turn in your work earlier if possible and be prepared for delays in the website on submission day.

### IMPORTANT

Candidates are reminded that Discipline Regulation 7 reads:

> No candidate shall make use of unfair means in any University examination. Unfair means shall include plagiarism[16] and, unless such possession is specifically authorized, the possession of any book, paper or other material relevant to the examination. No member of the University shall assist a candidate to make use of such unfair means.

To confirm that you are aware of this, you **must** check and sign the declaration below and include it with your work when it is submitted for credit.

> *The Faculty of Mathematics wishes to make it clear that failure to comply with this requirement is a serious matter that could render you liable to sanctions imposed by the University courts.*

[16] Plagiarism is defined as submitting as one's own work, irrespective of intent to deceive, that which derives in part or in its entirety from the work of others without due acknowledgement.

## DECLARATION BY CANDIDATE

I hereby submit my reports on the following projects and wish them to be assessed for examination credit:

| **Project Number** | **Brief Title** | **Credit Units** |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | **Total Credit Units** | |

I certify that I have read and understood the section *Unfair Means, Plagiarism and Guidelines for Collaboration* in the Projects Manual (including the references therein), and that I have conformed with the guidelines given there as regards any work submitted for assessment at the University. I understand that the penalties may be severe if I am found to have not kept to the guidelines in the section *Unfair Means, Plagiarism and Guidelines for Collaboration.* I agree to the Faculty of Mathematics using specialised software, including *Turnitin UK*, to automatically check whether my submitted work has been copied or plagiarised and, in particular, I certify that

- the composing and writing of these project reports is my own unaided work and no part of it is a copy or paraphrase of work of anyone other than myself;
- the computer programs and listings and results were not copied from anyone or from anywhere (apart from the course material provided);
- I have not shown my programs or written work to any other candidate or allowed anyone else to have access to them;
- I have listed below anybody, other than the CATAM Helpline or CATAM advisers, with whom I have had discussions or exchanged information at any more than a trivial level about the CATAM projects, together with the nature of those discussions and/or exchanges.

**Declaration of Discussions and Exchanges (continue on a separate sheet if necessary)**

Signed ............................................ Date ................................

# 1 Numerical Methods

## 1.1 Fourier Transforms of Bessel Functions (6 units)

*This project assumes only material contained in Part IA and Part IB core courses. Other than that, the project is self contained (although the Part II courses on Numerical Analysis, Further Complex Methods and Asymptotic Methods may provide relevant but non-essential background).*

## 1 Introduction

Bessel's equation of order $n$ is the linear second-order equation

$$x^2y'' + xy' + (x^2 - n^2)y = 0. \tag{1}$$

Bessel functions of the first kind are solutions of (1) which are finite at $x = 0$. They are usually written $J_n(x)$.

**Question 1** Investigate (1) for $n = 0$, 1, 4 using a Runge–Kutta (or similar) method commencing the integration for a strictly positive value of $x$ and a number of different values of $y$ and $y'$ of your choice. You may employ a library routine to solve (1); for example if using MATLAB you can employ the built-in `ode45` routine. Integrate forwards *and* backwards in $x$ for a few such initial conditions, plotting $y$. Describe what you observe, and illustrate any notable behaviour using appropriate plots.

Now try starting at $x = 0$. What happens, and why?

**Question 2** The series solution for $J_n(x)$ is

$$J_n(x) = \sum_{r=0}^{\infty} \frac{(-1)^r(\frac{1}{2}x)^{2r+n}}{r!(n+r)!}. \tag{2}$$

Write a program to sum a truncation of this series. Plot $J_n(x)$ for $n = 0$, 1, 4 for a range of $x$, e.g., for $0 \leqslant x \leqslant 100$. Discuss your choice of truncation, and identify a range of $x$ for which this summation method is not accurate and explain why.

## 2 The Discrete Fourier Transform

The Fourier Transform $\hat{F}(k)$ of a function $F(x)$ may be defined as

$$\hat{F}(k) = \int_{-\infty}^{+\infty} F(x)\exp(-2\pi ikx)\,dx. \tag{3}$$

If $F(x)$ is a function which is only appreciably non-zero over a limited range of $x$, say $0 < x < X$, then it is possible to approximate $\hat{F}(k)$ by means of finite sums. Suppose

$$F_r = F(r\Delta x) \quad \text{for} \quad r = 0, \ldots, (N-1)\ , \quad \text{where} \quad \Delta x = X/N\ . \tag{4}$$

An approximation to (3), known as the Discrete Fourier Transform (DFT), is

$$\hat{\mathcal{F}}_s = \frac{X}{N}\sum_{r=0}^{N-1} F_r\,\omega_N^{-rs}\ , \quad \text{where} \quad \omega_N = e^{2\pi i/N}\ . \tag{5}$$

The *exact* inverse of (5) is

$$F_r = \frac{1}{X} \sum_{s=0}^{N-1} \hat{\mathcal{F}}_s \, \omega_N^{rs}. \tag{6}$$

In order to deduce the relationship between the $\hat{\mathcal{F}}_s$ and $\hat{F}(k)$, we first note from (5) that $\hat{\mathcal{F}}_s$ represents values of the Fourier Transform spaced by the "wavenumber" interval $\Delta k$, where

$$\Delta k = 1/X \ . \tag{7}$$

Also $\hat{\mathcal{F}}_s$ is periodic in $s$ with period $N$; this corresponds to a "wavenumber" periodicity

$$K = N\Delta k = N/X = 1/\Delta x. \tag{8}$$

Now it is to be expected that (5) will fail to approximate to (3) when the exponential function oscillates significantly between sample points, that is when

$$|k| \gtrsim \frac{1}{2\Delta x} = \tfrac{1}{2}K. \tag{9}$$

This, together with its periodicity, suggests that $\hat{\mathcal{F}}_s$ will be related to $\hat{F}(k)$ by

$$\hat{\mathcal{F}}_s \cong \begin{cases} \hat{F}(s\Delta k) & s = 0, \ldots, \frac{1}{2}N - 1, \\ \hat{F}(s\Delta k - K) & s = \frac{1}{2}N, \ldots, N - 1. \end{cases} \tag{10}$$

Thus (6) is an approximation to

$$F(x) \cong \int_{-K/2}^{+K/2} \hat{F}(k) \exp(2\pi i k x) \, dk. \tag{11}$$

Because of the periodicity, the $\hat{\mathcal{F}}_s$ are usually thought of as a series with $s = 0, \ldots, N - 1$, the upper half being mentally re-positioned to correspond to negative "wavenumber". Note that if $F(x)$ is real, and $^*$ denotes a complex conjugate, then

$$\hat{F}(k) = \hat{F}^*(-k). \tag{12}$$

**Question 3** Carefully discuss under what limiting conditions for both $N$ and $X$ (possibly after a suitable change in origin in $x$), does the DFT tend to the Fourier Transform?

## 3 The Fast Fourier Transform

The Fast Fourier Transform (FFT) method provides an efficient way to evaluate the DFT. This method involves effcient evaluation of sums of the form

$$\lambda_s = \sum_{r=0}^{N-1} \mu_r \, \omega_N^{\sigma rs} \ , \quad s = 0, \ldots, N - 1 \ , \quad \sigma = \pm 1 \ , \tag{13}$$

where $N$ is an integer and the $\mu_r$ are a known sequence. The "fast" in FFT requires $N$ to be a power of a small prime, or combination of small primes; for simplicity we will assume that $N = 2^M$.

A brief outline of the FFT method is given in the Appendix. However, it is not necessary to understand the implementation details, since you may use the MATLAB one-dimensional Fast

Fourier Transform function `fft` (which has inverse `ifft`), or an equivalent routine in any other package. Alternatively you may write your own routine (however do ***not*** simply compute the series (5) and (6) simplemindedly*).

Note that MATLAB's `fft` function will work for any value of $N$ although it works best when $N$ is a power of 2. Further details can be found on the `fft` MATLAB `help` page.

## 4 Fourier Transforms of Bessel Functions

**Question 4** Show analytically that if $F(x)$ is a real even function and

$$I_1 = \int_0^X F(x)\exp(-2\pi ikx)\,dx\,, \qquad I_2 = \int_{-X}^{+X} F(x)\exp(-2\pi ikx)\,dx, \tag{14a}$$

then

$$\mathrm{Im}(I_2) = 0\,, \qquad \mathrm{Re}(I_2) = 2\mathrm{Re}(I_1). \tag{14b}$$

With the definitions of §2 and §3, the FFT algorithm is ideally suited to approximating $I_1$ rather than $I_2$. Hence if an approximation to $I_2$ is desired, an approximation to $I_1$ could first be calculated, and then the relations (14b) could be used. If this procedure for calculating $I_2$ is adopted, and $F_N \neq F_0$, explain why $F_0$ should be replaced by $\frac{1}{2}(F_0 + F_N)$ before calculating the DFT. What is the equivalent result to (14b) if $F(x)$ is a real odd function?

**Question 5** Using a FFT code, and the results of question 4, find numerically the Fourier Transform of $J_n(x)$:

$$\hat{J}_n(k) = \int_{-\infty}^{+\infty} J_n(x)\exp\left(-2\pi ikx\right)dx\ . \tag{15}$$

Compare it with the theoretical formula

$$\hat{\mathcal{J}}_n(k) = 2(-i)^n(1-4\pi^2k^2)^{-1/2}\,T_n(2\pi k), \tag{16}$$

where $T_n(\mu)$ is the Chebyshev polynomial of order $n$ defined by

$$T_n(\mu) = \begin{cases} \cos n\theta, & \mu = \cos\theta\,; \\ 0, & |\mu| > 1\,. \end{cases} \tag{17}$$

To obtain $J_n(x)$, you may either devise a method of your own (e.g., a combination of questions 1 and 2), or you may use the MATLAB procedure `besselj`.

You should obtain results for $n$ = 0, 1, 2, 4, and 8. Choose sufficient points in the transform to adequately resolve the functions.

Plots of $J_n(x)$ for a few representative values of $n$ should be included in your write-up. You should also include plots of $\hat{J}_n$ and $\hat{\mathcal{J}}_n$ on the same graph. Choose a range of $k$ which allows you to see the detailed behaviour in the interval $-1 \leqslant \pi k \leqslant 1$.

Comment on your results and discuss their accuracy. Discuss how the FFT deals with any values of $k$ which might be expected from the theoretical result to give problems. You

* You will not receive credit if you do not use the FFT method.

should also describe the effects of varying $N$ and $X$; in particular you should ***systematically*** examine how the numerical errors change as $N$ and/or $X$ are varied, e.g. in the light of your answer to question 3.

You should also find a way to demonstrate from your computational results how the execution time necessary to calculate the transform varies with $N$, and how this compares with the theoretical prediction. To do this accurately you should use in-built timing subroutines in Matlab, Python or similar. *Hint:* given the speed of current computers, timing a *single* run of your program is likely to be dominated by start/end overheads.

## Appendix: The Fast Fourier Transform

The Fast Fourier Transform (FFT) technique is a quick method of evaluating sums of the form

$$\lambda_r = \sum_{s=0}^{N-1} \mu_s\, \omega_N^{\sigma r s}, \qquad r = 0, \ldots, N-1, \qquad \sigma = \pm 1, \tag{18}$$

where $N$ is an integer, $\mu_s$ is a known sequence and $\omega_N = e^{2\pi i/N}$. The "fast" in FFT depends on $N$ being a power of a small prime, or combination of small primes; for simplicity we will assume that $N = 2^M$. Write

$$\lambda_r \longleftrightarrow \mu_s, \qquad r, s = 0, \ldots, N-1 \tag{19}$$

to denote that (18) is satisfied. Introduce the half-length transforms

$$\left.\begin{aligned} \lambda_r^E &\longleftrightarrow \mu_{2s} \\ \lambda_r^O &\longleftrightarrow \mu_{2s+1} \end{aligned}\right\} \qquad r, s = 0, \ldots, \tfrac{1}{2}N - 1; \tag{20}$$

then it may be shown that

$$\left.\begin{aligned} \lambda_r &= \lambda_r^E + \omega_N^{\sigma r} \lambda_r^O \\ \lambda_{r+N/2} &= \lambda_r^E - \omega_N^{\sigma r} \lambda_r^O \end{aligned}\right\} \qquad r = 0, \ldots, \tfrac{1}{2}N - 1. \tag{21}$$

Hence if the half-length transforms are known, it costs $\frac{1}{2}N$ products to evaluate the $\lambda_r$.

To execute an FFT, start from $N$ vectors of unit length (i.e., the original $\mu_s$). At the $s$th stage, $s = 1, 2, \ldots, M$, assemble $2^{M-s}$ vectors of length $2^s$ from vectors of length $2^{s-1}$ – this "costs" $2^{M-s} \times \frac{1}{2}(2^s) = 2^{M-1} = \frac{1}{2}N$ products for each stage. The complete discrete Fourier transform has been formed after $M$ stages, i.e., after $O(\frac{1}{2}N \log_2 N)$ products. For $N = 1024 = 2^{10}$, say, the cost is $\approx 5 \times 10^3$ products, compared to $\approx 10^6$ products in naive matrix multiplication.

A description and short history of the FFT are given in Chapter 12 of the book *Numerical Recipes* by Press *et al.*

# 1 Numerical Methods

## 1.6 Multigrid Methods (10 units)

*Knowledge of Part II Numerical Analysis would be advantageous for this project.*

## 1 Solution of Poisson's Equation by Relaxation Methods

We consider the problem of solving Poisson's equation in a square domain with homogeneous Dirichlet boundary conditions

$$\nabla^2 u = f \qquad \text{in} \qquad 0 < x < 1, \quad 0 < y < 1, \tag{1}$$

with $u = 0$ on $x = 0$, $x = 1$, $y = 0$ and $y = 1$.

A numerical solution is attempted by finding values for $u$ at grid points in a square $N \times N$ mesh. The $(i, j)$th point is given by $(x_i, y_j) = (ih, jh)$ where $h = 1/(N-1)$. The value of $\nabla^2 u$ is approximated at each of the interior points by a finite-difference formula

$$(\nabla^2 u)_{i,j} \simeq \frac{1}{h^2}\left[u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}\right]. \tag{2}$$

By requiring that $(\nabla^2 u)_{i,j}$ is equal to $f(x_i, y_j)$ at each of the interior points, we obtain $(N-2)^2$ linear equations for the $(N-2)^2$ unknowns $u_{i,j}$, $(1 \leqslant i \leqslant N-2,\ 1 \leqslant j \leqslant N-2)$, of the form

$$\frac{1}{h^2}\left[u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}\right] = f(x_i, y_j)\ . \tag{3}$$

The values of $u_{i,j}$ at the boundary points are set by the boundary conditions. Here $u_{i,j}$ is equal to zero at each boundary point.

We now have to solve these linear equations as quickly and as accurately as possible. Note that even if the solution of the linear equations were obtained with perfect accuracy, it would still be only an approximate solution to the original partial differential equation, since (2) is only an approximation to equation (1).

For larger values of $N$ it is impractical to solve the $(N-2)^2$ linear equations by direct methods, such as Gaussian elimination, because of storage limitations. An alternative approach is to use an iterative "relaxation" method. Equation (3) may be reordered to suggest the iteration scheme

$$u_{i,j}^{n+1} = \frac{1}{4}\left[u_{i-1,j}^{n+1} + u_{i,j-1}^{n+1} + u_{i+1,j}^{n} + u_{i,j+1}^{n} - h^2 f(x_i, y_j)\right] \quad \text{for } i, j = 1, \ldots, N-2\,, \tag{4}$$

where the superscripts denote the number of the iteration; this is conventionally called the Gauss-Seidel scheme. Note the appearance of $(n+1)$th iterates on the right-hand side. The calculation works through the grid with $i$ and $j$ increasing, and updated values are used as soon as they become available.

**Question 1** Take $f(x, y) = x^2(1-x)y^3(1-y)$. Write a program to apply the Gauss-Seidel scheme (4) to solve (3) on an arbitrary sized $(N \times N)$ mesh. Use your program to investigate the convergence properties of the scheme as $N$ varies. In particular, after a reasonably large number of iterations you should calculate:

(a) the variation over the grid of the residual error, $\epsilon_{i,j}^n$, where the residual error is the amount by which (3) is not satisfied, i.e.

$$\epsilon_{i,j}^n = \frac{1}{h^2}\left[u_{i+1,j}^n + u_{i-1,j}^n + u_{i,j+1}^n + u_{i,j-1}^n - 4u_{i,j}^n\right] - f(x_i, y_j)\,, \tag{5}$$

(b) an approximation for the asymptotic rate of convergence, i.e.

$$r_\infty = -\log\left(\lim_{n\to\infty}\left\{\frac{\max_{i,j}|\epsilon_{i,j}^{n+1}|}{\max_{i,j}|\epsilon_{i,j}^n|}\right\}\right)\,. \tag{6}$$

Why is this a good definition of the rate of convergence?

What do you conclude about the number of iterations needed for convergence to a specified accuracy (e.g. for the magnitude of residual error to be less than a given tolerance at each point)? Estimate as a power of $N$ the number of operations (i.e. additions, multiplications and divisions) needed for such convergence. Check your answer by measuring the computational time in different cases.[*] Suggested values for $N$ that you might try are 9, 17, 33, 65, etc. Also estimate as a power of $N$ the number of operations needed for convergence to an accuracy consistent with the truncation error of the discretisation (3) of equation (1).

## 2 The Multigrid Method

Your calculations should show that the part of the error that decays slowest for each $N$ (and therefore that which dominates after a large number of iterations) has a form very similar to the lowest Fourier mode that will fit into the domain. The convergence is thus limited by large scales, not by small scales.

This motivates the multigrid method described below. The basic idea is that the error left after a few iterations is on scales much larger than the grid scale. The correction needed to the approximate solution to remove this error may therefore be determined more efficiently by transferring the error to a coarser grid, iterating on the coarser grid where convergence is more rapid, then transferring the calculated correction back to the finer grid, updating the approximate solution, and iterating on the finer grid again. The whole procedure is then repeated until the required convergence is achieved. Furthermore the procedure need not be confined to two grids. It is natural to improve the convergence of the coarse grid problem by transferring the error in that to a coarser grid still, and so on.

The multigrid procedure may be defined more exactly as follows. Assume that we have a sequence of $K$ grids, labelled by $J = 1, \ldots, K$ in increasing order of fineness, the $J$th grid having size $N_J \times N_J$. It is convenient to take the mesh spacing of the $(J-1)$th grid to be twice that of the $J$th grid, i.e. $N_J = 2N_{J-1} - 1$.

On the $J$th grid we wish to solve the linear system

$$\mathcal{L}_J \mathbf{u}^{(J)} = \mathbf{r}^{(J)}\,, \tag{7}$$

where the operator $\mathcal{L}_J$ corresponds to that acting on the left-hand side of (3), if $N_J = N$. Note that it is important when writing down the form of $\mathcal{L}$ for arbitrary $J$ to remember that $h$ in (3) must be replaced by $1/(N_J - 1)$.

[*] *Hint:* given the speed of current computers, the timing of a *single* run of your program might be dominated by start/end overheads.

*Descending part of multigrid cycle*

(A) Apply the Gauss-Seidel iteration scheme (hereafter G-S) $\nu_1$ times to obtain an approximate solution $\tilde{\mathbf{u}}^{(J)}$. The error $\mathbf{v}^{(J)}$ in this solution therefore satisfies

$$\mathcal{L}_J\mathbf{v}^{(J)} = \mathbf{r}^{(J)} - \mathcal{L}_J\tilde{\mathbf{u}}^{(J)}. \tag{8}$$

(B) Transfer the problem of determining $\mathbf{v}^{(J)}$ to the coarser $(J-1)$th grid as

$$\mathcal{L}_{J-1}\mathbf{u}^{(J-1)} = \mathcal{R}(\mathbf{r}^{(J)} - \mathcal{L}_J\tilde{\mathbf{u}}^{(J)}) = \mathbf{r}^{(J-1)}\,, \tag{9}$$

where the operator $\mathcal{R}$ is known as the *restriction* operator (see below).

The descending part of the cycle repeats (A) and (B), transferring the correction at each stage to coarser and coarser grids, starting with $J = K$ and ending with $J = 2$.

*Coarsest grid*

(C) On the coarsest grid apply G-S $\nu_2$ times to obtain an approximate solution $\tilde{\mathbf{u}}^{(1)}$.

*Ascending part of cycle*

(D) Transfer the approximate solution on the $(J-1)$th grid to the $J$th grid to give a new approximation to the solution to the problem on that grid

$$\tilde{\mathbf{u}}^{(J)}_{new} = \tilde{\mathbf{u}}^{(J)}_{old} + \mathcal{P}\tilde{\mathbf{u}}^{(J-1)}\,, \tag{10}$$

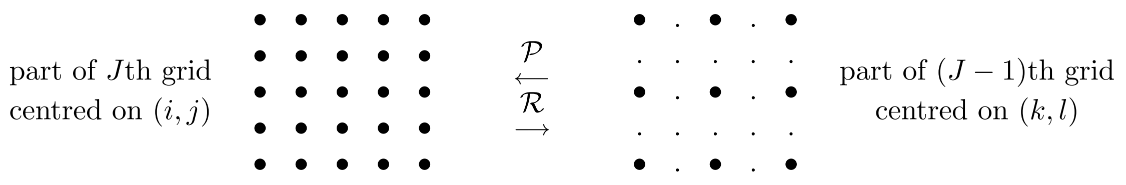
where $\mathcal{P}$ is the prolongation operator (see below).

(E) Apply G-S $\nu_3$ times on the $J$th grid to improve the approximation $\tilde{\mathbf{u}}^{(J)}$.

The ascending part of the cycle repeats (D) and (E), starting with $J = 2$ and ending with $J = K$ to leave an approximate solution to the full problem.

Note that within each multigrid cycle, the approximate solution $\tilde{\mathbf{u}}^{(J)}$ and the right-hand side $\mathbf{r}^{(J)}$ are generated from the problem on the $(J+1)$th grid during the descending part of the cycle and must be stored for use again at the $J$th level during the ascending part of the cycle. Each $\mathbf{r}^{(J)}$ changes from cycle to cycle, except $\mathbf{r}^{(K)}$ which is always equal to $\mathbf{f}^{(K)}$ (i.e. the vector whose elements are $f$ evaluated at each internal point of the $K$th grid).

It remains to specify the restriction and prolongation operators $\mathcal{R}$ and $\mathcal{P}$ that you should use. It is natural to take both to be linear operators. Consider the following two sets of points.


part of $J$th grid centred on $(i,j)$
$\mathcal{P}$
$\longleftarrow$
$\mathcal{R}$
$\longrightarrow$
part of $(J-1)$th grid centred on $(k,l)$

That on the left is a set of points in the $J$th grid with the centre point labelled $(i,j)$. That on the right is the same region in the $(J-1)$th grid. In the latter only those points marked with a • are included in the grid, with the centre point now labelled $(k,l)$ say.

The prolongation operator $\mathcal{P}$ maps a function defined on points in the $(J-1)$th grid onto the points in the $J$th grid. Similarly the restriction operator $\mathcal{R}$ maps a function defined on points in the $J$th grid onto the points in the $(J-1)$th grid. It is convenient to represent both by the "masks"

$$\mathcal{P} = \begin{bmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{bmatrix}, \qquad \mathcal{R} = \begin{bmatrix} \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \end{bmatrix}.$$

That for $\mathcal{P}$ means that if $f = 0$ for all points in the $(J-1)$th grid except that labelled $(k, l)$, then $\mathcal{P}f$ will be zero at all points on the $J$th grid except for the square of nine points centred on $(i, j)$ where it will take the values in the "mask". $\mathcal{P}f$ may be evaluated for general functions $f$ by linearity. Each number in the mask for $\mathcal{R}$ represents the contribution from a point in the $J$th grid, e.g. the square of nine points centred on $(i, j)$ to the point $(k, l)$ in the $(J-1)$th grid; note that points outside this square make no contribution.

**Question 2** Write a program to apply the multigrid method as specified above. You will probably find it useful to have separate procedures/subprograms, working on grids of arbitrary resolution, to carry out each of the operations of prolongation and restriction, to calculate the residual in the difference equations and to apply the Gauss-Seidel iteration (exploit your existing program from question 1 here).

Apply the multigrid method to the solution of the same equation as in question 1. Investigate the rate of convergence associated with a single multigrid cycle for a fixed resolution of the finest grid, particularly its dependence on

(i) the resolution of the coarsest grid;

(ii) the number of times that the G-S iteration is applied at each stage, i.e. $\nu_1$ (on each grid during the descending cycle), $\nu_2$ (on the coarsest grid), and $\nu_3$ (on each grid during the ascending cycle).

To start with, a suggested value for $N_K$ is 65, for $N_{K-1}$ is 33, etc. In each case estimate the total number of operations in a complete cycle, and give a measure of the numerical efficiency. ***Justify carefully the measure of efficiency that you are using*** (e.g. remember to include the cost of all operations within a cycle).

What are your conclusions about the best choices for the resolution of the coarsest grid, and for the numbers $\nu_1$, $\nu_2$ and $\nu_3$? Next, choose suitable values of $N_1$, $\nu_1$, $\nu_2$ and $\nu_3$, and investigate the dependence of the rate of convergence on $N_K$. Finally, discuss the improvement in efficiency of multigrid over the simple Gauss-Seidel iteration in question 1 when the aim is convergence to an accuracy consistent with the truncation error of the discretisation (3) of equation (1).

## References

[1] Briggs, William L., Henson, Van Emden, McCormick, Steve F. (2000) *A Multigrid Tutorial*, SIAM (ISBN 0-89871-462-1).

# 2 Waves

## 2.2 Dispersion (7 units)

*This project assumes only the elementary properties of dispersive waves, covered in the Part II Waves course (but the relevant material can be found in the references).*

## 1 Introduction

This project illustrates the way in which a disturbance in a 'dispersive-wave' system can change shape as it travels. In order to fix ideas we shall consider one-dimensional waves, depending on a single spatial coordinate $x$ and time $t$, which are modelled by a system of linear constant-coefficient partial differential equations that is (i) second-order in time and (ii) time-reversible. Such a system has single-Fourier-mode (aka 'plane-harmonic-wave') solutions proportional to

$$\mathrm{e}^{ikx \mp i\omega(k)t} \tag{1}$$

for any real '[angular] wavenumber' $k$, where the [angular] frequency' $\omega$ is real and related to $k$ by a system-dependent 'dispersion relation'. The waves are 'dispersive' if $\omega$ is not proportional to $k$ (and so 'group velocity' $d\omega/dk$ and 'phase velocity' $\omega/k$ vary with $k$, and are unequal). As an example, one-dimensional 'capillary-gravity' waves on the free surface of incompressible fluid of uniform depth $h$ have dispersion relation

$$\omega^2 = \left(gk + \rho^{-1}\gamma k^3\right)\tanh(kh) \tag{2}$$

where $g$ is gravitational acceleration, $\rho$ the fluid density and $\gamma$ the coefficient of surface tension.

If the disturbance is described by a function $F(x,t)$, representing say the [non-dimensionalised] vertical displacement of the fluid surface, the general solution for $F$ will be a superposition of all Fourier modes of the form (1):

$$F(x,t) = \int_{-\infty}^{\infty} \left(a_+(k)\,\mathrm{e}^{ikx - i\omega(k)t} + a_-(k)\,\mathrm{e}^{ikx + i\omega(k)t}\right) dk\ , \tag{3}$$

where the amplitudes $a_+(k)$ and $a_-(k)$ are fixed by the initial conditions. For simplicity we shall take these to be

$$F(x,0) = \exp\left(-\frac{x^2}{\sigma^2}\right)\cos(k_0 x) \quad \text{and} \quad \frac{\partial F}{\partial t}(x,0) = 0\ . \tag{4}$$

where $\sigma$ and $k_0$ are constants.

**Question 1** Show that (3) then becomes

$$F(x,t) = \int_{-\infty}^{\infty} A(k)\cos[\omega(k)t]\,\mathrm{e}^{ikx}\,dk\ , \tag{5}$$

where $A(k)$ is to be determined.

In order to plot the solution some method is needed for evaluating the Fourier integral (5).

## 2 The Discrete Fourier Transform

The Fourier Transform $\hat{G}(k)$ of a function $G(x)$ may be defined by[*]

$$\hat{G}(k) = \frac{1}{2\pi} \int_{-\infty}^{\infty} G(x)\, \mathrm{e}^{-ikx}\, dx \ , \tag{6}$$

with inverse

$$G(x) = \int_{-\infty}^{\infty} \hat{G}(k)\, \mathrm{e}^{ikx} dk \ . \tag{7}$$

The integral (7) can be approximated by the discretisation

$$\Delta k \sum_{n=-N/2+1}^{N/2} \hat{G}_n \mathrm{e}^{in\Delta kx} \ , \quad \hat{G}_n = \hat{G}(n\Delta k) \tag{8}$$

provided that $\Delta k$ is small enough to resolve the variation of the integrand with $k$, and that $\hat{G}(k)$ is only significant for $|k| < \frac{1}{2}N\Delta k$. With $\Delta k = 2\pi/L$ and $\Delta x = L/N$, this approximates $G(m\Delta x)$ by

$$g_m \equiv \frac{2\pi}{L} \sum_{n=-N/2+1}^{N/2} \hat{G}_n \mathrm{e}^{2\pi imn/N} \quad \text{for } -N/2+1 \leqslant m \leqslant N/2 \tag{9}$$

[note that $g_m$ is periodic in $m$ with period $N$, and cannot be expected to give a useful approximation to $G(m\Delta x)$ for $|m| > N/2$, i.e. for $|x| > L/2$, since the $\mathrm{e}^{ikx}$-factor in the integrand would be chronically under-resolved].

(9) is the *exact* inverse of

$$\hat{G}_n = \frac{L}{2\pi N} \sum_{m=-N/2+1}^{N/2} g_m \mathrm{e}^{-2\pi imn/N} \text{ for } -N/2+1 \leqslant n \leqslant N/2 \ ; \tag{10}$$

the right-hand side is a discretisation of the integral (6) with $k = n\Delta k$, but that will not be required in this project. The so-called *Discrete Fourier Transform* (10) and its inverse (9) converge to the Fourier Transform (6) and its inverse (7) in the double limit $L \to \infty$, $N/L \to \infty$.

## 3 The Fast Fourier Transform

The Fast Fourier Transform (FFT) technique is a quick method of evaluating sums of the form

$$\lambda_m = \sum_{n=0}^{N-1} \mu_n \left(\zeta_N\right)^{smn} \ , \quad m = 0, \ldots, N-1 \ , \quad \zeta_N = \mathrm{e}^{2\pi i/N} \ , \quad s = \pm 1 \tag{11}$$

where the $\mu_n$ are a known sequence, and $N$ is a product of small primes, preferably a power of 2. A brief outline of the FFT is given in the appendix for reference, but it is not necessary to understand the details of the algorithm in order to complete the project – indeed, you are strongly advised to use a black-box FFT procedure such as Matlab's `fft/ifft`. Note that since

$$\left(\zeta_N\right)^{smn} = \left(\zeta_N\right)^{s(m\pm N)n} = \left(\zeta_N\right)^{sm(n\pm N)} \tag{12}$$

[*]There are various conventions regarding the sign of the exponent and the placement of the $2\pi$-factor.

the sums in (9) and (10) can be converted to the form (11) by repositioning part of the series (and Matlab arrays are indexed from 1 to $N$ rather than 0 to $N-1$). Similar considerations also apply to available routines in other languages, and you may also need to take special care regarding sign conventions and scaling.

**Programming Task**: Write a program to compute a DFT approximation to $F(x,t)$.

## 4 No Dispersion

**Question 2**

In the limit of 'shallow water' ($|k|h \ll 1 \Rightarrow \tanh(kh) \approx kh$) and negligible surface tension ($\rho^{-1}\gamma|k|^3 \ll g|k|$), the dispersion relation (2) can be approximated by the 'dispersionless'

$$\omega^2 = c_0^2 k^2 \tag{13}$$

with $c_0 = \sqrt{gh}$. The integral (5) can then be evaluated analytically.

Use this to test the program for $t$ up to 10 s, taking $\sigma = 0.5\,\mathrm{m}$, $k_0 = 0\,\mathrm{m}^{-1}$ and $c_0 = 1\,\mathrm{m\,s}^{-1}$ [so $h \approx 0.1\,\mathrm{m}$ if $g = 9.81\,\mathrm{m\,s}^{-2}$]. Choose appropriate values for the parameters $L$ and $N$ so that your plots are correct to 'graphical accuracy'; present evidence of this accuracy in your write-up. Comment on your results [e.g. on the appropriateness of the 'shallow-water' approximation for these parameter values].

## 5 Gravity Waves

The 'deep-water' ($|k|h \gg 1 \Rightarrow \tanh(kh) \approx \mathrm{sign}(k)$) and negligible-surface-tension limit of the dispersion relation (2) is

$$\omega^2 = g|k|. \tag{14}$$

**Question 3** Take $g = 9.81\,\mathrm{m\,s}^{-2}$ and in the first instance use initial condition (4) with $\sigma = 1$ m, $k_0 = 0\,\mathrm{m}^{-1}$.

- For $t = 2\,\mathrm{s}$ investigate the effects of changing the values of $L$ and $N$ (maybe start with $L = 32$ m and $N = 32$). Report the results of this investigation in your write-up, especially with regard to the errors in the solution, using both numerical values and plots.

  *Note*: The behaviour of the solution for large $|x|$ can be understood asymptotically by performing integrations-by-parts on (5), but is not of primary interest here [and does not apply for waves on fluid of finite depth] the main concern should be locating the crests and troughs with reasonable accuracy.

- Display graphical results to illustrate how the solution for this initial condition evolves for $t$ up to at least 6 s, giving justification for your choices of $L$ and $N$. Do likewise for the initial condition (4) with $\sigma = 6$ m and $k_0 = 1\,\mathrm{m}^{-1}$, for $t$ up to at least 20 s.

  Comment on the solutions, particularly in the light of group and phase velocity.

## 6 Capillary Waves

Consider now the dispersion relation for 'deep-water' surface waves when surface-tension effects dominate over gravitational:

$$\omega^2 = \rho^{-1}\gamma|k|^3 \ . \tag{15}$$

**Question 4** Perform similar calculations to those in Q3 for water with $\rho = 10^3\,\mathrm{kg\,m^{-3}}$ and $\gamma = 0.074\,\mathrm{kg\,s^{-2}}$, using the initial condition (4) with $\sigma = 0.002$ m, $k_0 = 0\,\mathrm{m}^{-1}$ and with $\sigma = 0.005$ m, $k_0 = 1250\,\mathrm{m}^{-1}$, for $t$ up to at least 0.1 s. Compare and contrast your results with those in Q3. You will want to use different value(s) for $L$ (and maybe $N$): can the concept of group velocity help in choosing a suitable $L$ for given time?

How much difference would it make to these results if the exact 'deep-water' dispersion relation

$$\omega^2 = g|k| + \rho^{-1}\gamma|k|^3 \tag{16}$$

were used, with $g = 9.81\mathrm{m\,s^{-2}}$?

## References

Billingham, J. & King, A. C., *Wave Motion: Theory and Applications*, CUP.

Lighthill, M. J., *Waves in Fluids*, CUP.

Whitham, G. B., *Linear and Nonlinear Waves*, Wiley.

## Appendix: The Fast Fourier Transform

For simplicity restrict to the optimal case $N = 2^M$. Then the DFT (11) can be split into its even and odd terms

$$\lambda_m = \underbrace{\sum_{n'=0}^{N/2-1} \mu_{2n'} \left(\zeta_{N/2}\right)^{smn'}}_{\lambda_m^E} + \left(\zeta_N\right)^{sm} \underbrace{\sum_{n'=0}^{N/2-1} \mu_{2n'+1} \left(\zeta_{N/2}\right)^{smn'}}_{\lambda_m^O} \tag{17}$$

and since $\lambda_m^E$ and $\lambda_m^O$ are periodic in $m$ with period $N/2$, and $\left(\zeta_N\right)^{sN/2} = -1$,

$$\lambda_{m+N/2} = \lambda_m^E - \left(\zeta_N\right)^{sm} \lambda_m^O \ . \tag{18}$$

Thus if the half-length transforms $\lambda_m^E$, $\lambda_m^O$ are known for $0 \leqslant m \leqslant N/2-1$, the $\lambda_m$ for $0 \leqslant m \leqslant N-1$ can be evaluated at a 'cost' of computing $\frac{1}{2}N$ products [additions require relatively little computational effort]. The process can be performed recursively $M$ times, giving a decomposition in terms of $N$ transforms of length one – which are just the original $\mu_n$ $(0 \leqslant n \leqslant N-1)$.

To execute an FFT, start with these length-one transforms; at the $k$-th stage, $k = 1, 2, \ldots, M$, assemble $2^{M-k}$ transform of length $2^k$ from transforms of length $2^{k-1}$, at a 'cost' of $2^{M-1} = \frac{1}{2}N$ products. The complete DFT is formed after $M$ stages, i.e. after $\frac{1}{2}N\log_2 N$ products, as opposed to $N^2$ products in naive matrix multiplication – so for $N = 1024 = 2^{10}$ the 'cost' is $5 \times 10^3$ products as opposed to $10^6$ products!

For more details, see for example Press *et al.*, *Numerical Recipes*, CUP.

# 2 Waves

## 2.11 Fisher's Equation for Population Dispersal Problems (9 units)

*This project is essentially self-contained, and does not directly rely on any Part II lecture course. However, attendance at a Part II Numerical Analysis course may be of some help, as may attendance at the Part II course, Mathematical Biology.*

### Problem Formulation

An equation commonly encountered in population genetics is the one-dimensional diffusion equation

$$\frac{\partial \hat{\rho}}{\partial \hat{t}} = -\frac{\partial j}{\partial \hat{x}} + F(\hat{\rho}). \tag{1}$$

Here, $\hat{x}$ denotes the spatial position, $\hat{t}$ the time, $\hat{\rho}(\hat{x},\hat{t})$ the population density, $j$ the population flux, and $F(\hat{\rho})$ is a local source term that describes the net rate of growth in the population density.

A typical model for local population growth is given by the Pearl-Verhulst law

$$F(\hat{\rho}) = \begin{cases} \gamma\hat{\rho}(1-\hat{\rho}/\hat{\rho}_s) & 0 < \hat{\rho} < \hat{\rho}_s; \\ 0 & \hat{\rho} \leqslant 0 \quad \hat{\rho} \geqslant \hat{\rho}_s. \end{cases} \tag{2}$$

This describes how a homogeneous population would grow, initially in an exponential manner, until the population saturated at some density $\hat{\rho}_s$.

The flux $j$ is the source of the diffusive behaviour and is given by,

$$j = -D\frac{\partial \hat{\rho}}{\partial \hat{x}}. \tag{3}$$

If it is assumed that dispersal is due to random motion of individuals, then the diffusion coefficient $D$ is constant and Fisher's equation is obtained. However, as a remedy to overcrowding, dispersal would be much more effective if the diffusion coefficient were population density-dependent. In fact this has been observed in populations of small animals. Here we consider the case $D = D_0\hat{\rho}$. With suitable non-dimensionalisation, we obtain the modified Fisher equation,

$$\frac{\partial \rho}{\partial t} = \frac{\partial}{\partial x}\left(\rho\frac{\partial \rho}{\partial x}\right) + \rho(1-\rho). \tag{4}$$

A similar equation also arises in combustion dynamics.

Travelling wave solutions to this equation are the subject of project 2.11($a$). A situation of more practical interest is when the population density is known at some initial time, and the subsequent evolution of the population is required. In projects 2.11($b$) and 2.11($c$) the expansion of a population which is initially limited to a finite spatial range is considered. Thus solutions to (4) are required subject to the following boundary conditions:

$$\begin{aligned} \rho(x,0) &= \begin{cases} \rho_0(x), & 0 \leqslant x \leqslant 1, \\ 0, & x > 1, \end{cases} \\ \frac{\partial \rho}{\partial x}(0,t) &= 0, \qquad t > 0, \\ \rho(x,t) &\to 0 \qquad \text{as} \quad x \to \infty. \end{aligned} \tag{5}$$

The form of the initial data leads us to consider a solution which is piecewise continuous with a single jump across $x = s(t)$, where conditions given by conservation laws must be satisfied. The initial boundary-value problem can now be reformulated as follows:

$$
\begin{array}{ll}
0 \leqslant x \leqslant s(t): & \\
& \rho_t = (\rho\rho_x)_x + \rho(1-\rho), \\
& \rho(x,0) = \rho_0(x), \\
& \rho_x(0,t) = 0, \\
& \rho(s(t),t) = 0, \quad \rho_x(s(t),t) = -\dot{s}(t). \\
s(t) < x: & \\
& \rho(x,t) \equiv 0
\end{array}
\tag{6}
$$

We refer to $x = s(t)$ as the population front. From the initial population distribution (5) we see that $s(0) = 1$.

In project 2.11($b$), solutions of (4) are obtained for a particular initial distribution, and the behaviour as $t \to \infty$ is examined. In project 2.11($c$), the code developed in project 2.11($b$) is used to examine the propagation of the population front.

## Project 2.11(a): Travelling Wave Solutions

Here we consider solutions of (4) corresponding to steady expansion of a saturated population.

**Question 1** By writing $\rho(x,t) = \phi(\xi)$, where $\xi = x - ct - x_0$, show that $\phi$ is governed by the nonlinear ODE,

$$\phi\phi'' + (\phi')^2 + c\phi' + \phi(1-\phi) = 0 \tag{7}$$

This is to be solved subject to the boundary conditions $\phi \to 1$ as $\xi \to -\infty$, $\phi \to 0$ as $\xi \to \infty$. General analytic solutions to (7) are not available and hence numerical solutions are required. Such solutions could be obtained by shooting, but here it is preferable to consider the asymptotic form of $\phi$ as $\xi \to -\infty$.

**Question 2** By linearising (7) about $\phi = 1$, show that as $\xi \to -\infty$,

$$\phi \sim 1 - Ae^{\lambda\xi} \tag{8}$$

where $A$ is an arbitrary constant, due to the translational invariance of (7), and $\lambda$ is to be determined as a function of $c$. This then provides suitable initial conditions for a forward integration in $\xi$, *ie.*,

$$\phi(\xi_0) = 1 - \delta, \qquad \phi'(\xi_0) = -\lambda\delta, \qquad \delta \ll 1, \tag{9}$$

for arbitrary $\xi_0$.

**Question 3** Obtain solutions for $c = 2, 1.5, 1$ and $0.8$ using a suitable integration method. These travelling wave solutions should be plotted on the same graph on axes with the origin chosen such that $\phi(\xi = 0) = \frac{1}{2}$ (to within graphical accuracy).
Investigate the change in the wave profile as the wave speed $c$ is decreased still further.

**Question 4** Show that

$$\phi(\xi) = \begin{cases} 1 - e^{(\xi-\xi_1)/\sqrt{2}}, & -\infty < \xi < \xi_1; \\ 0 & \xi_1 \leqslant \xi. \end{cases} \tag{10}$$

is an exact solution for a particular value of $c$ to be determined. Comment on this solution, and plot the waveform, with $\xi_1$ chosen so that $\phi(\xi = 0) = \frac{1}{2}$, as before.

## Project 2.11(b): Large-Time Limit for the Initial Value Problem

Travelling wave solutions often give clues to the general behaviour of solutions of a nonlinear wave equation. However, a more commonly encountered problem is when the population density is known at some initial time, and the subsequent evolution of the population is required. In this exercise we obtain solutions to (6). For numerical efficiency, renormalise the domain $[0, s(t)]$, to $[0, 1]$, by introducing a new spatial coordinate $y = x/s(t)$.

**Question 5** Show that the evolution of $\rho(y, t)$ is given by

$$\frac{\partial \rho}{\partial t} = \frac{1}{2s^2}\frac{\partial^2(\rho^2)}{\partial y^2} + \frac{\dot{s}y}{s}\frac{\partial \rho}{\partial y} + \rho(1-\rho) \tag{11}$$

$$\rho_y(0,t) = 0, \qquad \rho(1,t) = 0, \qquad \rho_y(1,t) = -s\dot{s}. \tag{12}$$

The equation (11) is to be solved subject to the boundary conditions (12), and initial conditions

$$\rho(y, 0) = \rho_0(y). \tag{13}$$

The final condition in (12) then determines the motion of the population front, with $s(0) = 1$.

Many methods exist for the numerical solution of parabolic equations. Here we consider a very simple finite-difference method, where spatial derivatives are expressed using centred differences and the solution is advanced in time using forward Euler. Writing $t_j = j(\Delta t)$, $y_n = n/N$, $(n = 0, 1, \ldots, N)$, and using the notation $\rho_{j,n} \equiv \rho(t_j, y_n)$, $s_j \equiv s(t_j)$ we discretise (11) in the form,

$$\begin{aligned}
\frac{\rho_{j+1,n} - \rho_{j,n}}{\Delta t} &= \frac{1}{2s_j^2}\frac{\rho_{j,n+1}^2 - 2\rho_{j,n}^2 + \rho_{j,n-1}^2}{(\Delta y)^2} + \frac{\dot{s}_j y_n}{s_j}\frac{\rho_{j,n+1} - \rho_{j,n-1}}{2(\Delta y)} + \rho_{j,n}(1-\rho_{j,n}), \\
&\qquad\qquad n = 1, 2, \ldots, N-1 \\
\rho_{j,0} &= \rho_{j,1}, \\
\rho_{j,N} &= 0, \\
\frac{s_{j+1} - s_j}{\Delta t} &= \dot{s}_j = -s_j^{-1}\frac{\rho_{j,N-2} - 4\rho_{j,N-1}}{2\Delta y},
\end{aligned}$$

where $\Delta y = 1/N$. The expression for $\dot{s}_j$ is obtained by using the final condition in (12) with a three-point backward difference expression for $\rho_y(y = 1)$.

There are several more sophisticated numerical methods of solving this system of equations, but the method described is very simple to implement and proves sufficient for the current purposes. The main drawback is that $\Delta t$ must be chosen very small to ensure numerical stability.

**Question 6** Write your own program to solve (11) using the discretisation given above. Obtain solutions for initial distribution $\rho_0(x) = 0.3e^x(1-x)$. Start with $N = 100$ and $\Delta t = 0.0001$, but confirm that your code produces solutions that are independent of mesh-size. Plot the solution as a function of the original spatial variable $x$ at $t = 0$, 2, 4, 6, 8 and 10. Also plot the velocity of the wave front, $\dot{s}(t)$, as a function of time. Compare the large-time wave profile with the travelling wave solutions obtained in project 2.11$(a)$.

## Project 2.11(c): Motion of the Wave Front

Using the same program written for project 2.11($b$), we now investigate the early evolution of the wave front for different classes of initial population distribution.

Consider three different initial profiles:

(i) $\rho_0(x) = A_1 e^{x}(1-x)$;

(ii) $\rho_0(x) = A_2 e^{2x}(1-x)^2$;

(iii) $\rho_0(x) = A_3 e^{3x}(1-x)^3$;

where $A_i$ are numerical constants characterising the total initial population.

> **Question 7** Using the same mesh-size as above, obtain solutions for $0 < t \leqslant 0.5$, for initial distribution (i). Consider various values of the coefficient $A_1$, in the range $0.1 \leqslant A_1 \leqslant 0.9$. For the larger values of $A_1$ it may be necessary to reduce the time step-size. Do not include plots of $\rho(x,t)$ in your report, but concentrate on the motion of the wave front. Write down a relationship between the initial velocity of the wave front and the initial profile and show that this is in agreement with your numerical results.

> **Question 8** Calculate solutions for initial distribution (ii) with $A_2 = 0.2$ for $0 < t \leqslant 1$. As before plot $\dot{s}(t)$ as a function of time. Repeat these calculations with the spatial mesh-size reduced to $\Delta y = 0.002$ and then $\Delta y = 0.001$, adjusting $\Delta t$ as necessary. Describe the movement of the wave front. Repeat these calculations with $A_2 = 0.05$, for $0 < t \leqslant 0.75$.

Analysis suggests that for some classes of initial distributions, the population front is fixed until a certain waiting time $t_w$ has elapsed, after which the population expands.

> **Question 9** For initial distributions which are locally quadratic in the vicinity of the wavefront, it can be shown that the waiting time is given by
>
> $$t_w = \log\left(1 + \frac{1}{6g_2}\right) \qquad (14)$$
>
> where $\rho_0(x) \sim g_2\,(1-x)^2$, as $x \to 1$. Are the numerical results you have obtained in broad agreement with this result? Discuss why such a phenomenon may occur in the evolution of a population.

> **Question 10** Calculate the motion of the population front for initial distribution (iii) with $A_3 = 0.2$. As with case (ii), reduce the mesh-size. Compare your results with the results of (ii).

## References

A background to the biological models underlying these equations can be found in *Some exact solutions to a nonlinear diffusion problem in population genetics and combustion* by Newman (*J. Theoretical Biology* (1980) **85**, 325–334).

An in-depth analysis of equations of this form is presented in *The effects of variable diffusivity on the development of travelling waves in a class of reaction-diffusion equations* by King & Needham (*Phil. Trans. Roy. Soc. Lond.* A (1994) **348**, 229–260). This contains derivation of the results for waiting times, but reference to this paper is not necessary for the purposes of this project.

# 3 Fluid and Solid Mechanics

## 3.6 Particle Drift in a Periodic Flow Field (4 units)

*This project builds on material in the Part IB Fluid Dynamics course.*

A one-dimensional periodic flow in a fluid has velocity $u$ in the $x$-direction only, given by

$$u = \alpha \cos k(x - ct). \tag{1}$$

A material fluid element subject to this motion will have trajectory $X(t)$ satisfying

$$\frac{dX}{dt} = \alpha \cos k(X(t) - ct). \tag{2}$$

**Question 1** Explain why, without loss of generality, distance and time units may be chosen so that $k = 2\pi$ and $c = 1$, giving

$$\frac{dX}{dt} = a \cos 2\pi(X(t) - t). \tag{3}$$

How is $a$ related to $\alpha$?

**Question 2** Solve (3) numerically for a representative set of values of $a$, taking $X(0) = 0$. Describe your results qualitatively, and plot the solutions against time. You can use your own ODE integrator, or alternatively one such as the MATLAB function `ode45`. In either case you should justify the accuracy of your results (for example, by considering results produced with different step-sizes or tolerances). What if $X(0) \neq 0$?

**Question 3** Verify from your numerical results that for $|a|$ sufficiently small, there is a time-averaged mean 'drift' velocity of $\frac{1}{2}a^2$. Include details of your method.

**Question 4** Give a *physical* interpretation of the interaction between the flow and the material element. Do not confine your answer only to small $|a|$.

*Hint:* You may wish to consider a graph of $\frac{dX}{dt}$ against $X$.

**Question 5** Analyse mathematically the above system, using any approach you see fit, e.g. in the case of question 3 you might seek an approximate solution for small $|a|$.

# 3 Fluid and Solid Mechanics

## 3.9 Viscous Flow in a Collapsible Channel (9 units)

*This project requires a knowledge of lubrication theory for a viscous fluid, as taught in the Part II course Fluid Dynamics II and described in [1].*

## 1 Introduction

The tubes that carry fluid around the body (such as veins, arteries, lung airways, the urethra, etc.) have deformable walls. The shape of such a tube is strongly coupled to the flow within it through the internal pressure distribution. This nonlinear flow-structure interaction imparts to such systems unusual but biologically significant properties, notably "flow limitation" (so that airway flexibility limits the rate at which you can expel air from your lungs, for example). To explore such interactions, one can consider a simple model system in which an incompressible fluid flows steadily through a two-dimensional channel, one wall of which is formed by a membrane under longitudinal tension. Assuming that the channel is long and thin, and that the fluid is sufficiently viscous, lubrication theory can be used to describe the flow.

Suppose the channel lies in $0 \leqslant y \leqslant h(x)$, $0 \leqslant x \leqslant L$, where $L \gg h$. Applying no-slip and no-penetration conditions along the rigid wall $y = 0$ and the membrane $y = h$, the relationship between the steady, uniform flux $q$ of fluid along the channel and the local pressure gradient $p_x$ is approximately $q = -h^3 p_x/(12\mu)$, where $\mu$ is the fluid's viscosity, assumed constant. The fluid pressure distribution $p(x)$ is controlled by the shape of the channel wall according to $p = -Th_{xx}$, where $T$ is the tension in the membrane, assumed constant; the pressure outside the membrane is taken to be zero. We assume that the membrane is fixed at either end, so that $h(0) = h(L) = h_0$, for some constant $h_0$. The flow is controlled by the upstream and downstream pressures $p(0) = p_u$ and $p(L) = p_d$, and characterised by the relationship between the flux $q$ and the pressure drop along the channel, $p_u - p_d$, holding either $p_u$ or $p_d$ constant.

The problem can be simplified by nondimensionalisation. Let

$$h(x) = h_0 H(X)\,, \quad x = LX\,, \quad \text{and} \quad p(x) = p_0 P(X)\,,$$

where $p_0 = Th_0/L^2$. This yields nondimensional parameters $Q = 12\mu L^3 q/(Th_0^4)$, $P_u = p_u/p_0$, $P_d = p_d/p_0$ and governing equations

$$Q = -H^3 P_X, \qquad P = -H_{XX} \qquad (0 \leqslant X \leqslant 1) \tag{1}$$

subject to

$$H(0) = 1, \quad H(1) = 1, \quad P(0) = P_u, \quad P(1) = P_d. \tag{2}$$

We seek graphs of $\Delta P = P_u - P_d > 0$ as a function of $Q$, for fixed values of $P_u$ or $P_d$. (So only 3 of the 4 boundary conditions in (2) are relevant.)

This is a two-point, third-order, boundary-value problem. It can be solved by two different methods: shooting, which is relatively easy to program but which cannot normally be extended to problems in higher dimensions; and a direct finite-difference method, which is more complicated to set up but adaptable to more complex situations. The relatively straightforward problem given by (1) and (2) can be used to explore the relative merits of each method; both methods can be used to explore the fluid mechanics of collapsible channel flow.

## 2 The shooting method

Use a Runge-Kutta routine, or equivalent (e.g. the MATLAB function `ode45`), to integrate (1) from $X = 1$ to $X = 0$ (say), by fixing $Q$ and $P_d$, setting $H(1) = 1$, $H'(1) = \beta$, $H''(1) = -P_d$ and then varying $\beta$ until $H(0) = 1$. If you know that a solution exists for $\beta_1 < \beta < \beta_2$, say, a root-finding routine will be useful in quickly homing in to the required solution. Note that you should not use a boundary-value problem solver such as the MATLAB functions `bvp4c` or `bvp5c`, or equivalent.

You should check that your predicted channel shapes and pressure distributions are of sufficient accuracy by varying any tolerance you have specified on the step-length, etc. You will also need to compute solutions with $P_u$ fixed, shooting from $X = 0$ to $X = 1$ (see Question 3 below).

## 3 The direct finite-difference method

Writing (1) in the form $H_{XXX}H^3 = Q$, discretise this equation and the boundary conditions with second-order accurate finite differences on a uniform $N$-node grid with grid-spacing $\Delta = 1/(N-1)$ and grid points $X_j = (j-1)\Delta$ $(j = 1, \ldots, N)$. Use forward (or backward) differences for the discretisation of the second derivative in the upstream (or downstream) pressure boundary condition. In most of the interior domain you can use a central difference expression for the discretisation of $H_{XXX}$, but near one of the boundaries of the domain you will have to use a non-central difference expression; suitable difference formulae are given in Appendix A.

The three discretised boundary conditions and the discretised ODE, written at $(N-3)$ interior gridpoints $X_j$,$(j = 3, \ldots, N-1)$, provide a total of $N$ non-linear algebraic equations $\mathcal{F}_i(H_j) = 0$ $(i, j = 1, \ldots, N)$ for the discrete membrane heights $H_j = H(X_j)$ . Solve this set of equations with a Newton-Raphson method (e.g. [2]).

The Newton-Raphson method requires the Jacobian matrix of the non-linear equations

$$J_{ij} = \partial \mathcal{F}_i / \partial H_j \, ,$$

which can be determined by differentiating the discretised equations. At each stage of the iteration, the method requires solution of a set of linear equations. You might find the MATLAB function `spdiags` useful here (see also `help sparfun` in MATLAB).

You should include a brief summary of the equations needed for this method in your write-up.

## 4 Continuation techniques

The Newton-Raphson method usually requires a "good" initial guess in order to converge to a solution, so to generate solutions corresponding to strongly deformed channels a continuation technique should be used. (The shooting method can also benefit from this approach, but it is not usually necessary.) Start the computation with parameter values corresponding to a known solution (e.g. a slightly deformed channel with $P_d = 0$, $Q \ll 1$) and use the undeformed channel $(H = 1)$ as an initial guess. Having found this solution, slowly increment the parameters $Q$ and $P_d$ to construct solutions with the channel highly deformed.

## 5 Questions

Throughout this project you should comment on the physical interpretation of your computed results as well as their mathematical or numerical features.

**Question 1** Compute some static wall shapes when there is no flux through the channel, with the fluid pressure both positive and negative. These shapes can be determined analytically: compare your predictions using both numerical methods above with the exact analytical results.

By considering analytically the case when $|H - 1| \ll 1$, or otherwise, show that for $Q > 0$ there are three possible types of solution, depending on the values of $P_u$ or $P_d$: those with the channel dilated ($H > 1$ in $0 < X < 1$); those with the channel collapsed ($0 < H < 1$ in $0 < X < L$); and those with both dilation and collapse ($H > 1$ for $0 < X < X_b$ and $H < 1$ for $X_b < X < 1$, for some $X_b$). Show that both numerical methods predict the same channel shapes and pressure distributions for typical values of $Q$, $P_u$ and $P_d$. Comment on the qualitative differences between $Q = 0$ and $Q > 0$.

**Question 2** Using either method, produce graphs of $\Delta P$ as a function of $Q$ for fixed values of the downstream transmural pressure $P_d$, showing examples with $P_d$ both positive and negative (consider, say, $-3 \leqslant P_d \leqslant 3$, $0 \leqslant Q \leqslant 6$). Show that the slope of the graph of $\Delta P$ versus $Q$ falls as $Q$ increases, and show how the channel shape evolves as this happens. Explain this behaviour in *physical* terms.

**Question 3** By shooting from $X = 0$ to $X = 1$, produce graphs of $\Delta P$ as a function of $Q$ for fixed values of the upstream transmural pressure $P_u$, both positive and negative, and again describe the evolution of the channel shape. This case requires slightly more care than Question 2, as you may find that the solution is not unique. Show that for each value of $P_u$ there is a maximum possible flow rate through the channel. Obtain the same graphs using the direct finite-difference method; explain any techniques that you may need to introduce in order to obtain converged solutions. Explain the physical mechanism by which the flux may fall as the pressure drop across the channel is increased.

## References

[1] Acheson, D. J. *Elementary Fluid Mechanics*, Oxford University Press, 1990.

[2] Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P. *Numerical Recipes* (available in various editions for different languages). Cambridge University Press, 1992.

## A Finite difference formulae

Here is a collection of second-order accurate finite difference expressions for the second and third derivatives of the channel height $[H_j = H((j-1)\Delta)]$:

$$H''(X_j) = \frac{H_{j-1} - 2H_j + H_{j+1}}{\Delta^2}$$

$$H''(X_j) = \frac{2H_j - 5H_{j+1} + 4H_{j+2} - H_{j+3}}{\Delta^2}$$

$$H''(X_j) = \frac{2H_j - 5H_{j-1} + 4H_{j-2} - H_{j-3}}{\Delta^2}$$

$$H'''(X_j) = \frac{H_{j+2} - 2H_{j+1} + 2H_{j-1} - H_{j-2}}{2\Delta^3}$$

$$H'''(X_j) = \frac{3H_{j+1} - 10H_j + 12H_{j-1} - 6H_{j-2} + H_{j-3}}{2\Delta^3}$$

# 3 Fluid and Solid Mechanics

## 3.10 Smoke Rings (8 units)

*This project discusses a simple model of the motion of smoke rings. Knowledge of Part IB Fluid Dynamics is required, while knowledge of the Part II course Classical Dynamics will help with Question 2. The articles by Acheson [1] and Tophøjj & Aref [5], and the book by Saffman [4] may be found helpful.*

A smoke ring is a vortex tube wrapped around into a closed circle (a *vortex ring*), which propagates normal to the plane of the circle under its self-induced velocity field. The politically incorrect method of generating them involves the inhalation of noxious substances; a more socially acceptable method involves a volcano [6]. We will, throughout this project, neglect various effects and crudely model a three-dimensional axisymmetric vortex ring of diameter $a$ and strength $\kappa$ by a pair of point vortices in two-dimensional fluid of strengths $\kappa$ and $-\kappa$, a distance $a$ apart.

## 1 2D vortex dynamics: Theory

**Question 1** Show that the equations of motion of a set of point vortices of strengths $\kappa_i$ at positions $(x_i(t), y_i(t))$, in a two-dimensional inviscid fluid which is otherwise at rest, are

$$\frac{\mathrm{d}x_i}{\mathrm{d}t} = -\frac{1}{2\pi}\sum_{j\neq i}\frac{\kappa_j(y_i-y_j)}{r_{ij}^2}\,, \tag{1a}$$

$$\frac{\mathrm{d}y_i}{\mathrm{d}t} = \frac{1}{2\pi}\sum_{j\neq i}\frac{\kappa_j(x_i-x_j)}{r_{ij}^2}\,, \tag{1b}$$

where

$$r_{ij} = ((x_i-x_j)^2+(y_i-y_j)^2)^{1/2}\,. \tag{1c}$$

**Question 2** Show carefully that the equations of motion can be written in the form

$$\frac{\mathrm{d}x_i}{\mathrm{d}t} = \kappa_i^{-1}\frac{\partial H}{\partial y_i}, \qquad \frac{\mathrm{d}y_i}{\mathrm{d}t} = -\kappa_i^{-1}\frac{\partial H}{\partial x_i} \qquad \text{(no summation)}\,, \tag{2}$$

where

$$H = -\frac{1}{4\pi}\sum_{\substack{i,j\\ i\neq j}}\kappa_i\kappa_j\log r_{ij}. \tag{3}$$

$H$ is invariant under space translations and rotations, which implies the existence of three scalar conserved quantities. What are they?

*Hint:* Suppose $H$ has a continuous family of symmetries, i.e.

$$H(X(x,y,\delta), Y(x,y,\delta)) = H(x,y)\,,$$

such that $X(x,y,0)=x$ and $Y(x,y,0)=y$. Observe that

$$0 = \sum_i\left(\frac{\partial H}{\partial x_i}\frac{\partial X_i}{\partial \delta}+\frac{\partial H}{\partial y_i}\frac{\partial Y_i}{\partial \delta}\right)_{\delta=0}.$$

$H$ is also invariant under time translations — what conserved quantity does this give?

**Programming Task:** Write a program to integrate the equations of motion (1a), (1b) and (1c). You should use an adaptive stepsize ODE integrator (such as the MATLAB function `ode45`). You will find it useful to write your code to handle arbitrarily many vortices.

## 2 Simulations of smoke rings

**Question 3** Use your code to investigate the motion of a single "axisymmetric vortex ring" under this model. This problem can also be solved analytically; use the analytic solution to test your code. Demonstrate to what accuracy your code preserves the constants of the motion.

**Question 4** What happens when two smoke rings are fired towards each other on the same axis? Describe the resulting motion, giving clear physical explanations for each behaviour observed. You should start by considering two rings with equal strengths and widths, but should also explain what happens in the general case.

**Question 5** What happens when two smoke rings are fired in the same direction on the same axis? Describe the resulting motion, giving clear physical explanations for each behaviour observed. You should start by considering two rings with equal strengths and widths, but should also explain what happens in the general case. You should not need to integrate to large times, but you should (where relevant) show several cycles of the motion.

**Question 6** We have made a number of modelling assumptions in reducing the full three-dimensional problem to this simple two-dimensional version. Discuss how good they are. Would the behaviour that you have observed in question 5 occur in a real physical system? Are the physical explanations that you gave in questions 4 and 5 for two dimensional flow relevant in the real geometry? What other possible effects have we neglected?

## 3 Symmetries and instabilities

**Question 7** Repeat a typical one of the simulations you did for question 5, but now integrate to large times and show your output. What happens? Is the resulting behaviour physically plausible for a pair of 3D vortex rings? What happens if you adjust the error tolerances of your ODE solver?

**Programming Task:** Produce a program which can only model coaxial smoke rings, but which explicitly enforces the symmetry about the axis. In other words, use the mirror symmetry of the model system to reduce the number of ODEs that you have to solve.

*Note that this is not the best way to handle symmetries and conserved quantities in the numerical solution of ODEs. See Iserles* et. al. *[2] for more information. For this project, however, this method will suffice.*

**Question 8** Use this new program to repeat the simulation you did for question 7. Show your output and comment.

## 4 More smoke rings

**Question 9** Consider three coaxial smoke rings, fired in the same direction. Use your new program to investigate the resulting motion. Give a survey of the different kinds of behaviour you observe, including a selection of your plots (four or so should suffice).

*Note that the parameter space you have to search is rather large. You should think of ways to reduce it.*

## References

[1] Acheson, D. J. 2000, Instability of vortex leapfrogging, *Eur. J. Phys.*, **21**, 269–273. Follow links from `https://iopscience.iop.org/journal/0143-0807` for a downloadable version; at the time of writing `https://iopscience.iop.org/article/10.1088/0143-0807/21/3/310` should take you there directly.

[2] Iserles, A., Munthe-Kaas, H. Z., Nørsett, S. P. & Zanna, A. 2000, Lie-group methods, *Acta Numerica*, **9**, 215–365.

[3] Pullin, D. I. & Saffman, P. G. 1991, Long-time symplectic integration: the example of four-vortex motion, *Proc. Roy. Soc. Lond. A*, **432**, 481–494.

[4] Saffman, P. G. 1992, *Vortex Dynamics*, CUP.

[5] Tophøj, L. & Aref, H. 2013, Instability of vortex pair leapfrogging, *Physics of Fluids*, **25**, 014107; `https://doi.org/10.1063/1.4774333`.

[6] `http://www.swisseduc.ch/stromboli/etna/etna00/etna0002photovideo-en.html`.

# 4 Dynamics

## 4.5 Euler's Equations (8 units)

*This project is self-contained, though material from the Part II(C) course Classical Dynamics is relevant.*

## 1 Introduction

The angular velocity with respect to principal axes of inertia in a rigid body is taken to be

$$\boldsymbol{\omega} = (\omega_1, \omega_2, \omega_3). \tag{1}$$

If the principal moments of inertia are $A$, $B$, $C$ with respect to these axes, then the angular momentum is

$$\mathbf{h} = (A\omega_1, B\omega_2, C\omega_3). \tag{2}$$

These axes are fixed in the body and have angular velocity $\boldsymbol{\omega}$ with respect to an inertial frame instantaneously coincident with the principal axes. The rate of change of angular momentum with respect to such an inertial frame is

$$\frac{d\mathbf{h}}{dt} + \boldsymbol{\omega} \wedge \mathbf{h}. \tag{3}$$

In the case when there is no net moment of external forces acting on the body, the law "rate of change of angular momentum = moment of external forces" gives:

$$\frac{d\mathbf{h}}{dt} + \boldsymbol{\omega} \wedge \mathbf{h} \ = \mathbf{0}. \tag{4}$$

Expanding this equation into components gives:

$$\left.\begin{array}{l} A\frac{d\omega_1}{dt} + (C-B)\omega_2\omega_3 = 0 \\ \\ B\frac{d\omega_2}{dt} + (A-C)\omega_3\omega_1 = 0 \\ \\ C\frac{d\omega_3}{dt} + (B-A)\omega_1\omega_2 = 0 \end{array}\right\} \tag{5}$$

It can be shown analytically that these equations have two first integrals, which say that the energy and the magnitude of the angular momentum remain constant, as follows:

$$\tfrac{1}{2}A\omega_1^2 + \tfrac{1}{2}B\omega_2^2 + \tfrac{1}{2}C\omega_3^2 = E \tag{6}$$

$$A^2\omega_1^2 + B^2\omega_2^2 + C^2\omega_3^2 = H^2 \tag{7}$$

Since the moment of external forces is zero, we also know that the angular momentum vector $\mathbf{h}$ is constant when measured in an inertial frame.

## 2 Project work

### 2.1 Program requirements

Write a program to solve Euler's equations (5) numerically and plot the results. You should use MATLAB's 64-bit (8-byte) double-precision floating-point values or the equivalent in other programming languages. Output from your program should include:

1. Graphs of $\omega_1(t)$, $\omega_2(t)$ and $\omega_3(t)$ against $t$;
2. 3-D phase space plots of $\omega_1$, $\omega_2$ and $\omega_3$. Choose the $OX$ axis for $\omega_1$, etc.

Equations (6) and (7) can be used to check the accuracy of your numerical results by calculating and displaying the values of these expressions at the beginning and end of runs.

The objective of the project is to investigate and classify all possible types of motion. The following questions provide some guidelines for the investigation.

**Question 1** Since $A$, $B$, $C$, $\omega_1(0)$, $\omega_2(0)$ and $\omega_3(0)$ may all in principle take arbitrary values, the parameter space to be explored may seem very large. If $A$, $B$ and $C$ take distinct values, explain how the results from taking

$$A > B > C \tag{8}$$

can be generalised. Briefly explain what happens if any two (or all three) of $A$, $B$, $C$ are equal. For given values of $A/B$ and $C/B$, explain why we may take $B = 1$ without loss of generality.

Show further that choosing $E = 1$ is equivalent to re-scaling the time variable $t$, and give the scaling factor.

## 2.2 Results requirements

From this point on, take $B = 1$ and $E = 1$ and build these values into your program. Your program should allow you to set and change the values of $A/B$ and $C/B$. Assume that $A/B > 1$ and $C/B < 1$, and work with $A = 1.4$, $B = 1$ and $C = 0.7$ unless other values are suggested. You may find it convenient to accept arbitrary input for $\omega_1(0)$, $\omega_2(0)$ and $\omega_3(0)$ and then scale the input values so that $E = 1$.

**Question 2** Use your program to demonstrate that solutions are possible in which the vector $\boldsymbol{\omega}(t)$ rotates around the $OX$ axis with small amplitude deviation from $(\sqrt{2/A}, 0, 0)$ (i.e., $(1, 0, 0)$ before scaling), and that similar stable solutions exist near the $OZ$ axis. Include copies of your results.

**Question 3** Approximate Euler's Equations (5) by linearising them for the cases where (i) $\omega_1 \approx \sqrt{2/A}$ and $\omega_2$, $\omega_3$ are small, and (ii) $\omega_3 \approx \sqrt{2/C}$ and $\omega_1$, $\omega_2$ are small. Describe the analytic solutions. Are your solutions consistent with the results obtained for question 2 above? Give analytic expressions for the period of the motion in each case and compare with your results. (You need not compute the periods; an estimate by eye from your graphs of $\omega$ vs $t$ is sufficient.)

**Question 4** Are your numerical results consistent with equations (6) and (7)? To what extent are further checks on numerical accuracy needed?

**Question 5** What solutions do you obtain if starting conditions are chosen so that $\boldsymbol{\omega}(0)$ lies very close to the $OY$ axis? Describe the motion physically.

**Question 6** Make a plausible case based on your computed results that there exists a solution $\boldsymbol{\omega}(t)$ which begins away from the $OY$ axis but which tends towards the steady (but unstable) solution parallel to the $OY$ axis as $t \to \infty$. What happens if you attempt to simulate such a solution numerically? What value must $H$ take for such a solution?

**Question 7** Using the following scheme, classify all the possible qualitative types of motion of the system assuming $A$, $B$ and $C$ take distinct values. Take the solution discussed in question 6 as a type of its own, and use it to separate the remaining solutions into two types. Describe the range of behaviour observable for each type. Explain clearly how the solution discussed in question 6 divides the solution space into regions (can you find the equations of the boundaries of these regions?) and how solutions behave physically as the boundaries are approached.

Try different values of $A/B$ and $C/B$. How does the choice of these parameters affect your results?

**Question 8** The rigid body is now subjected to slow friction via a retarding couple $-k\boldsymbol{\omega}$, where $k$ is a very small parameter. How does this affect equations (5), (6) and (7)? Alter your program to incorporate the couple and investigate a few types of solutions for the original values of $A$, $B$ and $C$. You may find it useful to consider 3D phase plots of a *suitably normalised* $\boldsymbol{\omega}(t)$, as well as your normal plots.

**Question 9** Is your classification in question 7 still of any use or has it become irrelevant? Is there still a division of the solution space into regions?

# 5 Quantum Mechanics

## 5.2 Electron Scattering (7 units)

*This project relies on knowledge of material from the course* Applications of Quantum Mechanics*.*

## 1 Introduction

Scattering theory can be used to compute what happens when a beam of electrons is fired towards a target object. The electrons collide with the object and are scattered off in new directions. The angular distribution of the outgoing electrons is described by a sum of contributions, in which the simplest term corresponds to an isotropic distribution. If the energy of the incoming electron is not too large, this isotropic contribution dominates the distribution. This situation is called $s$-wave scattering.

## 2 Theory

The electrons are incident on a target that is located at the origin and is decribed by an isotropic interaction potential $U(r)$. The time-independent Schrodinger equation for the outgoing electrons has a general solution

$$\psi(r,\theta) = \sum_{l=0}^{\infty} \frac{\chi_\ell(r)}{r} P_\ell(\cos\theta) \tag{1}$$

where $\psi$ is the wave-function, $P_\ell$ is the Legendre Polynomial of order $\ell$, and $\chi_\ell$ satisfies

$$\frac{d^2\chi_\ell}{dr^2} + \left[k^2 - U(r) - \frac{\ell(\ell+1)}{r^2}\right]\chi_\ell = 0 \tag{2}$$

with boundary condition $\chi_\ell(0) = 0$. The constant $k$ is determined by the energy $E$ of the incoming electrons, with $E$ proportional to $k^2$.

Suppose that the target has radius $r_0$ and that $U(r) = 0$ for $r > r_0$. Then for large $r$ one has asymptotically $\chi_\ell(r) \approx A_\ell \sin(kr - \frac{1}{2}l\pi + \delta_\ell)$, where $\delta_\ell$ is the ($k$-dependent) phase shift of the $\ell^{th}$ partial wave. The total scattering cross-section $\sigma$ is given by

$$\sigma = \frac{4\pi}{k^2}\sum_{\ell=0}^{\infty}(2\ell+1)\sin^2\delta_\ell\,. \tag{3}$$

The $s$-wave contribution to (1) is the term with $\ell = 0$. The associated ($k$-dependent) cross section is therefore $\sigma_0 = (4\pi/k^2)\sin^2\delta_0$. To characterise the low-energy behaviour, a useful quantity is the scattering length $a$, which is defined by $(1/a) = \lim_{k\to 0}[-k\cot\delta_0(k)]$.

For further information on this theory, see for example

1. L.I. Schiff, *Quantum Mechanics*, §19. Scattering by spherically symmetric potentials.
2. S. Gasiorowicz, *Quantum Physics*, Ch. 24. Collision theory.

## 3 Computation

You will write a program to investigate numerically the relation between the phase shift $\delta_0$ and $k$. We consider the potential

$$U(r) = \begin{cases} -U_0\left(1-\frac{r}{\pi}\right)^3 & r \leqslant \pi \\ 0 & r > \pi \end{cases} . \tag{4}$$

**Programming Task:** Numerically solve (2) for $\chi_0(r)$ (i.e. the case $\ell = 0$) taking $\chi_0(0) = 0$ and $\chi_0'(0)$ an arbitrary value. (You should verify that this arbitrary value only affects the normalisation of the wavefunction.)

**Question 1** Explain your choice of numerical method and discuss the accuracy of the solutions you obtain.

**Question 2** Investigate the solutions for $\chi_0$ as you vary $k$ in the range 0 to 5 and $U_0$ from 0 to 10. Discuss the dependence of the wavefunction on $U_0$ for a few different values of $k$. Your report should include a few plots to support your observations, which should include the case $U_0 = 0$.

Note: throughout this project, you should provide graphs that illustrate clearly the similarities and differences between the various cases. Large numbers of graphs are *very unlikely to be effective* in communicating this information.

**Programming Task:** Modify your program to calculate $\delta_0$, using the formula

$$\tan\delta_0 = \frac{\chi_0(r_2)\sin kr_1 - \chi_0(r_1)\sin kr_2}{\chi_0(r_1)\cos kr_2 - \chi_0(r_2)\cos kr_1} \tag{5}$$

where $r_1$ and $r_2$ are two $r$-values in $r > \pi$. (This formula can be derived from the asymptotic expression for $\chi_\ell$ given in the Theory section.) It is conventional that $\delta_0$ depends continuously on $k$ and that $\delta_0 \to 0$ as $k \to \infty$, you should explain how you ensured that your results are consistent with this convention.

**Question 3** Explain how a poor choice of $r_1$ and $r_2$ in Equation (5) can lead to a large error in $\delta_0$. How did you avoid this? Note also that Equation (5) has multiple solutions for $\delta_0$: explain which solution you have taken.

**Question 4** For a few values of $U_0$, compute the phase shift and the cross-section as functions of $k$, with $0 < k < 5$. Plot graphs of these quantities. Take care to resolve the behaviour near $k = 0$.

**Question 5** For the results of Question 4, plot the phase shift $\delta_0$ as a function of $U_0$, for some small value(s) of $k$. Give a physical (quantum-mechanical) interpretation of the observed behaviour.

**Question 6** Still for the results of Question 4, plot the cross section $\sigma$ as a function of $U_0$ (always for small $k$). How are the features of this curve (extrema, etc) related to physical properties of the the scatterer?

**Question 7** Using again the results of Question 4, determine (numerically) the scattering length $a$ from the small-$k$ behaviour of $\delta_0$. Is the result consistent with your results for the cross section $\sigma$ in question 5?

# 5 Quantum Mechanics

## 5.3 Bound State Energies for One-Dimensional Potentials (9 units)

*This project can be done with knowledge of the course* Principles of Quantum Mechanics.

### 1 Introduction

One-dimensional bound states in quantum mechanics are investigated by using a matrix method to estimate eigenvalues of the Schrödinger operator. Several cases are considered and the answers are compared with theory, including the predictions of perturbation theory and variational methods.

### 2 The Schrödinger Equation

The Schrödinger equation in 1D (using units where $\hbar = 1 = m$) is

$$-\frac{1}{2}\frac{d^2\psi_i}{dx^2} + V(x)\psi_i = E_i\psi_i.$$

To obtain approximate solutions to this equation, the real-valued position $x$ is replaced by a discrete set of $2N$ points spaced by $\epsilon$, such that $-N\epsilon \leqslant x < N\epsilon$. The eigenfunction, $\psi(x)$, is replaced by a $2N$-dimensional vector, $\mathbf{e}$, where $\psi(x_n) = e_n$, with $x_n = (n - N)\epsilon$, $0 \leqslant n < 2N$. The Schrödinger equation becomes the matrix eigenvalue equation

$$M\mathbf{e}_i = \epsilon^2 E_i \mathbf{e}_i,$$

where $M$ is a $2N \times 2N$ symmetric tri-diagonal matrix with

| | |
|---|---|
| diagonal entries | $c_n = 1 + \epsilon^2 V(x_n)$ |
| off-diagonal entries | $b_n = -\frac{1}{2} \quad \forall n$ |

### 3 Given's Procedure

Given a symmetric tri-diagonal matrix, $M$, with diagonal entries $c_n$ and off-diagonal entries $b_n$, consider the sequence $q_n$, $(0 \leqslant n < 2N)$, for fixed real parameter $\lambda$:

$$\begin{aligned} q_0 &= c_0 - \lambda \\ q_n &= (c_n - \lambda) - b_n^2/q_{n-1} \qquad n > 0. \end{aligned} \tag{1}$$

Let $s(\lambda)$ be the number of the $q_n$ that are negative. Then **the number of eigenvalues of $M$ whose values are less than $\lambda$ is $s(\lambda)$**. That is, if the eigenvalues are ordered so that $E_i \leqslant E_{i+1}$, then

$$\epsilon^2 E_i < \lambda \quad \text{for } 0 \leqslant i < s(\lambda).$$

$s(\lambda)$ can be computed as a function of $\lambda$ by starting with a sufficiently small value of $\lambda$, incrementing $\lambda$ in small steps and computing the sequence $\{q_n\}$ for each value. When $s(\lambda)$ increases in value from one step to the next, $\lambda$ must have passed through an eigenvalue of $M$ (or through more than one, if $s(\lambda)$ increases by more than one, in which case you should use a smaller stepsize). An accurate value for this eigenvalue can then be determined by bisection before going on to the next eigenvalue.

Once the eigenvalue, $E$, has been found sufficiently accurately, to at least 3 decimal places, the corresponding eigenvector can be found using the equations

$$\begin{aligned} e_0 &= 1 \\ e_1 &= 2(c_0 - \epsilon^2 E) \\ e_{n+1} &= 2(c_n - \epsilon^2 E)e_n - e_{n-1} \qquad n > 0. \end{aligned}$$

Note: for bound states the relevant eigenvectors are required to decay exponentially for large $|x|$. It can be shown that the matrix $M$ only has eigenvectors which satisfy this boundary condition.

There are three cautions:

(a) In Equation (1) there is a division by $q_{n-1}$. Should $q_{n-1}$ become too small it is permissible to replace it by a small default value, to avoid numerical instabilities: the results are unaffected by this procedure. For the cases considered below this eventuality has not been found to occur in practice.

(b) You will compute eigenvectors that are normalised as

$$\epsilon \sum_{n=0}^{2N-1} e_n^2 = 1$$

which corresponds to the physical normalisation $\int_{-\infty}^{\infty} |\psi|^2 \, dx = 1$. The wavefunction dies away at least exponentially for large $|x|$ so we expect $e_0$ to be very small indeed. For this reason, it is useful to continually normalise the vector $\mathbf{e}$, as it is being computed. Specifically, if the $e_n$ have already been calculated for all $n < m$ then it is recommended to normalise them such that

$$\epsilon \sum_{n=0}^{m-1} e_n^2 = 1$$

before computing $e_m$.

(c) The wavefunction also decays exponentially for large positive $x$. This means that for large $n$ (bigger than $N$), the values of $e_n$ will become very small. However, if you continue the calculation to very large $n$, numerical (round-off) errors can lead to exponential growth of the numerical estimates of $e_n$. The calculation is not accurate in this regime: if this happens you should stop the calculation at some $n_{\max} < 2N$, in order to obtain an accurate estimate of the true eigenvector $\mathbf{e}$. Alternatively, use the fact that all wavefunctions are either even or odd so the $e_n$ only need to be computed for $0 \leqslant n \leqslant N$. (Take care with the normalisation if you use this method).

**Programming Task:** Write a program to determine the eigenvalues and eigenvectors for the four lowest bound states of a given (symmetric) potential. You should allow the values of $N$ and $\epsilon$ and the starting value $\lambda$ to be input variables.

## 4 Harmonic Oscillator

[See the Appendix for some theoretical results which may be of use here and in later sections.]

As a check of your code give the four lowest eigenenergies for the potential

$$V(x) = \frac{1}{2}x^2.$$

Adjust $N$ and $\epsilon$ to get results to at least 3 decimal places for the eigenvalues and accurate to within 1% for the significant part of the wavefunctions. Make sure that $N\epsilon$ is not too big since the wavefunctions are very small for $x = N\epsilon$ and so nothing is gained. However, if $\epsilon$ is too big and/or $N\epsilon$ is too small the results will be inaccurate. A bit of trial and error will yield good values with which to work, but in each case considered you should check that results are insensitive to changes in $N$ and $\epsilon$ within the accuracy required. Reasonable values to start with are $N = 50$ and $\epsilon = 0.1$ but you should be able to increase $N$ up to 500 at least.

**Question 1** State the values of $N$ and $\epsilon$ that you have chosen, to obtain the required accuracy. Justify these choices. For these values, include plots of the wavefunctions corresponding to the two lowest energies and compare with the known analytic form.

Note: throughout this project, you should provide graphs that illustrate clearly the effect of the parameters and the similarities and differences between the wavefunctions. Large numbers of graphs are *very unlikely to be effective* in communicating this information.

## 5 Anharmonic Oscillator

Now modify the potential energy and take

$$V(x) = \frac{1}{2}x^2 + \frac{b}{6}x^4(x^2+1)\ .$$

**Question 2** Labeling the harmonic oscillator eigenstates by $|n\rangle$, i.e. $H_0|n\rangle = E_n|n\rangle$ with

$$H_0 = \frac{p^2}{2} + \frac{x^2}{2} = a^\dagger a + \frac{1}{2}\,,$$

explain or show that $\langle n+k|x^6|n\rangle = 0$ for all $k > 6$.

**Question 3** Using perturbation theory derive expressions for the lowest two energies to second order in $b$. You may use without proof, for integer $j \geqslant 0$, that

$$\int_{-\infty}^{\infty} dz\, z^{2j}\, e^{-z^2} = \Gamma(j+\tfrac{1}{2})\,,$$

with $\Gamma(\frac{1}{2}) = \sqrt{\pi}$ and $\Gamma(\sigma+1) = \sigma\Gamma(\sigma)$.

**Question 4** Compute (numerically) the four lowest energy eigenvalues and plot the corresponding wavefunctions for $b = 0.02$. Compare the results with your perturbative estimates for the lowest two energies. Try values $b = 0.001$ and $0.1$, as well as any others that you feel might be relevant. You need not plot the wavefunctions for these cases, but you should consider tables and/or plots of the energies vs. $b$. How well does perturbation theory work here? Is second order perturbation theory an improvement over first order?

## 6 Double-well potential

Finally consider

$$V(x) = \frac{1}{9d^4}(x^2 - d^2)^2 \left(d^2 + \frac{x^2}{8}\right) \tag{2}$$

This problem is not so easy to study by perturbation theory. Instead we use trial wavefunctions. (This approach is the same as the *variational method* described in the *Applications of Quantum Mechanics* course, but no knowledge of this method is required as full details are given below.)

**Question 5** Change variable to $y = x - d$ and show that

$$V = \frac{1}{2}y^2 + V_3 y^3 + \frac{7}{24d^2}y^4 + V_5 y^5 + \frac{1}{72d^4}y^6 .$$

where $V_3$ and $V_5$ are constants that you should express as real numbers times $d$ to some (negative) power.

Observe that for small $y$, the potential $V$ resembles the harmonic oscillator from question 1. A similar result occurs if we change variable to $y = x + d$ instead: the potential is symmetric in $x$ so this must happen. Based on this observation we introduce two wavefunctions that are ground states of the relevant harmonic oscillators:

$$\begin{aligned}\psi_+(x) &= \frac{1}{\pi^{1/4}}\, e^{-\frac{1}{2}(x-d)^2}\\ \psi_-(x) &= \frac{1}{\pi^{1/4}}\, e^{-\frac{1}{2}(x+d)^2} .\end{aligned}$$

Define two trial wavefunctions as

$$\phi_\pm = C_\pm(\psi_+ \pm \psi_-)$$

You will investigate how close are these wavefunctions to the solutions of the Schrödinger equation.

**Question 6** Determine the normalization constants $C_\pm$ and show that the expectation values of the Hamiltonian, $E_\pm = \langle\phi_\pm|H|\phi_\pm\rangle$ are

$$E_\pm = \frac{(A \pm B)}{(1 \pm e^{-d^2})} ,$$

where

$$A = \frac{1}{2} + \frac{7}{32d^2} + \frac{5}{192d^4} \qquad \text{and} \qquad B = e^{-d^2}\left(-\frac{7d^2}{18} + \frac{7}{48} + \frac{1}{16d^2} + \frac{5}{192d^4}\right).$$

**Question 7** What is the physical interpretation of the situation where $d \gg 1$, and what happens to the energy in this case? We focus on large $d$ and define $\beta = 1/(9d^4)$ which is a small parameter in this situation. Use your program to compute the energies of the two lowest eigenstates for various values of $\beta$ including $\beta = 1.0$, 0.1, 0.01, and 0.001. How accurate are these numerical estimates? Compare them with the theoretical estimates $E_\pm$.

**Question 8** What are the symmetry properties of the ground state eigenvector and the first excited state? Explain your answer. By expanding $\phi_\pm$ on the complete set of eigenstates of the Hamiltonian, show that $E_\pm$ are upper bounds for the energies of the ground state and first excited state. You will need to consider the symmetries of the wavefunctions.

**Question 9** For the cases considered in question 7, how close are the bounds $E_\pm$ to the true energies? For two interesting values of $d$, plot the potential and indicate the four lowest lying energy levels on the same plot. Is it possible for some of the energy levels to be **lower** than the height of the central peak, i.e. than $V$ at $x = 0$?

**Question 10** How well does the trial wavefunction method estimate the energy difference, $\Delta E$, between the first excited state and the ground state? What happens to $\Delta E$ as $\beta \to 0$?

## Appendix

You are given that the harmonic oscillator wavefunctions are

$$\psi_n(x) = \frac{C_n}{\pi^{1/4}} H_n(x)\, e^{-x^2/2}$$

where

$$\begin{aligned}
H_0 &= 1\,, & C_0 &= 1\\
H_1 &= 2x\,, & C_1 &= \frac{1}{\sqrt{2}}\\
H_2 &= 2(2x^2-1)\,, & C_2 &= \frac{1}{2\sqrt{2}}\\
H_3 &= 4x(2x^2-3)\,, & C_3 &= \frac{1}{4\sqrt{3}}\\
H_4 &= 4(4x^4-12x^2+3)\,, & C_4 &= \frac{1}{8\sqrt{6}}\\
H_5 &= 8x(4x^4-20x^2+15)\,, & C_5 &= \frac{1}{16\sqrt{15}}\\
H_6 &= 8(8x^6-60x^4+90x^2-15)\,, & C_6 &= \frac{1}{96\sqrt{5}}\\
H_7 &= 16x(8x^6-84x^4+210x^2-105)\,, & C_7 &= \frac{1}{96\sqrt{70}}
\end{aligned}$$

and so on.

# 7 Mathematical Methods

## 7.3 Minimisation Methods (8 units)

*There are no prerequisites for this project.*

*You should write your own minimisation method programs: it is not sufficient to use routines that are distributed as part of MATLAB, other software packages, or other programming languages. You can, however, exploit the matrix manipulation capabilities of MATLAB, other software packages, or other progamming languages. You should use MATLAB's 64-bit (8-byte) double-precision floating-point values or the equivalent in other programming languages. Because we will be investigating functions of at most three parameters, your program will take little time to complete the number of iterations that are specified in the questions. It is possible, therefore, to carry out many more iterations than are specified in the questions but you should answer questions based on the number of iterations that are specified. You may, of course, like to check what happens when much larger numbers of iterations are used.*

## 1 Introduction

There are many numerical methods for finding the least value of a function of $N$ variables, $f(x_1, x_2, x_3, \ldots) = f(\mathbf{x})$, say, given that the first derivatives

$$g_i = \frac{\partial f}{\partial x_i}, \qquad i = 1, 2, \ldots N \; , \tag{1}$$

can be calculated. Most of the methods are iterative and each iteration reduces the value of $f(\mathbf{x})$ by searching along a descent direction in the space of the variables in the following way.

The iteration begins with a starting point $\mathbf{x}_0$, and at this point the gradient vector $\mathbf{g}$ is calculated. Then a search direction, $\mathbf{s}$, say, is chosen, that satisfies the condition $\mathbf{g} \cdot \mathbf{s} < 0$ (the dot denotes a scalar product). It follows that if we move from $\mathbf{x}_0$ in the direction of $\mathbf{s}$, then the value of $f(\mathbf{x})$ becomes smaller initially. In other words the function of one variable

$$\phi(\lambda) = f(\mathbf{x}_0 + \lambda \mathbf{s}) \tag{2}$$

satisfies the condition $\phi'(0) < 0$ which is equivalent to $\mathbf{g} \cdot \mathbf{s} < 0$. The next stage is to consider the function $\phi(\lambda)$, and choose a value of $\lambda$, $\lambda^*$ say, that satisfies the inequality

$$f(\mathbf{x}_0 + \lambda^* \mathbf{s}) < f(\mathbf{x}_0) \; . \tag{3}$$

Usually $\lambda^*$ will be chosen [*] to minimise $\phi(\lambda)$. The vector $\mathbf{x}_0$ is replaced by $\mathbf{x}_1 = \mathbf{x}_0 + \lambda^* \mathbf{s}$ and another iteration is begun.

The project is to investigate three well-known algorithms (*Steepest Descents*, the *Conjugate Gradient algorithm*, and the *DFP algorithm*) by applying them to two functions. Using $x$ for $x_1$, $y$ for $x_2$, etc., consider the "bedpan function"

$$x + y + \frac{x^2}{4} - y^2 + (y^2 - \frac{x}{2})^2 \; , \tag{4}$$

[*] In your program for this project, you should allow for manual input of estimates for $\lambda^*$, based on plots of $\phi(\lambda)$. To speed up longer runs you will probably wish also to use MATLAB routines or other automated search algorithms to minimise $\phi(\lambda)$, but this is not required. In either case, the values of $\lambda^*$ used should appear in the hard copy of results.

and the following function, which has similar properties to the Rosenbrock function,

$$(1-x)^2 + 80(y-x^2)^2 \ . \tag{5}$$

In addition the following quadratic function of three variables will be used to demonstrate some properties of the DFP algorithm:

$$0.4x^2 + 0.2y^2 + z^2 + xz \ . \tag{6}$$

## 2 Steepest Descents

The Steepest Descents method simply uses the search direction $\mathbf{s} = -\mathbf{g}$. Write a program to implement the algorithm as described above. Use a simple $x$–$y$ plot of $\phi(\lambda)$ to help you determine $\lambda^*$ at each stage (you need never determine $\lambda^*$ to more than 2 significant figures). At each stage after the first, arrange for your program to display the current value of $f(\mathbf{x})$ and the decrease achieved over the last step. Also arrange for a plot of the iteration points $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, etc., (a sequence of line segments will illustrate the methods well). The iteration point plot may be built up as the calculation proceeds, or you can store the data and produce it on command from your programme at a point of your choosing.

[N.B. A well-implemented fully automatic algorithm for general use will need to have checks for special cases and exceptions built into it. For example, if a point $\mathbf{x}_n$ is encountered for which $\mathbf{g} \approx \mathbf{0}$ then a stationary point has been found and the process should quit. Likewise, if the iteration points are not changing significantly a fully automatic algorithm ought to quit. You may find it helpful to include such features in your program. If you wish to proceed semi-automatically, with $\lambda^*$ being decided by eye from the plot at each stage, there is no need to include the special checks in your code.]

**Question 1** Obtain contour plots and/or surface plots of functions (4) and (5) (this should be fairly straightforward to do using MATLAB).

Work out analytically where they have minima and find their minimum values. Suitable axis intervals are $-1.5 \leqslant x, y \leqslant 1.5$.

**Question 2** Using function (4) and starting from $(-1.0, -1.3)$, run the Steepest Descents method for 10 iterations. Produce a plot of the progress of the iteration. On the basis of your numerical results (i.e., imagine that you do not know the analytical answer), estimate the minimum value of the function at the point to which your iteration is converging, and estimate intervals in which the co-ordinates of the minimum lie. What general statement can you make about the precision with which the minimum value itself can be found, compared to the precision with which the minimum point is known? What property of the function being minimised gives rise to this effect?

**Question 3** Using function (5) and starting from $(0.676, 0.443)$, run the Steepest Descents method for 9–15 iterations, and produce a plot of the progress of the iteration. To what point do you think the iteration will eventually converge? Comment on the rate of convergence. How sensitive is the iteration path to variations in the choice of $\lambda^*$ at each stage? Comment on the circumstances that can make steepest descents inefficient.

## 3 Conjugate Gradients

The conjugate gradients algorithm uses steepest descents for its first step and then adjusts the search direction in an attempt to overcome the problems of steepest descents alone. Let $\mathbf{x}_0$, $\mathbf{x}_1$ be two successive points where $\mathbf{x}_1$ has been obtained using steepest descents from $\mathbf{x}_0$, and let $\mathbf{g}_0$, $\mathbf{g}_1$ be the corresponding gradients (the initial search direction is $\mathbf{s}_0 = -\mathbf{g}_0$). Take the second search direction as

$$\mathbf{s}_1 = -\mathbf{g}_1 + \beta\, \mathbf{s}_0 = -\mathbf{g}_1 - \beta\, \mathbf{g}_0 \;\text{ where }\; \beta = \frac{\mathbf{g}_1 \cdot \mathbf{g}_1}{\mathbf{g}_0 \cdot \mathbf{g}_0}. \tag{7}$$

If $f(\mathbf{x})$ is a quadratic function of $N$ variables then the choice of directions may be continued up to the $N^{\text{th}}$ search direction to give the $N$ conjugate directions

$$\mathbf{s}_k = -\mathbf{g}_k + \beta\, \mathbf{s}_{k-1} \;\text{ where }\; \beta = \frac{\mathbf{g}_k \cdot \mathbf{g}_k}{\mathbf{g}_{k-1} \cdot \mathbf{g}_{k-1}}\ .$$

In this case, if all the values of $\lambda^*$ had been chosen to minimise the $\phi(\lambda)$ exactly at each stage, the algorithm would have converged. In practice of course $f(\mathbf{x})$ may not be quadratic and the values $\lambda^*$ may not be chosen exactly, and in this case it is usual in practice to restart the method after $N$ steps. When $N = 2$, as it is for functions (4) and (5), this implies that every other step is a steepest descent.

Write a program to implement the conjugate gradients algorithm, with the same features as used for the steepest descents method, but with the search direction determined as just described.

> **Question 4** For the function (4), repeat Q2 using the conjugate gradients algorithm, and compare results.

> **Question 5** For the function (5), repeat Q3 using the conjugate gradients algorithm, and compare results.

Does the conjugate gradients algorithm offer much of an improvement over steepest descents?

## 4 DFP Algorithm

The Taylor Series expansion of any smooth function $f(\mathbf{x})$ may be written

$$f(\mathbf{a} + \Delta\mathbf{x}) \cong f(\mathbf{a}) + \mathbf{g} \cdot (\Delta\mathbf{x}) + \tfrac{1}{2}(\Delta\mathbf{x})^T \mathbf{H}^{-1}(\Delta\mathbf{x}) + \cdots$$

where the gradient vector $\mathbf{g}$ is evaluated at $\mathbf{x} = \mathbf{a}$ and $\mathbf{H}^{-1} \equiv \mathbf{G}$ is the Hessian matrix, i.e., the matrix of second derivatives

$$G_{ij} = \frac{\partial^2 f}{\partial x_i\, \partial x_j}\ .$$

Finding a point where $\mathbf{g}$ vanishes is therefore similar to the Newton–Raphson method for a system of equations, and if $\mathbf{H}$ were known and $f(\mathbf{x})$ were a quadratic function, the point could be found in a single step. However the matrix $\mathbf{H}$ is not available initially unless second derivatives are calculated; this is not always easy and in any case can be time-consuming, especially for large $N$. Therefore we now study a very successful technique that extends the steepest descent method by forming a suitable $\mathbf{H}$-matrix as the calculation proceeds. It is known as the DFP algorithm and is one of the class of "variable metric methods". It can be shown that $\mathbf{H}^{-1}$ converges to the Hessian (you are not required to prove this).

The DFP algorithm works as follows. The search direction is taken as $\mathbf{s} = -\mathbf{H}\mathbf{g}$, where $\mathbf{H}$ is taken initially as the identity matrix. At each stage $\phi(\lambda)$ is minimised by choosing a value $\lambda^*$ as before, but then $\mathbf{H}$ is modified by replacing it with

$$\mathbf{H}^* = \mathbf{H} - \frac{\mathbf{H}\mathbf{p}\mathbf{p}^T\mathbf{H}}{\mathbf{p}^T\mathbf{H}\mathbf{p}} + \frac{\mathbf{q}\mathbf{q}^T}{\mathbf{p}^T\mathbf{q}}, \tag{8}$$

where $\mathbf{p}$ and $\mathbf{q}$ are column vectors giving the changes in $\mathbf{g}$ and $\mathbf{x}$ respectively during the step, that is

$$\mathbf{p} = \mathbf{g}(\mathbf{x}_0 + \lambda^*\mathbf{s}) - \mathbf{g}(\mathbf{x}_0), \quad \mathbf{q} = \lambda^*\mathbf{s} \tag{9}$$

(Note: $\mathbf{H}^*\mathbf{p} = \mathbf{q}$ which is useful when checking your program.)

Write a program to implement the DFP algorithm, with the same features as used for the two preceding programs, but with the search direction determined as just described. Include provision to print out $\mathbf{H}$.

**Question 6** A property of the DFP algorithm is that it calculates the least value of a quadratic function in at most $N$ iterations for any initial choice of $\mathbf{x}_0$ if on each iteration the value of $\lambda^*$ is calculated to minimise exactly the function $\phi(\lambda)$. Apply the DFP algorithm to (6) for three iterations from starting point $\mathbf{x}_0 = (1, 1, 1)$ using the sequence of values

$$\lambda^* = 0.3942,\ 2.5522,\ 4.2202.$$

There is no need to verify these values to this precision, but your program will already have facilities for checking that these values are appropriate. Investigate how sensitive the result obtained after three iterations is to small changes in these values. Verify that $\mathbf{H}$ does indeed tend to the inverse Hessian matrix. You may note that

$$\begin{pmatrix} 0.8 & 0 & 1 \\ 0 & 0.4 & 0 \\ 1 & 0 & 2 \end{pmatrix}^{-1} = \begin{pmatrix} 3.3333 & 0 & -1.6667 \\ 0 & 2.5 & 0 \\ -1.6667 & 0 & 1.3333 \end{pmatrix} \tag{10}$$

**Question 7** For the function (4), repeat Q2 using the DFP algorithm. Examine $\mathbf{H}$ and compare with the true value.

**Question 8** For the function (5), repeat Q3 using the DFP algorithm. Examine $\mathbf{H}$ and compare with the true value.

**Question 9** Compare the performance of the three methods for these functions.

## References

[1] Fletcher, R. and Powell, M.J.D. *Rapidly convergent descent method for minimisation*, Computer Journal, 7 (1963).

[2] McKeown, J.J., Meegan, D., and Sprevak, D. *An Introduction to Unconstrained Optimisation - A Computer Illustrated Text*, IOP Publishing (1990). Although references to the computer programs (designed for BBC micro) are best ignored, the text is still relevant.

# 7 Mathematical Methods

## 7.4 Airy Functions and Stokes' Phenomenon (9 units)

*This project uses ideas from the Further Complex Methods course. It also covers some material which is lectured as part of the Asymptotic Methods course, but students not taking this course are at no disadvantage.*

## 1 Introduction

The Airy functions $\mathrm{Ai}(z)$ and $\mathrm{Bi}(z)$, where $z$ is a complex variable, are two linearly independent solutions of the differential equation

$$\frac{\mathrm{d}^2}{\mathrm{d}z^2}\, y(z) = zy(z) \tag{1}$$

satisfying

$$\mathrm{Ai}(0) = \alpha, \qquad \mathrm{Ai}'(0) = -\beta, \qquad \mathrm{Bi}(0) = \sqrt{3}\,\alpha, \qquad \mathrm{Bi}'(0) = \sqrt{3}\,\beta$$

where

$$\alpha = \frac{1}{3^{2/3}\,\Gamma(\frac{2}{3})} \approx 0.355028053887817, \qquad \beta = \frac{1}{3^{1/3}\,\Gamma(\frac{1}{3})} \approx 0.258819403792807.$$

Here $\Gamma$ is the Gamma function, defined by

$$\Gamma(z) = \int_0^\infty e^{-t} t^{z-1} \mathrm{d}t, \tag{2}$$

but you do not need to know anything about its properties for this project. The Airy functions are useful in many problems involving transition regions of all kinds, for example in optical diffraction (the transition between relatively light and dark regions), wave theory, electron tunnelling, and asymptotic analysis. Ai and Bi have Maclaurin series given by

$$\mathrm{Ai}(z) = \alpha f(z) - \beta g(z), \qquad \mathrm{Bi}(z) = \sqrt{3}\big(\alpha f(z) + \beta g(z)\big)$$

where

$$f(z) = 1 + \frac{1}{3!}z^3 + \frac{1\cdot 4}{6!}z^6 + \frac{1\cdot 4\cdot 7}{9!}z^9 + \cdots$$

and

$$g(z) = z + \frac{2}{4!}z^4 + \frac{2\cdot 5}{7!}z^7 + \frac{2\cdot 5\cdot 8}{10!}z^{10} + \cdots .$$

For large $|z|$, any solution $y(z)$ of (1) is given asymptotically by the relation

$$y(z) \sim AF(z) + BG(z)$$

where $A$ and $B$ are complex constants, and where

$$F(z) = \frac{1}{\sqrt{\pi}}\, z^{-1/4} \exp(-\frac{2}{3}z^{3/2})(1 - \frac{5}{48}z^{-3/2} + \cdots)$$

and

$$G(z) = \frac{1}{\sqrt{\pi}}\, z^{-1/4} \exp(\frac{2}{3}z^{3/2})(1 + \frac{5}{48}z^{-3/2} + \cdots),$$

where the principal value is taken for any multi-valued function. The values of the constants $A$ and $B$ depend, of course, on precisely which solution $y$ is being considered (Ai and Bi have different asymptotic behaviour, for instance). More surprisingly, perhaps, the values of $A$ and $B$ may also depend on which region of the complex plane is under consideration. This is known as Stokes' phenomenon, and the rays from the origin that divide the complex plane into different regions are known as Stokes lines. In the current case, there are three Stokes lines, two of which are given by the rays $\arg z = \pm\pi/3$.

In this project, we shall concentrate to start with on the region $R$ given by $|\arg z| < \pi/3$. In that region, the appropriate values of $A$ and $B$ are $\frac{1}{2}$ and 0 respectively for $\mathrm{Ai}(z)$; for $\mathrm{Bi}(z)$, $B = 1$ but $A$ is not important and can take any value (because $F(z)$ is negligible compared to $G(z)$ for large $|z|$ in $R$; that is, $F$ is *subdominant*). Hence $\mathrm{Ai}(z) \to 0$ and $|\mathrm{Bi}(z)| \to \infty$ as $|z| \to \infty$ in $R$.

**Programming note:** You should write your own programs to compute the Airy functions: it is not sufficient simply to use the inbuilt MATLAB functions, or equivalent inbuilt functions for other software packages or programming languages, to calculate Airy functions although they are, of course, a convenient way to check your results. All calculations and evaluations are to be performed for *complex* numbers, not just real ones. You should use MATLAB's 64-bit (8-byte) double-precision floating-point and complex number values or the equivalent in other programming languages. Although MATLAB handles complex numbers quite well, most programming languages handle only real numbers, so you may have to write your own code to perform simple complex number operations such as multiplication.

**Question 1** Show that

$$y(z) = \frac{1}{2\pi i}\int_C \exp(zt - \frac{1}{3}t^3)\,\mathrm{d}t$$

is a solution of 1. Here $C$ is any contour that starts at $\infty e^{-2\pi i/3}$ and ends at $\infty e^{2\pi i/3}$. Show furthermore that this solution satisfies $y(0) = \alpha$, $y'(0) = -\beta$ and that it is therefore equal to $\mathrm{Ai}(z)$. [*Hint: deform $C$ into two (straight) rays that meet at the origin. You may assume without proof the reflection formula for the Gamma function, viz.* $\Gamma(z)\,\Gamma(1-z) = \pi/\sin(\pi z)$.]

This integral representation of $\mathrm{Ai}(z)$ can be used to check the asymptotic expansion given above for large $|z|$, but you are not required to do this.

## 2 Numerical Integration of the Differential Equation

**Question 2** Write a program to find $\mathrm{Bi}(z)$ for any $z \in R$, accurate to at least 4 significant figures, by performing a numerical integration of the defining differential equation (1) using any standard method. You should perform your integration along a ray joining the origin to $z$, using a real variable $t$ to denote distance along the ray: this will require you to find a system of differential equations satisfied by $\operatorname{Re} y$ and $\operatorname{Im} y$ along the ray. Include the derivation of this system of equations in your write-up as well as the initial conditions (over which you are advised to take care). Also explain what checks you carried out to ensure the accuracy of your solutions. As a very simple first check, you may find it useful to know that $\mathrm{Bi}(1) \approx 1.20742$.

Use your program to evaluate $\mathrm{Bi}(z)$ at $z = 2, 4, 8, 16, e^{\pm i\pi/6}$ and one other non-real point of your choosing. Draw a graph of the behaviour of the (modulus of the) solution along one particular non-real ray of your choosing and give a plausible demonstration that the leading order asymptotic behaviour, $G(z)$, is indeed as stated in the Introduction.

**Question 3** Modify your program to instead calculate $\mathrm{Ai}(z)$, and try to evaluate $\mathrm{Ai}(z)$ at the same points as in Question 2. You may find it useful to know that $\mathrm{Ai}(1) \approx 0.13529$. Draw a graph of $\mathrm{Ai}(z)$ for real positive $z$. Which of your evaluations are you confident are accurate? What goes wrong with the method? Why is this unavoidable?

One way to avoid this problem is, instead of integrating from $z = 0$ towards infinity, to start from a value of $z$ with *large* modulus, and step towards the origin. The asymptotic expansion for $\mathrm{Ai}(z)$ (and the derivative of this expansion) can be used to approximate the initial conditions.

**Question 4** Explain why this alternative approach should work. Write a program to implement it; start from $|z| = a$, for some large fixed constant $a$, and integrate towards the origin. Use only the zeroth order term of the asymptotic expansion (i.e., ignore $\frac{5}{48}z^{-3/2}$ and higher order terms in $F(z)$); a more advanced implementation might take more terms into account.

To start with you might like to use $a = 20$; but you should experiment with other values and explain what difference they might make. State the value you finally settle on and why.

Use your program to evaluate $\mathrm{Ai}(z)$ at the same points as in Question 2.

## 3 Matched expansions

**Question 5** By finding series expansions about the origin, or otherwise, prove that the given expressions for the Maclaurin series of $\mathrm{Ai}(z)$ and $\mathrm{Bi}(z)$ are correct.

A much quicker, and more accurate, approach to evaluating the Airy functions is to avoid numerical integration altogether and instead use the analytic series expansions. In theory, the Maclaurin series for Ai and Bi are valid for all $z$, but in practice they are not very helpful for larger values of $|z|$ because of rounding errors caused by adding together large numbers of terms. Here we will try an approach based on using the Maclaurin series when $|z| < b$, for some fixed constant $b$, and using the asymptotic expansion when $|z| \geqslant b$; we hope to achieve accuracy at least as high as 4 significant figures, and preferably more.

**Question 6** Investigate the feasibility and potential accuracy of this approach for evaluating $\mathrm{Ai}(z)$ on the positive real axis. You should use only the first two terms in the asymptotic expansion (i.e., do not attempt to find more terms in $F(z)$ than are given above), though you may use as many terms of the Maclaurin series as you wish. You should try various different values of $b$, and experiment with the number of terms to use from the Maclaurin series for best results. What level of accuracy is attainable?

Include a plot of your composite approximation and some sample values close to $|z| = b$.

How did you sum the Maclaurin series in order to minimize rounding errors?

How do you expect the time taken by this algorithm to compare with that for Question 4?

A professional implementation of this method (at least for real $z$) would use a selection of Chebyshev polynomial approximations in different overlapping regions and choose the best one automatically.

## 4 Stokes lines

**Question 7** Use the programs you have developed in previous questions to describe how the behaviour of Ai and Bi with $|z|$ changes as the rays approach the Stokes line at $\arg z = \pi/3$ from within $R$.

**Question 8** By experimenting with rays *outside* $R$, determine the location of the third Stokes line. How do Ai and Bi behave on this line?

**Question 9** What can you say about the values of $A$ and $B$ in each of the three regions which lie between each pair of Stokes lines? Can you estimate these values from your numerical results?

What, if anything, can you say *on* the Stokes lines?

## 5 A particle in a constant force field

[Note that no knowledge of Quantum Mechanics is required for this section of the project: all required equations are given below.]

A one-dimensional quantum-mechanical particle is confined to the region $x > 0$ and is subjected to a force of constant magnitude $k$ directed towards the origin. The governing equation for the wavefunction $\psi(x)$ is

$$-\frac{\hbar^2}{2m}\frac{\mathrm{d}^2\psi}{\mathrm{d}x^2} + kx\psi = \lambda\psi$$

with boundary conditions $\psi(0) = 0$, $\psi(x) \to 0$ as $x \to \infty$, where $\lambda$ is the energy of the particle. This is a Sturm–Liouville problem with eigenvalue $\lambda$.

**Question 10** Show, using your computed results from earlier questions, that there is a discrete set of energy eigenvalues $\lambda_n$. Find an approximate value for the first two of these eigenvalues in units where $\hbar^2 k^2/2m = 1$.

# 9 Dynamic Programming

## 9.1 Policy Improvement for a Markov Decision Process (4 units)

*This project is self-contained mathematically; background information is provided in the Part II course on Optimisation and Control (see reference [1]).*

### 1 A Car Replacement Problem

Car owners are haunted by the following problem. Every day, the operating cost for their car increases, as does the probability that the car breaks down. Even worse, when trading in the car for a different one dealers will pay less for older cars and charge more for newer ones. The problem, then, is to find an optimal policy for trading in the car.

We model the problem as a Markov decision process. Let $g_j(u)$ be the instantaneous cost incurred if one takes action $u$ in state $j$ and let $p_{jk}(u)$ be the probability of then moving to state $k$. Define sequences $\gamma^{(n)}$, $f_j^{(n)}, u_j^{(n)}$ by the recursions

$$\gamma^{(n)} + f_j^{(n)} = g_j\big(u_j^{(n)}\big) + \sum_k p_{jk}\big(u_j^{(n)}\big) f_k^{(n)} \tag{1}$$

and

$$u_j^{(n+1)} \text{ is the } u\text{-value minimising } g_j(u) + \sum_k p_{jk}(u) f_k^{(n)}. \tag{2}$$

The following exercise may help to gain some intuition.

**Question 1** Consider the following stationary policy: for fixed $n$, whenever state $j$ occurs take action $u_j^{(n)}$. What is the long-term average cost of this policy? Explain.

Note that the values $f_j^{(n)}$ determined by (1) are arbitrary up to an additive constant and can be normalised, for example by letting $f_1^{(n)} = 0$. If the matrix of transition probabilities is irreducible in every stage, then (1) will always have a solution for $f$. The sequence $\gamma^{(n)}$ is non-increasing, and will converge to a minimum value $\gamma$ in a finite number of steps if $u$ can take only a finite number of values. The policy $u_j^{(n)}$ will then have converged to an average optimal policy.

**Question 2** Instantiate the above framework for the car replacement problem. You may want to introduce states representing the age of the car in appropriately chosen units of time, and an additional state in which the car is written off and has a trade-in value of zero. Describe the set of actions, and define the instantaneous costs $g_j(u)$ and the transition probabilities $p_{jk}(u)$.

**Question 3** Write a program to find the optimal replacement policy. You are not required to write your own linear algebra routines, but you should describe any mathematical manipulations involved in bringing the equations in the desired form. Give a

| $j$ | age in years | purchase price | trade-in price | operating cost | survival probability |
|---|---|---|---|---|---|
| 1 | 0 | 5000 | 3500 | 860 | 0.963 |
| 2 | 2 | 3150 | 2170 | 1025 | 0.794 |
| 3 | 4 | 2285 | 1500 | 1225 | 0.568 |
| 4 | 6 | 1545 | 900 | 1430 | 0.255 |
| 5 | 8 | 1050 | 590 | 1815 | 0.001 |
| 6 | 10 | 600 | 330 | 2240 | 0.000 |

Table 1: Instance of the car replacement problem with time units of two years and $N = 6$

| $j$: | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| sell/keep: | keep | keep | keep | keep | keep | sell | sell |
| buy car of age: | – | – | – | – | – | 2 | 2 |

Table 2: Policy for the problem of Table 1

| $j$ | age in years | purchase price | trade-in price | operating cost | survival probability |
|---|---|---|---|---|---|
| 1 | 0 | 5000 | 3500 | 200 | 0.999 |
| 2 | 0.5 | 4285 | 3000 | 210 | 0.995 |
| 3 | 1 | 3750 | 2650 | 220 | 0.990 |
| 4 | 1.5 | 3430 | 2375 | 230 | 0.979 |
| 5 | 2 | 3150 | 2170 | 240 | 0.968 |
| 6 | 2.5 | 2900 | 1950 | 250 | 0.956 |
| 7 | 3 | 2645 | 1850 | 260 | 0.936 |
| 8 | 3.5 | 2475 | 1625 | 275 | 0.917 |
| 9 | 4 | 2285 | 1500 | 290 | 0.898 |
| 10 | 4.5 | 2130 | 1350 | 300 | 0.879 |
| 11 | 5 | 1970 | 1225 | 315 | 0.860 |
| 12 | 5.5 | 1760 | 1060 | 320 | 0.836 |
| 13 | 6 | 1545 | 900 | 335 | 0.801 |
| 14 | 6.5 | 1400 | 780 | 350 | 0.761 |
| 15 | 7 | 1260 | 700 | 365 | 0.697 |
| 16 | 7.5 | 1140 | 625 | 380 | 0.600 |
| 17 | 8 | 1050 | 590 | 400 | 0.482 |
| 18 | 8.5 | 940 | 520 | 430 | 0.300 |
| 19 | 9 | 830 | 470 | 465 | 0.129 |
| 20 | 9.5 | 720 | 400 | 520 | 0.020 |
| 21 | 10 | 600 | 330 | 560 | 0.000 |

Table 3: Instance of the car replacement problem with time units of six months and $N = 21$

clear and concise description of your algorithm; don't forget to mention what starting conditions you use. Run your program on the data contained in the file *table1.csv* available from the CATAM website and displayed in Table 1, and compare your results to the policy in Table 2. What is the value of $\gamma$?

**Question 4** Give the optimal replacement policy for the data in the file *table2.csv* available from the CATAM website and displayed in Table 3. What is the value of $\gamma$?

**Question 5** Suppose that purchase price, trade-in price, operating cost, and survival probability are all monotonically increasing or decreasing in the obvious direction. Suppose further that the optimal policy tells you to sell a car when it reaches a certain age, but that you neglect to do so. Is it possible that the same policy stipulates hanging on to the car now that it is older? Either construct an example for which the optimal policy is of this kind, or prove that this is impossible.

## References

[1] R.R.Weber, *Course notes on Optimisation and Control*, Section 8.4. `http://www.statslab.cam.ac.uk/~rrw1/oc/`.

# 9 Dynamic Programming

## 9.4 Option Pricing in Mathematical Finance (6 units)

*This project is connected with material in the Stochastic Financial Models course. Students who are not taking that course but who wish to attempt the project will find the necessary definitions and background material in the references.*

### 1 Black-Scholes model

A standard model used in option pricing is that the logarithm of the stock price follows a Brownian motion. Hence, if $S_t$ is the stock price at time $t$, we assume that $\log(S_t/S_0)$ is normally distributed with mean $\mu t$ and variance $\sigma^2 t$, where $\sigma$ is the volatility, $\mu = \rho - \sigma^2/2$, and $\rho$ is the continuously-compounded riskless interest rate.

The celebrated Black-Scholes formula gives the price of a call option (exercised only at expiry). The price of the option is

$$S_0\Phi\Big(\frac{\log(S_0/c)+(\rho+\sigma^2/2)t_0}{\sigma\sqrt{t_0}}\Big) - ce^{-\rho t_0}\Phi\Big(\frac{\log(S_0/c)+(\rho-\sigma^2/2)t_0}{\sigma\sqrt{t_0}}\Big) \tag{1}$$

where $c$ is the strike price and $t_0$ is the expiry time. (See the Appendix for details of how to calculate $\Phi$.)

> **Question 1** Write a routine to evaluate the Black-Scholes price (1). Compile a table of the price when $c = 40$, $S_0 = 52, 100$ or 107, $\sigma = 0.5$, $\rho = 0.035$, and $t_0 = 2$ or 3.

> **Question 2** How does the price vary with each of the parameters $c$, $S_0$, $\sigma$, $\rho$, $t_0$? Keep your explanations brief, but support them with solid mathematics where necessary.

### 2 Bernoulli approximation

The most widely-used method for approximating option prices which are based on the Black-Scholes model is to replace the Brownian motion by a discrete-time simple random walk. This approximation breaks up the interval $[0, t_0]$ into $[0, t_0/n, 2t_0/n, \ldots, (n-1)t_0/n, t_0]$ and assumes that between the times $it_0/n$ and $(i+1)t_0/n$, $i = 0, \ldots, n-1$, the increment in the logarithm of the price is $g$ or $-g$ with probability $p$ or $1-p$ respectively, where $g$ and $p$ are chosen so that the increment has mean $\mu t_0/n$ and variance $\sigma^2 t_0/n$.

This approximation is primarily of interest for cases such as the American put where exact formulae are not available. For a European (or American) call option, the price obtained from the random walk will approximate the true price obtained from the Black-Scholes formula, and this can be a useful benchmark to judge the performance of the approximation.

One way to implement the approximation is to set

$$V_{i,j} = \big(pV_{i+1,j+1} + (1-p)V_{i+1,j}\big)e^{-\rho t_0/n} \quad \text{for } j = 0, \ldots, i \text{ and } i = n-1, \ldots, 0 \tag{2}$$

with boundary conditions

$$V_{n,j} = \Big(S_0 e^{(2j-n)g} - c\Big)^+ \quad \text{for } j = 0, \ldots, n. \tag{3}$$

Then $V_{0,0}$ is the approximate price.

**Question 3** Calculate $g$ and $p$ as functions of the parameters.

**Question 4** For the data in Question 1 and $n = 27$, compile a table of the approximate prices. How do they compare to the prices obtained from the Black-Scholes formula?

**Question 5** What is the complexity of this algorithm, as a function of $n$?

**Question 6** Consider an at-the-money case ($c = S_0$). Plot a graph of your approximation as a function of $n$. Indicate on your graph the true value obtained from the Black-Scholes formula. What do you notice? Explain this behaviour.

**Question 7** Estimate the rate at which the error decreases as $n$ increases. Explain every step and justify your answers.

## 3 American Put

Consider the case of an American put; now early exercise of the option may be optimal and no closed-form formula exists for the price.

**Question 8** Modify your programs to approximate the price of the option by considering how equations (2) and (3) should be changed in this situation. Compile a table of the approximate price of the option for the same values of the parameters used in Question 1. Comment briefly on how the approximate price varies with $n$ in this case.

## 4 Extrapolation

Suppose that $f_n$ is the approximation to the option price, and we wish to find the limiting value of $f_n$ as $n \to \infty$. One method of extrapolation assumes that $f_n$ may be approximated by a polynomial in $1/n$:

$$f_n \approx g_0 + g_1 n^{-1} + g_2 n^{-2} + \cdots + g_s n^{-s}.$$

The limiting value of $f_n$ is then approximated by $g_0$. One way to achieve this is as follows. Let $n_m = r^m n_0$ and calculate $f_n$ at $n = n_0, \ldots, n_s$. Set

$$a_{m,0} = f_{n_m} \quad \text{for } m = 0, \ldots, s$$

and recursively calculate

$$a_{m,i} = a_{m,i-1} + \frac{a_{m,i-1} - a_{m-1,i-1}}{r^i - 1} \quad \text{for } m = i, \ldots, s \text{ and } i = 1, \ldots, s.$$

Then $a_{s,s}$ is taken as the approximation for $g_0$.

**Question 9** Experiment with this extrapolation procedure for small values of $r$ and $s$, say 2 to 4, for the at-the-money European call case studied above. How does this extrapolation compare in accuracy with just calculating $f_n$ for a single suitably large value of $n$? Try to estimate the error analytically. Does your answer depend on whether $n_0$ is odd or even? If so, carefully explain why.

## 5 Binomial approximation

The approximation in Section 2 uses a Bernoulli (two-valued) distribution between time steps. The method may be refined by replacing the Bernoulli distribution between times $it_0/n$ and $(i+1)t_0/n$ by, say, a binomial distribution taking $k+1$ equally-spaced values (for some $k \geqslant 1$) with mean and variance chosen to match those of the Brownian motion.

> **Question 10** Implement this refinement, and explain how you calculate $p$ and $g$ in this case. How do prices produced by the refined algorithm ($k > 1$) differ from those produced by the Bernoulli scheme ($k = 1$)? Does your answer depend on whether you are pricing the European call or American put option, and if so why? How does the computation time for this algorithm vary with $n$ and $k$?

## Appendix

An easy method to approximate the standard normal distribution function is as follows. For $x \geqslant 0$ set

$$1 - \Phi(x) = \frac{t}{2}\exp\Big(-\frac{x^2}{2} + \sum_{i=0}^{9} a_i t^i\Big)$$

where $t = (1 + x/\sqrt{8})^{-1}$ and

$$\begin{aligned}(a_0, \dots, a_9) = (&-1.26551223, 1.00002368, 0.37409196,\\ &0.09678418, -0.18628806, 0.27886807,\\ &-1.13520398, 1.48851587, -0.82215223, 0.17087277).\end{aligned}$$

For $x < 0$ set $\Phi(x) = 1 - \Phi(-x)$.

## References

[1] J. Hull, *Options, Futures and Other Derivative Securities*. Prentice-Hall, 1989.

# 10 Statistics

## 10.9 Markov Chain Monte Carlo (6 units)

*Bayesian inference is covered in the IB Statistics course, and developed further in the II(D) Principles of Statistics course. Knowledge of the IB Markov Chains course, whilst useful, is not necessary, and there is no requirement to quote results from it.*

## Introduction

In Bayesian statistics, it is essential to be able to sample from the posterior distribution of unknown parameters given some data. In arbitrary, high-dimensional problems, this is not possible analytically, but in recent years Markov Chain Monte Carlo methods (MCMC) have become a popular alternative.

### Markov Chain

The key idea is quite simple. We want to sample from a distribution $\pi(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^m$, but cannot do so directly. Instead we create a discrete-time Markov chain $\mathbf{X}(n)$ (taking values in $\mathbb{R}^m$) such that $\mathbf{X}(n)$ has equilibrium distribution $\pi$. Then

$$\mathbf{X}(n) \to \mathbf{X} \sim \pi \quad \text{in distribution, as } n \to \infty \tag{1}$$

and
$$\frac{1}{N}\sum_{n=1}^{N} f\big(\mathbf{X}(n)\big) \to \mathbb{E}_\pi\big(f(\mathbf{X})\big) \quad \text{as } N \to \infty$$

where $f$ is any real-valued function on $\mathbb{R}^m$ for which the right-hand side above is well-defined. The second of these limits can be used to calculate means and variances of components of $\mathbf{X}$, as well as approximations to the distribution functions. For example, $f(\mathbf{x}) = x_i$ gives the mean of the $i$th component $X_i$, and $f(\mathbf{x}) = I(x_i \leqslant b)$ gives the distribution function of $X_i$ at point $b$.

### Gibbs sampler

Suppose we do not have a tractable closed-form expression for the equilibrium density $\pi(\mathbf{x}) = \pi(x_1, \ldots, x_m)$, but we do know the induced full conditional densities $\pi(x_i|\mathbf{x}_{-i})$, where $\mathbf{x}_{-i}$ is the vector $\mathbf{x}$ omitting the $i$th component, $\mathbf{x}_{-i} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_m)$.

A systematic form of the Gibbs sampler algorithm proceeds as follows. First, pick an arbitrary starting value $\mathbf{x}^0 = (x_1^0, \ldots, x_m^0)$. Then successively make random drawings from the full conditional distributions $\pi(x_i|\mathbf{x}_{-i})$, $i = 1, \ldots, m$, as follows:

$$\begin{aligned}
&x_1^1 \text{ from } \pi(x_1|\mathbf{x}_{-1}^0)\\
&x_2^1 \text{ from } \pi(x_2|x_1^1, x_3^0, \ldots, x_m^0)\\
&x_3^1 \text{ from } \pi(x_3|x_1^1, x_2^1, x_4^0, \ldots, x_m^0)\\
&\vdots\\
&x_m^1 \text{ from } \pi(x_m|\mathbf{x}_{-m}^1).
\end{aligned}$$

This cycle completes a transition from $\mathbf{x}^0 = (x_1^0, \ldots, x_m^0)$ to $\mathbf{x}^1 = (x_1^1, \ldots, x_m^1)$. Repeating the cycle produces a sequence $\mathbf{x}^0, \mathbf{x}^1, \mathbf{x}^2, \ldots$ which is a realization of a Markov chain, which is known as the Gibbs sampler. We call $\pi(\mathbf{x}, \mathbf{y})$ the transition probability density of this Markov chain.

> **Question 1** Assume that the Markov chain $\mathbf{X}(n)$ takes values in a finite subset $S \subset \mathbb{R}^m$. Verify that $\pi$ is an equilibrium distribution for this chain. That is, check that for all $\mathbf{y} \in S$,
>
> $$\sum_{\mathbf{x}} \pi(\mathbf{x})\pi(\mathbf{x}, \mathbf{y}) = \pi(\mathbf{y}).$$

It can be shown that this implies that $\pi$ is the equilibrium distribution of the Gibbs sampler, in the sense of (1), but do not attempt to prove it. Thus our estimate of $\mathbb{E}_\pi\big(f(\mathbf{X})\big)$, taken over $N$ iterations, is

$$\frac{1}{N}\sum_{n=1}^{N} f(\mathbf{x}^n).$$

## Football data

Data from the performance of $K$ football teams, over $T$ years has been scored on a scale of 0 (no wins) to 114 (win in all 38 games), with a win scoring three and a draw scoring one point. Let us model $Y_{kt}$, the score of the $k$th team in year $t$, as

$$Y_{kt}|\text{parameters} \sim \mathrm{N}(\mu_k, \sigma_k^2), \quad \text{for } k = 1, \ldots, K \text{ and } t = 1, \ldots, T$$

with the hierarchical prior structure that the team mean $\mu_k$ and variance $\sigma_k^2$ are independently distributed, given $\theta$, as

$$\begin{aligned} \mu_k|\theta &\sim \mathrm{N}(\theta, \sigma_0^2) \\ \sigma_k^{-2} &\sim \Gamma(\alpha_0, \beta_0), \end{aligned}$$

where $\sigma_0^2$, $\alpha_0$ and $\beta_0$ are known parameters, and $\theta$ is a second-stage prior with distribution

$$\theta \sim \mathrm{N}(\mu_0, \tau_0^2),$$

where $\mu_0$ and $\tau_0^2$ are known parameters.

The Gibbs sampler is well suited to the analysis of hierarchical models, since the full one-dimensional conditional distributions often have extremely simple forms. For example, in the above model,

$$\begin{aligned} \mu_k|\boldsymbol{\mu}_{-k}, \theta, \boldsymbol{\sigma}^2, \mathbf{y} &\sim \mathrm{N}\left(\frac{\sigma_k^{-2}\sum_{t=1}^{T} y_{kt} + \theta\sigma_0^{-2}}{T\sigma_k^{-2} + \sigma_0^{-2}}, \frac{1}{T\sigma_k^{-2} + \sigma_0^{-2}}\right) \\ \theta|\boldsymbol{\sigma}^2, \boldsymbol{\mu}, \mathbf{y} &\sim \mathrm{N}\left(\frac{\sigma_0^{-2}\sum_{k=1}^{K}\mu_k + \mu_0\tau_0^{-2}}{K\sigma_0^{-2} + \tau_0^{-2}}, \frac{1}{K\sigma_0^{-2} + \tau_0^{-2}}\right) \\ \sigma_k^{-2}|\boldsymbol{\sigma}_{-k}^2, \boldsymbol{\mu}, \theta, \mathbf{y} &\sim \Gamma\left(\alpha_0 + \frac{T}{2}, \beta_0 + \frac{1}{2}\sum_{t=1}^{T}(y_{kt} - \mu_k)^2\right). \end{aligned}$$

> **Question 2** Verify the one-dimensional conditional distributions given above. What is the marginal prior distribution of $\mu_k$?

**Question 3** Implement the Gibbs sampler to sample from the posterior distribution of $(\boldsymbol{\mu}, \boldsymbol{\sigma}^2, \theta)$ given $\mathbf{y}$. You can find the data for $\mathbf{y}$ in the file `II-10-9-2021football.csv` on the CATAM website. Take as known $\sigma_0 = 10$, $\alpha_0 = 10^{-5}$, $\beta_0 = 10^{-3}$, $\mu_0 = 60$, $\tau_0 = 20$. Briefly discuss what these prior parameter values have assumed about the football data. With reference to these priors, how did you choose the initial state of the Markov chain?

If you wish, you can use a package to simulate distributions, but you should implement the Gibbs sampler yourself without using library routines.

**Question 4** Use your Gibbs sampler to estimate the posterior mean of each parameter $\theta$, $\mu_k$, $\sigma_k^2$. Plot a histogram of the posterior distribution of $\theta$ and comment on your histogram. Explain how you obtained it.

**Question 5** Now choose a team $k$. Estimate the posterior probability that your chosen team is above average, $\mathbb{P}(\mu_k > \theta|\mathbf{y})$.

**Question 6** Build up an idea of how accurate your estimates for $\mu_k$ and $\mathbb{P}(\mu_k > \theta|\mathbf{y})$ are, for your chosen team $k$, by performing independent runs of the Gibbs sampler, computing estimates for each of the parameters on each run and then computing the sample variances of these estimates. Comment on how fast your estimates converge by considering sample variances at different values of $N$.

**Question 7** Now try letting the algorithm run for an initial period of $M$ cycles before calculating estimates based on a further $N$ iterations. This might allow the distribution to settle down to equilibrium before being measured. Calculate sample variances (as the previous question) for a few suitable values of $M$ to see if this makes any noticeable difference. Explain why you do or do not see a difference.

**Question 8** In any MCMC procedure we must ensure that we are exploring the full sample space. One way to check this is to run a number of chains that start from different points. Using a few widely dispersed starting points confirm, or otherwise, that your results are independent of the starting point. What is the effect of running an initial $M$ cycles in this situation?

**Hint.** Recall that the Gamma distribution $\Gamma(\gamma, \lambda)$ has density

$$f(x) = \frac{\lambda^\gamma x^{\gamma-1} e^{-\lambda x}}{\Gamma(\gamma)},$$

with mean $\gamma/\lambda$ and variance $\gamma/\lambda^2$. Also recall that a Gamma $\Gamma(n/2, \lambda)$ has the same distribution as the scaled chi-squared $(2\lambda)^{-1}\chi_n^2$.

You can assume that a $\Gamma(2.50001, \lambda)$ is approximately distributed as a $\chi_5^2$ (suitably scaled), which in turn is exactly equal to the sum of two independent exponentials plus an independent normal squared.

# 10 Statistics

## 10.16 The Tennis Modelling Challenge (8 units)

*This project requires concepts from the Part II course Statistical Modelling. The use of R is highly recommended, although Python is a viable alternative (see Programming notes at the end of the project).*

## 1 Introduction

In this project you shall analyse a dataset scraped from the website `tennis-data.co.uk`. The file `mensResults.csv` available from the CATAM website contains information about matches between the top male professional players in the period 2000–2016. Each row corresponds to a match and lists the following variables:

- `Winner`: The match's winner.
- `Loser`: The match's loser.
- `W1-W5`: The number of games won in sets 1-5 by the winner. If an entry is missing, the match ended before the corresponding set.
- `L1-L5`: Like `W1-W5`, but for the loser.
- `Surface`: The surface on which the match was played.
- `Series`: Name of ATP tennis series (Grand Slam, Masters, International or International Gold).
- `Tournament`: Name of tournament.
- `Round`: Round within the tournament.
- `Date`
- `B365W`: Betting odds on website Bet365 for the winner.
- `B365L`: Betting odds on website Bet365 for the loser.

## 2 Bradley–Terry model

The Bradley–Terry model for a tournament considers each match to be independent, and

$$\Pr(\text{Player } a \text{ wins a match against Player } b) = \frac{\exp(\beta_a - \beta_b)}{1 + \exp(\beta_a - \beta_b)}, \tag{1}$$

for a vector of parameters $\beta$ of equal length as the number of players. This model was first invented by the mathematician Ernst Zermelo in 1929 as a way to rank chess players.

**Question 1** The Bradley–Terry model is a Generalised Linear Model (GLM). Specify the exponential family distribution of the response, the link function, and the design matrix. What happens when we exchange the order of the players in Eq. (1)?

**Question 2** Fixing the coefficient for the player "Agassi A." at 0, obtain the maximum likelihood estimator for the coefficients $\beta$, using the data for the period 2000–2014. Does the choice of the player "Agassi A." for the reference class affect the fitted values? Report the logistic loss (the negative log-likelihood divided by the number of samples) in the training data, as well as test data containing all the matches in 2015 and 2016.

**Question 3** Obtain a 68% confidence interval for the probability that Roger Federer beats Andy Murray in a match.

It is well-known among tennis enthusiasts that certain players enjoy an advantage on specific surfaces. For example, Rafael Nadal does very well on the Roland–Garros tournament, which is played on clay, while Roger Federer has an excellent record in Wimbledon, played on grass. You suggest a new model with

$$\Pr(\text{Player } a \text{ wins a match against Player } b \text{ on surface } s) = \frac{\exp(\beta_a + \beta_{a,s} - \beta_b - \beta_{b,s})}{1 + \exp(\beta_a + \beta_{a,s} - \beta_b - \beta_{b,s})},$$

where $\beta_{a,s}$ can be interpreted as the advantage of player $a$ on surface $s$ compared to a baseline fitness $\beta_a$.

**Question 4** Describe an appropriate constraint to make the parameters of this model identifiable. Fit the model using the data from 2000–2014 and compare to the model fit in Question 2 using the logistic loss incurred on training and test data.

**Question 5** Perform a formal hypothesis test which might allow you to reject the model in Question 2 in favour of the alternative in Question 4. Can you reject the simpler model at the 1% level? Does this agree with the cross-validation comparison of the two models?

**Question 6** Discuss how you might use the variables `W1-W5` and `L1-L5` as an output in a GLM.

## 3 Regularisation

The *lasso* estimator $\hat{\beta}^{(\lambda)}$ for a GLM without an intercept solves the problem

$$\underset{\beta \in \mathbb{R}^p}{\text{minimise}} \; -\frac{1}{n}\mathcal{L}(\beta) + \lambda \|\beta\|_1, \tag{2}$$

where $\mathcal{L}(\beta)$ is the log-likelihood, $\lambda \geqslant 0$ and $\|\beta\|_1 = \sum_{j=1}^{p} |\beta_j|$. When $\lambda = 0$, $\hat{\beta}^{(0)}$ is the maximum likelihood estimator. When $\lambda > 0$, the second term penalises large values of the coefficients. While this biases the estimator toward 0, it can actually reduce the mean squared error, especially when there are many input variables.

This estimator is also convenient because it can be exactly 0 for a subset of the coefficients, which is a form of automatic variable selection. Increasing the parameter $\lambda$ produces estimators of increasing sparsity.

The penalty term in (2) is separable, i.e. it is a sum over the coefficients in the model. The function `glmnet` in R allows us to choose a vector $w \in \mathbb{R}^p$ through the argument `penalty.factor` in order to solve the problem

$$\underset{\beta \in \mathbb{R}^p}{\text{minimise}} \; -\frac{1}{n}\mathcal{L}(\beta) + \lambda \sum_{j=1}^{p} w_j |\beta_j|. \tag{3}$$

**Question 7** Using the same constraints applied previously, refit the model in Question 4 with a lasso penalty on the parameters $\beta_{a,s}$ for every player $a$ and surface $s$, but no penalty on the parameters $\beta_a$ for each player $a$. Fit the model to the data from 2000–2014 and use 10-fold cross validation to find the optimal value of $\lambda$. The function `glmnet` standardises the columns of the design matrix by default. Why might this make sense, in general? Does this make sense in your model? Explain why or why not and adjust the argument `standardize` accordingly.

**Question 8** With the optimal value of $\lambda$, how many of the estimates for coefficients $\beta_{a,s}$ are non-zero? Compute the logistic loss on the held-out data from 2015 and 2016 and compare to the maximum likelihood estimator in Question 4.

**Question 9** It is likely that the ability of each player drifts from year to year. Modify the model in Question 2 to take account of this fact and apply a lasso penalty of your choice. Plot the probability that Federer wins against Nadal as a function of time. Compare your proposal against previously considered alternatives in any reasonable way.

## 4 Can you outperform the betting market?

The dataset contains betting odds from the site Bet365. Each number is a ratio of the payoff, including the gambler's stake, to the stake. For example, suppose that in a match between Federer and Nadal which was won by Federer, the variable `B365W` equals 1.85 and the variable `B365L` equals 1.90. Then, betting £1 on the event that Federer won would have a payoff of £0.85 in addition to the gambler's stake, while betting £1 on the event that Nadal won would have a payoff of £0.90 in addition to the gambler's stake.

Suppose you have a budget of £1,000 to bet on $k$ tennis matches played in 2015, and you have access to data from 2000–2014. Let $\phi_{i,W}$ and $\phi_{i,L}$ be the amounts we bet on the winner and the loser in match $i$, respectively*. We call the vector $\phi = (\phi_{1,W}, \phi_{1,L}, \dots, \phi_{k,W}, \phi_{k,L})$ the *portfolio*.

A Markowitz portfolio aims to maximise the expected profits while minimising the risk, measured by the variance of the profit, under a model which specifies the probabilities of each outcome in each match. Letting $R_{i,W}, R_{i,L}$ be the profit made on the bet on the winner and loser of match $i$, and $R = \sum_{i=1}^{k}[R_{i,W} + R_{i,L}]$ be the total profit, we can formalise the problem as follows

$$\underset{\phi\in\mathbb{R}^{2k}}{\text{minimise}} \ -\mathbb{E}(R) + \nu \text{Var}(R) \tag{4}$$

$$\text{subject to } \phi_{i,W} \geqslant 0, \phi_{i,L} \geqslant 0 \text{ for } i = 1, \dots, k, \ \sum_{i=1}^{k}[\phi_{i,W} + \phi_{i,L}] = 1000.$$

**Question 10** Write this problem as a quadratic program with affine constraints

$$\underset{w\in\mathbb{R}^{2k}}{\text{minimise}} \ w^\top Q w - w^\top q$$
$$\text{subject to } w^\top \mathbf{1} = 1000, \ w_i \geqslant 0 \text{ for } i = 1, \dots, 2k.$$

for some positive semidefinite matrix $Q$ and a vector $q$. Provide an expression for $Q$ and $q$ in terms of the betting odds and the predictions of a logistic regression model.

*The winner and loser are not known *a priori*, but we use these labels as a way to distinguish the two players in each match.

> **Question 11** Using any software library for quadratic programming, find the optimal portfolio for $\nu \in \{10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}\}$ with the maximum likelihood predictions for the model in Question 4. Plot the profits which would have been obtained with this portfolio in 2015 against the parameter $\nu$. Comment on the results.

The variance of the profits $R$ in the definition of the Markowitz portfolio (4) assumes that the probability of each outcome for each match is known exactly. This does not take into account the uncertainty of the model parameters, which can increase a portfolio's risk. A *Bayesian Markowitz portfolio* assumes that the probabilities predicted for each match by the model are random and distributed according to some posterior distribution. The expectation and variance in Eq. (4) take into account this source of randomness.

> **Question 12** Prove that a Bayesian Markowitz portfolio is still a quadratic optimisation problem with affine constraints.

In reality bets are made sequentially, as we wouldn't know which matches are played in 2015 in advance. One approach to sequential betting is the *Kelly* criterion. Before match $i$, we bet fractions $\phi_{i,W}$ and $\phi_{i,L}$ of our current bankroll on the winner and loser of the match, which maximise the expectation of the logarithm of the bankroll after the bet. Here, $\phi_{i,W}, \phi_{i,L}$ are constrained to be non-negative and $\phi_{i,W} + \phi_{i,L} \leqslant 1$. The *fractional Kelly* strategy further reduces the risk by staking fractions $\rho\phi_{i,W}$ and $\rho\phi_{i,L}$ of the bankroll on each player, where $\rho$ is a constant smaller than 1.

> **Question 13** Write a function to compute the Kelly fractions numerically using a grid search. Evaluate the fractional Kelly strategy with $\rho = 0.1$ on the data from 2015 using the predictions of the model in Question 4. Plot the bankroll as a function of the date in 2015.

The results of this section should be rather surprising, as the betting market pools the opinions of many experts and the betting odds ensure that the bookmaker turns a small profit. It is possible to improve the statistical model significantly, using any of the variables provided for the period 2000–2014. **One way to obtain excellence marks in this project, but not the only way, is to try to improve upon the models outlined above and share your results[†].**

## 5 Programming notes

The use of R is strongly recommended. In particular, the package `glmnet` can be used to fit all the models in the project and may be installed through the following commands.

```
install.packages("glmnet")
library(glmnet)
```

The function `glmnet` fits the more general *elastic net* estimator, in which the penalty term in Eq. (2) is replaced by

$$\lambda \left\{ \frac{1-\alpha}{2} \|\beta\|_2^2 + \alpha \|\beta\|_1 \right\};$$

[†]You may also guard them as a trade secret.

setting $\alpha = 1$ specialises to the lasso. The function `cv.glmnet` may be used to optimise the parameter $\lambda$ through 10-fold cross validation. The documentation for these functions must be read carefully prior to the analysis. Refer to [1] for further guidance.

You may find `glmnet` useful to obtain maximum likelihood estimators in Section 2, as this function accepts design matrices which are sparse. Familiarity with the data type `sparseMatrix` and the function `sparse.model.matrix` will prove useful. In addition, the function `as.date` is useful to handle variables which indicate dates.

The function `solve.QP` in the `R` library `quadprog` may be used to solve the portfolio optimisation problem. When the matrix $Q$ is singular, an approximate solution may be obtained by adding weight to the diagonal elements, replacing $Q$ by $Q + 10^{-8}I$.

The Python library `Scikit-learn` also has methods to fit GLMs with a lasso penalty, and may be used together with the package `pandas` for manipulating data frames.

## References

[1] James, G., Witten, D., Hastie, T. and Tibshirani, R. *An Introduction to Statistical Learning with Applications in R.* Springer, New York, 2013. [`http://www-bcf.usc.edu/~gareth/ISL/ISLR%20First%20Printing.pdf`]

# 11 Statistical Physics

## 11.3 Classical gases with a microscopic thermometer (8 units)

*This project can be done with knowledge of the course Statistical Physics.*

## 1 Introduction

Consider a gas of $N$ non-interacting classical particles. The momentum of the $i$th particle is $\mathbf{p}_i$ and its kinetic energy is $E_i$. The energy $E_g$ of the gas is

$$E_g = \sum_{i=1}^{N} E_i \,.$$

To this system we add one additional degree of freedom, which acts as a thermometer. The thermometer stores energy, and can exchange it with the gas. The energy of the thermometer is $E_d$ and the total energy $E = E_g + E_d$ is conserved (we consider the microcanonical ensemble). We will show that measuring the average value of $E_d$ can be used to infer the temperature of different kinds of classical gas.

## 2 Algorithm

We use a stochastic (random) algorithm to calculate the statistical behaviour of this system. This is an example of a Monte Carlo algorithm. It operates as follows:

1. As an initial configuration, set $\mathbf{p}_i = \mathbf{e}_1$ for all $i$, where $\mathbf{e}_1$ is a unit vector in the $x$-direction. Initialise also $E_d = 0$.

2. Choose one of the $N$ particles at random and compute its *current* energy $E_{\text{curr}}$. Generate a random vector $\Delta\mathbf{p}$ and propose a change of the particle's momentum, from $\mathbf{p}_i$ to $\mathbf{p}_i+\Delta\mathbf{p}$. A good choice is to take each component of the vector $\Delta\mathbf{p}$ to be a random number from $(-\varepsilon, \varepsilon)$ with $\varepsilon = 0.1$. Compute the energy that the particle would have if its momentum was $\mathbf{p}_i + \Delta\mathbf{p}$: this is the *proposed* energy $E_{\text{prop}}$.

3. Define $\Delta E \equiv E_{\text{prop}} - E_{\text{curr}}$. If $\Delta E \leqslant E_d$ then accept the change. That is, update the momentum of particle $i$ to a new value $\mathbf{p}_i + \Delta\mathbf{p}$, and update $E_d$ to a new value $E_d - \Delta E$. If $\Delta E > E_d$ then the change is rejected and no variables are updated.

4. Whether or not the change was accepted, record the value of $E_d$ as a new value in an array (or list) which will later be used to plot a histogram. Also record the energy of the particle. (This is called the *single-particle energy.*) If the change was accepted, you should record these values *after* the update was performed.

5. Repeat steps 2-4 until the total number of attempted updates is $N_{\text{updates}}$. Since each update only affects one particle, it is useful to define $N_{\text{sweeps}} = N_{\text{updates}}/N$ so that $N_{\text{sweeps}}$ is the typical number of times that each particle has been chosen for an update.

**Question 1** In the microcanonical ensemble each microstate (of the whole system) is equally likely. For the thermometer, suppose that every possible value of $E_d$ corresponds to a single microstate. Hence explain why the probability distribution for $E_d$ behaves as

$$P(E_d) \propto \Omega_g(E_g)\,.$$

where $\Omega_g(E_g)$ gives the number of microstates of the gas.

**Question 2** The temperature of the gas is related to its entropy as

$$\frac{1}{T} = \frac{\partial S_g}{\partial E_g}\,.$$

Assuming that $E_d \ll E_g$, use this fact to show that

$$P(E_d) \propto \exp\left(-\frac{E_d}{k_\mathrm{B}T}\right)\,. \tag{1}$$

where $k_\mathrm{B}$ is Boltzmann's constant. [It is also acceptable to take $k_\mathrm{B} = 1$.]

## 3 Ideal gas

**Programming Task:** Write a program to simulate a gas of $N$ particles using the Monte Carlo algorithm outlined above. Consider a 3-dimensional gas of nonrelativistic particles, so $\mathbf{p} = (p_1, p_2, p_3)$ and

$$E(\mathbf{p}) = \frac{|\mathbf{p}|^2}{2}\,.$$

You will need to keep track of the momentum vectors for the $N$ particles in the gas. It will be useful in later questions if your program includes a function which returns the particle energy, given $\mathbf{p}$ as input.

You will also need to plot histograms of the quantities that were recorded in step 4 of the algorithm: the value of $E_d$ and the single-particle energy. Remember, a histogram is a graph of the relative frequency that a quantity such as $E_d$ lies within a particular bin. This relative frequency is $f(E_d)$.

Your program should also calculate the average of $E_d$.

Throughout this project, should compare your results with the behaviour that you would expect from the theory of statistical physics. The results should be presented in such a way that this comparison is clear.

**Question 3** For $N = 100$, plot a histogram of $E_d$ for $N_\text{sweeps} = 10, 100, 1000$. [You may wish to plot $\log f(E_d)$ instead of $f(E_d)$.] Your program should not take more than a few minutes to run. Discuss (and explain) the results, including the dependence on $N_\text{sweeps}$. Do the results depend on the parameter $\varepsilon$ that appears in step 2 of the algorithm?

**Question 4** If $N_\text{sweeps}$ is large enough, the system should be in an equilibrium state. For this case, compare the histogram of $E_d$ with Equation (1), and estimate the temperature of the gas. If the distribution of $E_d$ is consistent with (1), you can also estimate the temperature from the average of $E_d$. Quantify the numerical uncertainties on these two estimates of the temperature.

**Question 5** For the equilibrium state, plot a histogram of the single-particle energy. Show that the result is consistent with the theory of ideal gases from statistical physics.

**Programming Task:** Modify your program so that each particle is initialised with a randomly assigned momentum (instead of all starting with $\mathbf{p}_i = \mathbf{e}_1$). For example, assign each component of $\mathbf{p}_i$ independently at random from $(-a, a)$, with $a = 1$.
(Note: depending on $a$, you may want to change the parameter $\varepsilon$ that appears in step 2 of the algorithm.)

**Question 6** How does this change in initial conditions affect the histograms of $E_d$ and the single-particle energy? What happens for different values of $a$? How does the temperature depend on $a$? Explain your observations, including their consistency with the theory of ideal gases from statistical physics.
(Note: depending on $a$, you may want to change the parameter $\varepsilon$ that appears in step 2 of the algorithm.)

## 4 Relativistic gases

**Programming Task:** Continue with random initial conditions [each component of $\mathbf{p}_i$ chosen independently at random from $(-a, a)$]. Modify your program to consider ultra-relativistic particles that move in two dimensions: this means that $\mathbf{p}$ is a vector with two components and that

$$E = |\mathbf{p}| \ .$$

(For the purposes of statistical physics, we still refer to this system as a classical gas, because quantum mechanical effects have been neglected.)

**Question 7** For $a = 1$, compute and plot histograms of $E_d$ and of the single particle energy. Estimate the temperature of the gas. Vary $a$ and compute the temperature. Plot this temperature as a function of the total energy of the system. Compare the result with the case considered in question 5 (non-relativistic particles in three dimensions), and discuss their consistency with the theory of ideal gases from statistical physics.

**Programming Task:** Consider relativistic particles in three dimensions so that $\mathbf{p}$ is a vector with three components, and

$$E(\mathbf{p}) = \sqrt{1 + |\mathbf{p}|^2} - 1 \ .$$

**Question 8** Consider different values of the total energy by varying $a$ in the range 0.1 to 2.0. How does the temperature depend on the total energy? By considering the behaviour of $E(\mathbf{p})$ for large and small values of $|\mathbf{p}|$, comment on the relation of this result to the cases from previous questions. Compare the histograms of single-particle energies for a few representative cases.

# 12 Nonlinear Dynamics/Dynamical Systems

## 12.3 The Lorenz Equations (10 units)

*Some familiarity with the Part II course Dynamical Systems would be helpful for this project, which is concerned with bifurcations and chaos in ordinary differential equations.*

## 1 The Lorenz equations

The Lorenz equations are named after the meteorologist who first studied them in 1963:

$$\begin{aligned} \dot{x}(t) &= f_1(x,y,z) &&= 10(y-x) \ , \\ \dot{y}(t) &= f_2(x,y,z) &&= rx - y - xz \ , \\ \dot{z}(t) &= f_3(x,y,z) &&= xy - 8z/3 \ . \end{aligned}$$

**Question 1** Integrate the equations for values of $r = 0$, 15, 21 and 29. Use $x = y = 1$, $z = r - 1$ as the initial conditions. You may use any standard integrating packages that are available and enable you to choose an appropriate step-length and then fix on it*; comment however on the effect of changing the step-length and why you chose your particular value. You should plot $x(t)$ against $z(t)$ to show your results. For some of the above values of $r$ consider plotting the solution only for $t > T$ for some time $T > 0$; why can this be useful?

A stationary point is a point $(x, y, z)$ where $\dot{x} = \dot{y} = \dot{z} = 0$. It is (locally) stable if all the eigenvalues of the Jacobian matrix $Df(x) = (\partial f_i/\partial x_j)_{1\leqslant i,j\leqslant 3}$ evaluated at the stationary point have negative real part.)

**Question 2** Investigate analytically the existence and stability of stationary points of the flow. How do these results relate to the behaviour observed in question 1 above?

## 2 The strange attractor

The persistent erratic non-periodic oscillations seen when $r = 27$ are due to the existence of a "strange attractor" in the flow. (The existence of this attractor was discovered numerically by Lorenz but there is still no completely rigorous proof that it exists and has the properties we are about to study). This attractor is stable for (approximately) $r > 24.06$, but for $r < 24.06$ some solutions spend a long time wandering about near it before eventually tending towards a stable stationary point. (It exists, but is unstable, for approximately $13.9236 < r < 24.06$.) This phenomenon is known as *intermittency.*

**Question 3** For various initial conditions as given in the following list, plot $x(t)$ against $t$ at $r$-values of your choice in $23 < r < 25$: in each case include in your write-up one or two plots showing the different possible behaviours.

(i) Start very close to the origin $(0, 0, 0)$ but not on the $z$-axis (why not?).

(ii) Start very close to one of the other fixed points.

*For example you can download a suitable solver from http://www.mathworks.com/matlabcentral/answers/98293

(iii) Start near $x = y = 1$, $z = r - 1$.

Which type of initial condition is best suited for deciding the $r$-value at which the strange attractor becomes attracting? Which is useful to confirm your stability analysis for the stationary points obtained in question 2 above? Which best illustrates intermittency?

**Question 4** For initial conditions which produce trajectories displaying intermittent behaviour in $r < 24.06$, plot the time spent wandering erratically before the trajectory spirals steadily into one of the stationary points against $(24.06 - r)$. You should decide on some criterion for deciding the time $t_c$ at which the solution you are calculating starts heading towards a stable stationary point, and calculate the average of the values of $t_c$ obtained for 5 different initial conditions (all of which should display several "erratic" oscillations before $t_c$ is reached) at each $r$-value. Explain how you determine $t_c$.

You will find that nearby initial conditions sometimes give very different values of $t_c$, and as $r$ increases towards 24.06 you may find that it becomes increasingly difficult to find $t_c$ for all of your chosen initial conditions; you should start with $r = 20$ and be prepared to stop increasing $r$ when the amount of machine time used becomes excessive.

**Question 5** Suggest a formula for the way in which the average $t_c$ value increases with $r$. You will need a fairly large sample of $t_c$ values to make a reasonable estimate.

**Question 6** For $r = 27$, write a program to record the successive $z$-values $z_1$, $z_2$, $z_3$, ... at which a trajectory achieves a local maximum in $z$. Plot these on a scatter diagram of $z_{n+1}$ against $z_n$; include also for reference the diagonal line $z_{n+1} = z_n$. What property do portions of the trajectory which generate high points (large values of $z_{n+1}$) on this diagram have? Does the information that the origin $(0, 0, 0)$ is actually part of the strange attractor help you to find a numerical method to compute (approximately) the largest value of $z_{n+1}$ that could appear on this diagram? If so, do it and add an appropriate point to your figure.

You should not plot the first few points obtained from any given trajectory in order to give any transient behaviour time to die out. You may generate points from many trajectories or from one long trajectory. You will observe that the points on this scatter diagram all lie very near to a certain curve $C$, which can therefore be used as a predictor for the successive $z_i$ values.

**Question 7** Describe in some detail the chief features of this curve and how they relate to your numerical solutions. In particular, consider the following points:

- Where are the intersections of $C$ with the diagonal?
- Does $C$ intersect the diagonal at $z = r - 1$?
- On your diagram, is it possible to draw a square whose top-right and bottom-left corners lie on the diagonal, whose top side touches the peak of $C$, whose bottom-right corner lies on $C$, but whose bottom edge does not otherwise intersect $C$?

**Question 8** On a copy of your diagram, draw an approximation to the curve $C$ and use this hand-drawn curve (which you should include in your write-up) to predict a succession of $z_i$ values. For how many steps does your prediction agree well with an actual sequence produced by the numerically computed trajectory? Are there any features of the curve which would lead you to expect this result?

## 3 The effect of varying $r$

**Question 9** How does the curve $C$ vary as $r$ decreases? Draw the curves for $r = 24.3$ and 22.9, extending them in a sensible way to $z = r - 1$. (For $r = 22.9$ you will need to use initial conditions which give intermittent trajectories in order to generate much of the curve). Describe how the features of the curve change, and explain how these changes relate to the other aspects of behaviour studied in this project.

## References

[1] Colin Sparrow *The Lorenz equations: bifurcations, chaos, and strange attractors.* Springer, 1982.

# 12 Nonlinear Dynamics & Dynamical Systems

## 12.6 Chaos and Shadowing (10 units)

*Familiarity with the Part II Dynamical Systems course would be very helpful for this project, which is concerned with the behaviour of nonlinear maps and uses concepts and tools from nonlinear dynamics.*

## 1 Introduction: dynamical systems, chaos and shadowing

This project considers issues that arise in the numerical solution of dynamical systems which display complicated 'chaotic' behaviour. We first consider the discrete-time case, defining Lyapunov exponents which measure the rate at which nearby points separate under iteration. Then we discuss how 'noisy' trajectories of an iterated map, where the 'noise' arises through numerical errors, are actually close to true trajectories of the system - this property is known as 'shadowing'. Finally the project considers a continous-time (ODE) example of complicated motion motivated by celestial mechanics.

Let $D$ be a closed bounded subset of $\mathbb{R}^m$, and let $F(\mathbf{x})$ be a continuously differentiable map from $D$ to itself. A major task in dynamical systems is to characterise the behaviour of points under repeated iteration of the map $F$. We call the sequence of points $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \ldots$ constructed by setting $\mathbf{x}_{n+1} = F(\mathbf{x}_n)$ the *trajectory* from the initial condition $\mathbf{x}_0$. The standard notation for the repeated composition of $F$ is to let $F^n$ denote the $n$-fold composition of $F$ with itself, i.e. $\mathbf{x}_n = F(\mathbf{x}_{n-1}) = F^2(\mathbf{x}_{n-2}) = \cdots = F^n(\mathbf{x}_0)$.

In many situations the rate at which nearby trajectories separate from each other is of interest. This can be characterised by the Lyapunov exponents $\lambda(\mathbf{x}_0, \mathbf{v})$, defined to be the asymptotic rate of divergence of trajectories with initial conditions $\mathbf{x}_0$ and $\mathbf{x}_0 + \mathbf{v}$, where $\mathbf{v}$ is a small perturbation from $\mathbf{x}_0$:

$$\lambda(\mathbf{x}_0, \mathbf{v}) = \lim_{n\to\infty} \lim_{\epsilon\to 0} \frac{1}{n} \log \frac{\|F^n(\mathbf{x}_0 + \epsilon\mathbf{v}) - F^n(\mathbf{x}_0)\|}{\|\epsilon\mathbf{v}\|} \tag{1}$$

Under the conditions given above it can be shown that the limit exists. For a given $\mathbf{x}_0$ there will in general be $m$ (possibly non-distinct) values of $\lambda(\mathbf{x}_0, \mathbf{v})$ as we choose different vectors $\mathbf{v}$ – divergence occurs at different rates in directions corresponding to the different eigenvectors of the Jacobian matrix $DF$ evaluated at $\mathbf{x}_0$. The formula above for $\lambda(\mathbf{x}_0, \mathbf{v})$ will give the largest positive Lyapunov exponent of the system for almost all choices of the vector $\mathbf{v}$. We denote the largest positive Lyapunov exponent by $\lambda(\mathbf{x}_0)$, or simply by $\lambda$. If $\mathbf{x}_0$ is a fixed point then the Lyapunov exponents are simply the (real parts of the) Floquet multipliers, so in some sense the idea of a Lyapunov exponent developed above is a generalisation of the idea of a Floquet multiplier to arbitrary trajectories.

For the purposes of this project we will define a map to be chaotic if it appears that $\lambda(\mathbf{x}_0) > 0$ for almost all $\mathbf{x}_0$, so that in general neighbouring points will separate exponentially.

## 1.1 A map on a square

Here we consider a 2-dimensional (area preserving) map on the unit square $(x, y) \in [0, 1]^2$. Given some initial condition $(x_0, y_0)$, we define

$$x_{n+1} = x_n + \frac{K}{2\pi}\sin(2\pi y_n) \pmod 1 \tag{2}$$

$$y_{n+1} = y_n + x_{n+1} \pmod 1 \tag{3}$$

Note that here(mod 1) means that the map is restricted to the unit square.

**Question 1** For $K = 3$, generate a set of double precision pairs $(x_n, y_n)$, $1 \leqslant n \leqslant 1000$, for a range of choices of $(x_0, y_0)$. Some suggested choices of initial conditions are : (0.5,0.5); ($10^{-8}$, 0); (0.1,0.5); (0.8,0.6); (0.3521,0.424).

Plot the distribution of your points $(x_n, y_n)$. Describe the structure of the phase portrait. What are the fixed points of the map? Describe how the different features of the map change with K.

## 1.2 Local Chaos

In contrast to the asymptotic quantity $\lambda(\mathbf{x}_0)$ as defined above, a possibly more useful quantity is the local Lyapunov exponent, $\lambda_l(\mathbf{x}_0)$, defined as

$$\lambda_l(\mathbf{x}_0) = \lim_{\boldsymbol{\Delta}\to\mathbf{0}} \frac{1}{N}\sum_{n=0}^{N} \log \frac{\|\mathbf{x}_{n+1} - \mathbf{x}_{n+1}^{(\boldsymbol{\Delta})}\|}{\|\mathbf{x}_n - \mathbf{x}_n^{(\boldsymbol{\Delta})}\|} \tag{4}$$

where $\mathbf{x}_n$ is the nth iterate of $\mathbf{x}_0$, $\mathbf{x}_n^{(\boldsymbol{\Delta})}$ is the nth iterate of $\mathbf{x}_0 + \boldsymbol{\Delta}$, and $N$ is a suitable finite number of iterations of the map. For infinitesimal perturbations $\boldsymbol{\Delta}$, $\lambda_l(\mathbf{x}_0) > 0$ indicates a local expansion of trajectories starting near $\mathbf{x}_0$.

Note that $N$ should be chosen neither too small, nor too large. In practice, you might also want to discard the first few terms of this sum in your numerical calculations.

**Question 2** For $K = 3$, find the maximum local Lyapunov exponent for different initial conditions $(x_0, y_0)$ and small perturbation $\|\boldsymbol{\Delta}\| \ll 1$, using the Euclidean norm in equation (4). What value of $N$ did you use? How did you decide? What is your estimate of the global maximum Lyapunov exponent?

**Question 3** We can define a different "Lyapunov exponent", $\lambda_2 = \log_2 e^{\lambda_l}$. Why, when doing binary arithmetic, might $\lambda_2$ be more interesting than $\lambda$? What is your interpretation of the information $\lambda_2$ provides? Given that the calculations here are done to 16 significant figures (or to whatever precision achieved by the code you have used), what would you expect the number of iterations required to be before the results obtained become meaningless? How does your answer compare with what you found numerically?

## 1.3 Shadowing

Numerical calculations introduce round-off and truncation errors into iteration. For chaotic maps, such as the 2D map above this introduces an effective error at each iteration; this is in some sense equivalent to the explicit perturbation in the initial conditions we considered above.

Since a large class of interesting problems is reducible to iterating nonlinear, and chaotic, maps, it is of some interest to consider whether any numerical calculation can be said to follow the "true" trajectory of such systems.

Here we will consider the simple example used above, assuming the "true" trajectory is given by a double precision calculation of the trajectory, while a single precision calculation provides a "noisy" trajectory.

For some nonlinear systems it is possible to define a "shadow" trajectory to a noisy trajectory (obtained by adding a small perturbation), such that the shadow trajectory is a "true" trajectory of the system, and the "shadow distance" (initially the perturbation) of the shadow trajectory from the noisy trajectory is bounded ([1], [2], [5]). In 2D when there exists one *unstable* (expanding) direction and one *stable* (contracting) direction, it has been proved that, for sufficiently small perturbations, "shadow" trajectories can exist for arbitrarily long times. In many other systems it is still possible to define a "shadow" trajectory for a finite time.

**Question 4** Let $L_n$ be the Jacobian matrix of the map at iteration $n$. Construct and write down explicitly the four elements of the Jacobian matrix of the standard map above.

Define $e_{n+1} = p_{n+1} - f(p_n)$, where $e_n$ is the error iterating the map, $f$, on the vector $p$ by one step. We want to construct a correction term, $\Phi_n$, such that

$$\tilde{p}_n = p_n + \Phi_n \tag{5}$$

defines a "shadow" orbit of $p$, i.e. $\{\tilde{p}_n\}$ is a true orbit of the dynamical system.

Solving for $\Phi$, we find:

$$\Phi_{n+1} = f(\tilde{p}_n) - e_{n+1} - f(p_n). \tag{6}$$

For $\Phi_n$ small, we can expand $f(\tilde{p}_n)$ to linear order, and

$$\Phi_{n+1} = L_n\Phi_n - e_{n+1}. \tag{7}$$

At each iteration, small perturbations along the contracting direction will decay exponentially forward in time, while small perturbations in the expanding direction will grow exponentially forward in time. The reverse will happen when evolving backwards in time.

We therefore want to find basis vectors $u_n$, $s_n$ aligned with the directions defining the maximum expansion and contraction of the local volume of phase space at step $n$. We can construct $u_n$, $s_n$ by iterating the equations:

$$u_{n+1} = \frac{L_n u_n}{||L_n u_n||} \tag{8}$$

and

$$s_{n+1} = \frac{L_n s_n}{||L_n s_n||}. \tag{9}$$

That is, take some initial $u_0, s_0$ (eg. $(1/\sqrt{2}, 1/\sqrt{2})$, $(-1/\sqrt{2}, 1/\sqrt{2})$), and a vector $p_0 = (x_0, y_0)$. Iterate $u_n$ forwards, i.e. start with your $u_0$ and iterate equation (8) forward until it has converged to the local direction of expansion. To construct $s_n$, do the same, but take the initial $s_N$ for some finite (not too big nor too small) number of iterations of the map, and iterate $s_n$ backwards to find $s_0$. You will want $N \gg 1$ and $N \ll N_c$, where $N_c$ was the number of iterations at which the sum in equation (4) needed to be truncated.

This procedure will naturally converge onto the direction of maximum expansion when going forward in time, because the term corresponding to the maximum eigenvalue will become dominant for sufficiently large $n$. Conversely, when going backwards in time the term corresponding to the smaller eigenvalue will become dominant, because it will be proportional to the inverse of a small quantity.

Clearly, since $u_n$, $s_n$ span the phase space, we can write

$$\Phi_n = \alpha_n u_n + \beta_n s_n \tag{10}$$

and

$$e_n = \eta_n u_n + \xi_n s_n \tag{11}$$

for some $\alpha, \beta, \eta, \xi$.

Using equation (7) we find

$$\alpha_{n+1} u_{n+1} + \beta_{n+1} s_{n+1} = L_n(\alpha_n u_n + \beta_n s_n) - (\eta_{n+1} u_{n+1} + \xi_{n+1} s_{n+1}). \tag{12}$$

**Question 5** Substitute equations (8) and (9) into equation (12) to find a recursion relation for $\alpha_n$, $\beta_n$. As before, solve for the $\alpha_n$ by forward iteration from $n = 0$ and for the $\beta_n$ by backward iteration from $n = N$ for some suitable, fixed $N$.

You now have a constructed shadow map of the trajectory.

**Question 6** Integrating the standard map in double precision, from some known initial condition, with a known error, show that the shadow map of the erroneous initial conditions follows the true trajectory within some shadow distance. If necessary, iterate the shadowing to get a more closely shadowed orbit.

Now integrate your initial condition with single precision (introducing some, in principle unknown) error per iteration, and construct the corresponding double precision shadow trajectory.

Plot your trajectories and comment. (Finding and plotting such trajectories can be tricky!)

*Note that shadowing does not always work. A trivial counter-example is provided by the one dimensional logistic map $f(x) = 1 - 2x^2$, $x \in (-1, 1)$.*

*Near $x = 0$, no true orbit can shadow general noisy orbits, as noise in $f(x)$ may take the map out of the domain and iterating the subsequent trajectory will take $x$ to $-\infty$.*

## 1.4 Application: the Sitnikov Problem

It is known that the $N$-body problem, of $N > 2$ bodies moving under their own mutual gravitational attraction only, is chaotic.

Here we consider a well known special case of the restricted three-body problem, where one of the bodies has zero mass. In this particular problem, known as the Sitnikov problem ([4], [3]), the motion of the zero mass is restricted to the $z$ axis, defined by the normal to the plane of motion of the two massive bodies, through the center of mass. The two massive bodies move on Keplerian ellipses with eccentricity $\epsilon \in [0, 1]$ around their centre of mass.

Without loss of generality, we consider the two massive bodies to have masses, $M_1 = M_2 = 1/2$. We are interested in bound motion, with semi-major axis $a = 1$.

The motion of the two massive bodies is uniquely described by their elliptic orbit (the phase is irrelevant to the dynamics we are interested in, by rotational symmetry). The separation of the massive bodies from the center of mass is $r(t) = (1 - \epsilon \cos t) + O(\epsilon^2)$, .

We want to consider the motion of the third, zero mass body on the $z$-axis. Define $v = dz/dt$, then

$$\frac{dv}{dt} = -\frac{z}{(z^2+1)^{3/2}} - \frac{3z\epsilon\cos(t+t_0)}{(z^2+1)^{5/2}}. \tag{13}$$

The equations of motion may be integrated numerically using a high order integrator, such as the Runge–Kutta scheme, given some initial conditions. Without loss of generality, we choose initial conditions $t_0 = 0, z(0) = 0,\ v(0) = v_0$.

**Question 7** Write down the energy of the third mass, i.e. $\lim_{m\to 0}(E/m)$ and solve for the critical velocity, $v_c$, for which the energy is zero. Write down $z(t)$ for $\epsilon = 0$. Write down the Jacobian matrix of this map.

It is useful to define the initial velocity as some multiple of $v_c$. We are interested in (initially) bound motion, so $v_0 \leqslant v_c$.

For $\epsilon = 0.03, 0.04, 0.05$ and $v_0/v_c = 0.92, 0.94, 0.96$, plot $z(t)$ vs $t$. Comment.

In continuous time we can define a (maximum) Lyapunov exponent exactly analogous to the discrete-time case:

$$\lambda(\mathbf{z}_0) = \lim_{t\to\infty}\lim_{\epsilon\to 0}\frac{1}{t}\log\frac{\|\phi_t(\mathbf{z}_0+\epsilon\mathbf{w}) - \phi_t(\mathbf{z}_0)\|}{\|\epsilon\mathbf{w}\|} \tag{14}$$

for almost all choices of perturbation $\mathbf{w}$, where $\mathbf{z} = (z, \dot{z})$ and $\phi_t$ denotes the evolution operator defined by integrating the ODEs forwards in time.

**Question 8** As before, construct a trajectory in $(z, \dot{z})$ space with an initial "error", $\delta$, and integrate the true and erroneous trajectories for a chosen value of $v_0 \approx 0.95 v_c$.

Estimate numerically the Lyapunov exponent of the mapping for the different cases. Is the motion chaotic?

**Question 9** Using the method discussed in the previous section, construct a shadow trajectory for the zero mass body, and compare "true" trajectories integrated with double precision arithmetic, with the corresponding "shadow" trajectories integrated from the same initial condition with single precision arithmetic.

Comment on the integrability of the $N$-body problem. Do you think numerical integrations of $N$-body systems are reliable – or can be made reliable – in some sense?

## References

[1] Bowen, R, 1975 *J. Diff. Eqns.*, 18, 333.

[2] Grebogi, C., Hammel, S.M., Yorke, J.A., Sauer, T., 1990 *Phys. Rev. Lett.*, 65, 1527

[3] Liu, J., Sun, Y.-S., 1990 *Cel. Mech. and Dyn. Astro.*, 49, 285.

[4] Marchal, C., 1990 *The Three-Body Problem*, Elsevier (Oxford).

[5] Quinlan, G.D., Tremaine, S., 1992 *MNRAS*, 259, 505.

# 13 Logic and Computation

## 13.1 Minimisation of Deterministic Finite-State Automata (10 units)

*This project requires an understanding of the Part II course* Automata and Formal Languages*.*

### 1 The DFA Minimisation Algorithm

A *deterministic finite-state automaton* (DFA) is one of the simplest computing machines, consisting of a finite number of states and bounded read/write memory. Such machines are very limited in the things they can compute. However, the tasks they do compute are performed extremely fast, like integer reductions mod $m$.

Associated to each DFA is a unique minimal state DFA; the unique one (up to renumbering of states) which accepts the same language and has the smallest possible number of states. Thus, every regular language can be represented by a unique minimal state DFA. From hereon, an $(n, k)$-DFA is one with $n$ states and an alphabet of size $k$.

There are many well-known algorithms that, on input of any DFA, produce the minimal DFA associated to it. We suggest that in this project you use *Hopcroft's table-filling algorithm*, as detailed in [1] or in the lecture notes. The purpose of this project is to count all the minimal $(n, k)$-DFAs for sufficiently small $(n, k)$, and to investigate which of these define finite languages.

There are many symmetry properties of DFAs you can appeal to when going through this project. These might make your programs run faster and more efficiently (and in some cases, remove the need for unnecessary additional code). Moreover, you may find that you can verify *some* of the small cases of what you are asked to compute, by hand, which might be useful to do to check that your programs are running correctly.

### 2 Conventions

Without loss of generality, we will assume that our states are always labelled by integers, and that any DFA with $n$ states has these labelled by $\{1, \ldots, n\}$. We will also always assume that the state labelled 1 is the start state. Finally, we will assume that any DFA on $k$ letters has alphabet $\{1, \ldots, k\}$. We will keep all these conventions throughout the project.

An $(n, k)$-transition table is a transition table for an $(n, k)$-DFA (recalling all the conventions set out above).

Before you begin this project, you will need to establish a way to store transition tables. As we have adopted the convention that state 1 is the start state, this does not need to be reflected in the table. One way to store such tables could be as a matrix (indeed, MATLAB uses matrices as one of its primary data structures). You can use an $n \times (k + 1)$ matrix to represent an $(n, k)$-transition table, with the final column consisting of 1's and 0's to denote which states are accepting or non-accepting. Conventions vary, but in this project the start state can be an accept state.

## 3 Counting DFAs

The number of $(n, k)$-DFAs can be vast, even for relatively small $k$ and $n$. However, the number of unique *languages* defined by these is much less than the total number of possible DFAs.

> **Question 1** Give a closed-form expression for the number of DFAs with $n$ states and alphabet of size $k$. Present this as a $6 \times 4$ table of values (in exponent form, to 2 significant figures) for $1 \leqslant n \leqslant 6$ and $1 \leqslant k \leqslant 4$.

## 4 Accessible states

A state in a DFA is said to be *accessible* if it can be reached by starting at the start state and following a path in the transition diagram labelled by some word $w$. Otherwise, it is *inaccessible.* Clearly, a minimal DFA can have no inaccessible states.

In all subsequent questions, an *arithmetic operation* can be taken to be the addition, subtraction, multiplication or division of two integers; or the reading or re-writing of a matrix entry.

> **Question 2** Write a program to determine the set of accessible states from any arbitrary transition table. Run your program on the DFAs given by `Table1, Table2, Table3` and `Table4` in the file `DataTables` on the CATAM website. What is the complexity of your algorithm for an arbitrary $(n, k)$-transition table, in terms of $n$ and $k$? Compute the number of $(n, k)$-DFAs with no inaccessible states, for $k = 2$ and $n = 1, 2, 3, 4$.

## 5 The Table-Filling Algorithm

Two states $p, q$ of a DFA are said to be *equivalent* if, for every word $w$, if we follow the two (unique) paths in the transition diagram from $p$ and from $q$ labelled by $w$, then either both end at accepting states or both end at non-accepting states. Hopcroft's table-filling algorithm is a quick and efficient way to determine which states of a DFA are equivalent.

> **Question 3** Write a program to implement the table-filling algorithm on any arbitrary transition table and then produce a transition table for a minimal DFA defining the same language. At some point you may need to use part of your program from Question 2 to remove inaccessible states. Run your program on the DFAs given by `Table1, Table2` and `Table3` in the file `DataTables` on the CATAM website. What is the complexity of your algorithm for an arbitrary $(n, k)$-transition table, in terms of $n$ and $k$?

## 6 Finding all languages describable by an $(n, k)$-DFA.

It is often useful in mathematics to compute and analyse some special cases of a mathematical object, to gain some intuition for how the object behaves as a whole. In this project, we will try and understand properties of minimal DFAs and the language they define, by looking at *all* the minimal $(n, k)$-DFAs for small $n$ and $k$. This section will require you to use the programs developed in the previous sections as sub-routines.

**Question 4** Compute the number of distinct regular languages definable by a minimal $(n, 2)$-DFA, for each of $n = 1, 2, 3, 4$. Remember that there may be several transition tables which define equivalent DFAs (and thus define the same language), so you will need to account for this.

## 7 Finite languages

It is immediate that every finite language is regular. We will now investigate which of the minimal DFAs computed in previous questions define finite languages.

**Question 5** Let $D$ be a DFA with $n$ states. Prove that $\mathcal{L}(D)$ is infinite if and only if $\mathcal{L}(D)$ contains a word $w$ of length $n \leqslant |w| \leqslant 2n - 1$.

**Question 6** Write a program to determine if an arbitrary transition table defines a finite language, and if so, outputs the size of the language it defines. Run your program on the DFAs given by `Table1, Table2` and `Table3` in the file `DataTables` on the CATAM website. It may help to use your program from Question 3 to minimise your DFAs first, or you may find your algorithm takes a long time to run.

From Question 4 you should have a list (possibly with some repetition) of all the minimal $(n, 2)$-transition tables, for $n = 1, 2, 3, 4$. You should make use of this list in the remaining questions.

**Question 7** Let $f(n, k, s)$ be the number of regular languages of size $s$ definable by a minimal $(n, k)$-DFA. For $n = 1, 2, 3, 4$, compute all values of $f(n, 2, s)$ for $0 \leqslant s \leqslant 2^n - 1$. What about when $s \geqslant 2^n$?

**Question 8** For $n = 1, 2, 3, 4$, write out all minimal $(n, 2)$-transition tables (only *one* for each equivalence class) which give the finite languages of size 1 and of size $2^{n-1} - 1$ (if any exist). Write out the language each of these defines.

**Question 9** For $n \geqslant 1$, give closed-form expressions in terms of $n$ for $f(n, 2, 1)$ and $f(n, 2, 2^{n-1} - 1)$. Prove your expressions hold. What about $f(n, 2, s)$ when $2^{n-1} \leqslant s \leqslant 2^n - 1$?

## References

[1] J.E. Hopcroft, R. Motwani and J.D. Ullman, *Introduction to automata theory, languages and computation* (Chapters 2–4), 2nd edn, Addison-Wesley, 2001.

# 14 General Relativity

## 14.5 Cosmological distances (8 units)

*Although this project is based on general relativistic cosmology, no detailed knowledge of General Relativity is required. All relevant equations are defined and explained in the project itself.*

## 1 Introduction

In cosmology there are many ways to specify the distance between two points because, in the expanding Universe, the distances between objects are changing and Earth-bound observers look back in time as they look out in distance. All these distances measure the separation between events on radial null trajectories, trajectories of photons which terminate at the observer.

The metric for a homogeneous isotropic universe, in spherical polar coordinates for the spatial part, is

$$ds^2 = c^2dt^2 - R(t)^2\left[\frac{dr^2}{1-kr^2} + r^2(d\theta^2 + \sin^2\theta d\phi^2)\right], \tag{1}$$

where, by a suitable choice of radial coordinate $r$, $k = -1$, 0 or 1 for open, Euclidean or closed geometries.

In this metric the redshift relative to an observer at the spatial origin is given by

$$1 + z = R(t_0)/R(t_1), \tag{2}$$

where $t_0$ is the coordinate time at which the photon is received and $t_1$ that at which it was emitted. Thus, for a given observer, the redshift depends only on the radial scale factor of the Universe at the time the photon was emitted divided by its value at the observer's time. The redshift is important because it can be measured easily from the observed wavelengths of atomic transition lines with known rest wavelengths.

When the matter density at time $t$ is $\rho$ and the pressure is zero one of the Einstein field equations with the cosmological term becomes

$$\frac{\dot{R}^2}{R^2} + \frac{kc^2}{R^2} - \frac{\Lambda c^2}{3} = \frac{8\pi G}{3}\rho, \tag{3}$$

where $\dot{R} = \frac{dR}{dt}$ and $\Lambda$ is a constant. The other field equation can be combined with this to give the conservation of matter equation

$$\rho R^3 = \text{const.} \tag{4}$$

For small distances the redshift $cz = H_0 d$, where $d$ is the distance to the source. Then $H_0$, the Hubble constant, gives the local expansion rate. It is often written in the form $H_0 = 100h\,\text{km}\,\text{s}^{-1}\,\text{Mpc}^{-1} = 3.2409 \times 10^{-18}h\,\text{s}^{-1}$, where $h$ is dimensionless. The actual value of $h$ is still uncertain, and hotly debated, but most would agree on measurements of $0.72 \pm 0.08$. The megaparsec is an astronomical length unit appropriate for separations between galaxies, $1\,\text{Mpc} = 3.0856 \times 10^{22}\,\text{m}$. The Hubble time $t_\text{H} = 1/H_0 = 3.0856 \times 10^{17}h^{-1}\,\text{s}$ and the Hubble distance $D_\text{H} = c/H_0 = 9.26 \times 10^{25}h^{-1}\,\text{m}$. Take the number of seconds in one year to be $3.1556926\times10^7$ s.

Our Universe can be described by two parameters, the matter density now $\rho_0$ and the cosmological constant $\Lambda$, and we can express these in a dimensionless form using $H_0$ as

$$\Omega_\text{m} \equiv \frac{8\pi G\rho_0}{3H_0^2} \tag{5}$$

and

$$\Omega_\Lambda \equiv \frac{\Lambda c^2}{3H_0^2}. \tag{6}$$

By means of equation (3), at time $t_0$, the curvature value $k$ can be parameterised by $\Omega_k$ so that

$$\Omega_\mathrm{m} + \Omega_\Lambda + \Omega_k = 1. \tag{7}$$

Then the function

$$E(z) = \sqrt{\Omega_\mathrm{m}(1+z)^3 + \Omega_k(1+z)^2 + \Omega_\Lambda}$$

is proportional to the time derivative of the logarithm of the scale factor, $\dot{R}/R$, at redshift $z$ (see *e.g.* Peebles 1993, pp 310 – 321).

Where specific values are required in what follows you should take $H_0 = 72\,\mathrm{km\,s^{-1}\,Mpc^{-1}}$.

## 2 Lookback Time

The lookback time $t_\mathrm{L}$ is the difference between the age $t_0$ of the Universe now and the age $t_\mathrm{e}$ when the photons were emitted

$$t_\mathrm{L} = t_\mathrm{H} \int_0^z \frac{dz'}{(1+z')E(z')}. \tag{8}$$

**Question 1** If $\Omega_\mathrm{m} = 1$ and $\Omega_\Lambda = 0$, obtain an expression for the lookback time to an object with redshift $z$ and show that the age of the Universe is $t_\mathrm{L}(z=\infty) = \frac{2}{3}t_\mathrm{H}$.

**Question 2** Write a program to determine the lookback time in Gyr for general $H_0$, $\Omega_\mathrm{m}$ and $\Omega_\Lambda$. If $H_0 = 72\,\mathrm{km\,s^{-1}\,Mpc^{-1}}$, tabulate the lookback time to $z$ = 0.1, 1.0, 2.0, 4.0 and 6.7 (one of the highest individual object redshifts measured so far) for

(1) an Einstein-de-Sitter universe $\Omega_\mathrm{m} = 1$, $\Omega_\Lambda = 0$,

(2) a classical closed universe $\Omega_\mathrm{m} = 2$, $\Omega_\Lambda = 0$,

(3) a baryon dominated low density universe $\Omega_\mathrm{m} = 0.04$, $\Omega_\Lambda = 0$ and

(4) the currently popular Universe $\Omega_\mathrm{m} = 0.27$, $\Omega_\Lambda = 0.73$.

What is the age of the Universe for each of these models [to the nearest 100 million years]?

Produce a graph showing lookback time against redshift for the four models and comment on any overall trends.

## 3 Distance Measures

There are three useful ways to define distance.

(1) The line of sight comoving distance

$$D_\mathrm{C} = D_\mathrm{H} \int_0^z \frac{dz'}{E(z')}. \tag{9}$$

(2) The angular diameter distance is the ratio of an object's physical size to its angular size (in radians). For an object of size $\ell$ at redshift $z$ the angular size is $\theta = \ell/D_\mathrm{A}$, where $\theta$ is a small angle (so $\sin\theta \approx \tan\theta \approx \theta$).

$$D_{\rm A} = \begin{cases} D_{\rm H}\frac{1}{\sqrt{\Omega_k}(1+z)}\sinh\left[\sqrt{\Omega_k}D_{\rm C}/D_{\rm H}\right], & \text{for } \Omega_k > 0, \\ D_{\rm C}/(1+z), & \text{for } \Omega_k = 0, \\ D_{\rm H}\frac{1}{\sqrt{|\Omega_k|}(1+z)}\sin\left[\sqrt{|\Omega_k|}D_{\rm C}/D_{\rm H}\right], & \text{for } \Omega_k < 0. \end{cases} \tag{10}$$

(3) The luminosity distance $D_{\rm L}$ is defined by the relationship between the observed photon energy flux $f$, integrated over all frequencies, and the intrinsic energy output from the source $L$ by

$$f = \frac{L}{4\pi D_{\rm L}^2}.$$

It is related to the angular diameter distance by

$$D_{\rm L} = (1+z)^2 D_{\rm A}. \tag{11}$$

**Question 3** Obtain an analytic expression for the angular diameter distance $D_{\rm A}$ as a function of redshift in the case where $\Omega_{\rm m} = 1$ and $\Omega_\Lambda = 0$ and show that it has a maximum value when $z = 1.25$.

**Question 4** Write a program to determine the luminosity and angular diameter distances given the redshift $z$ and plot the dimensionless values $D_{\rm A}/D_{\rm H}$ and $D_{\rm L}/D_{\rm H}$ for redshifts $0 < z < 7$ for $(\Omega_{\rm m}, \Omega_\Lambda) = (1,0)$, $(0.04, 0)$ and $(0.27, 0.73)$. For these three cases tabulate the values at redshifts $z = 1$, 1.25, 2.0 and 4.0.

## 4 Comoving Volume

The comoving volume $V_{\rm C}$ is the volume measure in which the number density of non-evolving objects is constant with redshift. The comoving volume element in solid angle $\sin\theta d\theta d\phi$ and redshift interval $dz$ is

$$dV_{\rm C} = D_{\rm H}\frac{(1+z)^2 D_{\rm A}^2}{E(z)}\sin\theta\, dz\, d\theta\, d\phi.$$

Integrating this from the present to redshift $z$ gives the total comoving volume over the whole sky to redshift $z$,

$$V = \frac{4\pi}{3}\frac{D_{\rm L}^3}{(1+z)^3} = \frac{4\pi}{3}D_{\rm C}^3 \qquad \text{for} \quad \Omega_k = 0. \tag{12}$$

A method to test whether a sample of objects has a uniform comoving density and luminosity which does not change with cosmic time is to use the $< V/V_{\rm max} >$ test. It is assumed that all objects with observed flux $f > f_0$ are detected and the observed flux $f$ and the redshift $z$ are measured for each object. For a given luminosity $L$ there is a maximum redshift $z_{\rm max}(L)$ at which the observed flux is $f_0$ so that the object is just included. Corresponding to this redshift is a maximum volume $V_{\rm max}(L)$. Then, if we have a distribution of luminosities so that $\Phi(L)dL$ is the number per unit comoving volume with luminosity between $L$ and $L + dL$, the total number of objects in the sample is

$$\int_0^\infty \Phi(L)\int_0^{V_{\rm max}(L)} dV dL.$$

where $V$ is the comoving volume.

**Question 5** Show that for a uniform comoving distribution of objects the expectation value $\langle V/V_{max}\rangle = \frac{1}{2}$.

**Question 6** Write a program to read pairs of numbers $z$ and $f/f_0$, determine $V$ and $V_{\max}$ for these for Universe models for which $\Omega_k = 0$ and determine the average value $< V/V_{\max} >$ for each model.

Verify that for small values of $z$ the program gives the Euclidean limit, for individual cases $V/V_{\max} \propto (f/f_0)^{-\frac{3}{2}}$.

Apply the program to the sample, listed below, of 114 quasars from an area of sky. What is the value of $< V/V_{\max} >$ for this sample if $\Omega_m = 0.27$ and $\Omega_\Lambda = 0.73$? Is the value of $< V/V_{\max} >$ what you would expect from a constant comoving population? How might you interpret the result you obtain?

**Question 7** The sample in the previous question was also subject to the constraints $z > 0.20$ and $z < 3.0$, because it is only in this range that an object be recognised as a quasar. How would you modify the $V/V_{\max}$ quantity so that for a uniform distribution in this redshift range the average value is still $\frac{1}{2}$? What is the result of using this on the sample of 114 quasars?

## References

[1] Peebles, P. J. E., 1993, *Principles of Physical Cosmology*, Princeton University Press.

**Quasar data.** The following may also be found in the file `quasar.dat` in the `data` directory on the CATAM website:

| $z$ | $f/f_0$ | $z$ | $f/f_0$ | $z$ | $f/f_0$ | $z$ | $f/f_0$ | $z$ | $f/f_0$ | $z$ | $f/f_0$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.202 | 1.570 | 0.217 | 3.250 | 0.225 | 2.884 | 0.237 | 3.630 | 0.246 | 1.213 | 0.259 | 1.330 |
| 0.274 | 1.614 | 0.298 | 1.330 | 0.315 | 2.032 | 0.322 | 1.066 | 0.332 | 1.976 | 0.351 | 1.018 |
| 0.362 | 1.096 | 0.373 | 1.191 | 0.385 | 2.937 | 0.402 | 2.355 | 0.433 | 1.853 | 0.449 | 4.168 |
| 0.460 | 5.105 | 0.479 | 1.706 | 0.492 | 1.629 | 0.507 | 1.940 | 0.530 | 1.472 | 0.549 | 1.419 |
| 0.571 | 2.511 | 0.582 | 2.089 | 0.590 | 1.599 | 0.609 | 1.406 | 0.624 | 1.018 | 0.641 | 1.018 |
| 0.659 | 3.564 | 0.672 | 2.511 | 0.679 | 2.013 | 0.692 | 1.294 | 0.714 | 1.294 | 0.723 | 1.584 |
| 0.737 | 1.342 | 0.754 | 1.076 | 0.774 | 1.753 | 0.781 | 2.128 | 0.791 | 2.779 | 0.803 | 2.421 |
| 0.832 | 1.106 | 0.847 | 1.527 | 0.874 | 1.158 | 0.892 | 1.202 | 0.913 | 2.167 | 0.934 | 1.629 |
| 0.955 | 1.887 | 0.973 | 2.208 | 0.993 | 2.558 | 1.012 | 1.355 | 1.025 | 1.247 | 1.040 | 1.318 |
| 1.056 | 1.213 | 1.072 | 1.803 | 1.092 | 1.330 | 1.115 | 1.342 | 1.140 | 1.086 | 1.152 | 1.180 |
| 1.182 | 1.180 | 1.205 | 1.393 | 1.220 | 1.247 | 1.234 | 1.342 | 1.247 | 2.535 | 1.263 | 1.047 |
| 1.288 | 1.541 | 1.313 | 1.028 | 1.332 | 1.037 | 1.343 | 1.235 | 1.376 | 2.779 | 1.388 | 1.202 |
| 1.400 | 1.086 | 1.440 | 1.127 | 1.455 | 1.009 | 1.469 | 1.056 | 1.487 | 1.330 | 1.511 | 1.330 |
| 1.543 | 1.028 | 1.559 | 1.202 | 1.583 | 1.819 | 1.593 | 1.294 | 1.619 | 1.614 | 1.641 | 3.047 |
| 1.664 | 1.393 | 1.684 | 1.158 | 1.700 | 1.513 | 1.727 | 1.629 | 1.756 | 1.137 | 1.776 | 1.355 |
| 1.810 | 2.831 | 1.844 | 1.018 | 1.878 | 2.208 | 1.913 | 2.606 | 1.941 | 1.342 | 1.961 | 1.555 |
| 1.976 | 2.910 | 2.005 | 1.106 | 2.035 | 1.306 | 2.075 | 1.887 | 2.092 | 1.958 | 2.106 | 1.367 |
| 2.134 | 1.406 | 2.187 | 2.089 | 2.244 | 1.202 | 2.297 | 1.106 | 2.329 | 1.009 | 2.388 | 1.737 |
| 2.442 | 1.644 | 2.523 | 1.000 | 2.595 | 1.393 | 2.649 | 1.282 | 2.786 | 1.527 | 2.936 | 1.445 |

# 14 General Relativity

## 14.6 Isolating Integrals for Geodesic Motion (8 units)

*This project assumes material taught in the Part II course General Relativity. Some of the calculations may be done more simply using a Computer Algebra System (CAS) such as Mathematica or Maple, or the symbolic toolbox in MATLAB. Throughout we use geometrical units with $c = G = 1$.*

## 1 Geodesic Motion in Axisymmetric Spacetimes

A general axisymmetric metric can be written in the form

$$\mathrm{d}s^2 = g_{tt}\mathrm{d}t^2 + 2g_{t\phi}\mathrm{d}t\mathrm{d}\phi + g_{\phi\phi}\mathrm{d}\phi^2 + g_{rr}\mathrm{d}r^2 + g_{\theta\theta}\mathrm{d}\theta^2 \tag{1}$$

where the metric components are functions of the coordinates $r$ and $\theta$ only.

> **Question 1** By considering the Euler-Lagrange equations for $t$ and $\phi$ of the geodesic action
>
> $$\mathcal{S} = \int \sqrt{g_{ij}\frac{\mathrm{d}x^i}{\mathrm{d}\tau}\frac{\mathrm{d}x^j}{\mathrm{d}\tau}}\,\mathrm{d}\tau \tag{2}$$
>
> where the affine parameter $\tau$ is the proper time along the geodesic, or otherwise, show that
>
> $$\begin{aligned} E &= g_{tt}\frac{\mathrm{d}t}{\mathrm{d}\tau} + g_{t\phi}\frac{\mathrm{d}\phi}{\mathrm{d}\tau} \\ L_z &= -\left(g_{t\phi}\frac{\mathrm{d}t}{\mathrm{d}\tau} + g_{\phi\phi}\frac{\mathrm{d}\phi}{\mathrm{d}\tau}\right) \end{aligned} \tag{3}$$
>
> are constants of geodesic motion in any axisymmetric spacetime (1). Hence, derive the mass conservation integral
>
> $$g_{rr}\left(\frac{\mathrm{d}r}{\mathrm{d}\tau}\right)^2 + g_{\theta\theta}\left(\frac{\mathrm{d}\theta}{\mathrm{d}\tau}\right)^2 = -V_{\text{eff}}(r,\theta,E,L_z) \tag{4}$$
>
> where the *effective potential*, $V_{\text{eff}}$, should be found in terms of $E$, $L_z$ and the metric components.

The effective potential defines the allowed regions of geodesic motion for a particular choice of the energy, $E$, and angular momentum, $L_z$. Motion is only possible where $V_{\text{eff}} \geqslant 0$.

The Kerr metric has components

$$\begin{aligned} &g_{tt} = 1 - \frac{2\,m\,r}{\Sigma}, \qquad g_{t\phi} = \frac{2\,a\,m\,r\,\sin^2\theta}{\Sigma}, \qquad g_{\phi\phi} = -\left(\Delta + \frac{2\,m\,r\,(r^2+a^2)}{\Sigma}\right)\sin^2\theta, \\ &g_{rr} = -\frac{\Sigma}{\Delta}, \qquad g_{\theta\theta} = -\Sigma \end{aligned} \tag{5}$$

where $\Sigma = r^2 + a^2\cos^2\theta$, $\Delta = r^2 - 2\,m\,r + a^2$, $m$ is a constant (the mass of the black hole) and $a$ is another constant (the spin parameter of the black hole).

**Question 2** Using a CAS, or otherwise, derive expressions for the Christoffel symbols

$$\Gamma^i_{jk} = \frac{1}{2}\, g^{im}\left(g_{jm,k} + g_{km,j} - g_{jk,m}\right) \tag{6}$$

for the Kerr metric (5). You can present these results in your write-up in the form of a printout from a CAS worksheet.

**Programming Task:** Write a program to numerically integrate the second order timelike geodesic equations

$$\frac{\mathrm{d}^2 x^i}{\mathrm{d}\tau^2} = -\Gamma^i_{jk}\frac{\mathrm{d}x^j}{\mathrm{d}\tau}\frac{\mathrm{d}x^k}{\mathrm{d}\tau} \tag{7}$$

for the Kerr metric (5). Do not make use of the first integrals derived above (3)–(4), but write the four second order equations as a set of eight coupled first order equations. We will use the first integrals to verify the numerical accuracy of the integrations.

## 2 Geodesic motion in the Schwarzschild metric

The Schwarzschild metric may be obtained by setting $a = 0$ in the Kerr metric (5).

**Question 3** What are the non-zero Christoffel symbols for the Schwarzschild metric? Take $m = 1$ and find the zeros of the effective potential (4) in the equatorial plane, $\theta = \pi/2$, for the case $E = 0.97$, $L_z = 4$. Hence determine the allowed range of radii, $r$, of bound orbits in the equatorial plane. Then, using your geodesic code, do the following

a Take initial conditions $r = 15$, $\theta = \pi/2$, $\mathrm{d}r/\mathrm{d}\tau = 0$ and the value of $\mathrm{d}\theta/\mathrm{d}\tau$ determined from the effective potential (4). Plot the coordinates, $(t,\, r,\, \theta,\, \phi)$, of the particle as a function of $\tau$ over several orbits. Check that the three conservation laws (3)–(4) are satisfied at a reasonable level of numerical accuracy.

b For the same choice of $E$ and $L_z$, take a range of initial conditions that lead to bound motion (e.g., consider initial conditions in the equatorial plane with $\mathrm{d}r/\mathrm{d}\tau = 0$ and a range of values of $r(0)$). Output the values of $r$ and $\mathrm{d}r/\mathrm{d}\tau$ every time the orbit crosses the equatorial plane, $\theta = \pi/2$, with $\mathrm{d}\theta/\mathrm{d}\tau > 0$. Plot these values on a graph, with $r$ on the horizontal axis, and $\mathrm{d}r/\mathrm{d}\tau$ on the vertical axis. What do you notice?

c Experiment with a few different values of $E$, $L_z$ and initial conditions.

You have plotted a Poincaré map for these orbits. If the Poincaré map of an orbit is a closed curve it indicates the possible existence of an extra isolating integral for the motion.

## 3 Geodesic motion in the Kerr metric

We now consider $a \neq 0$ in the Kerr metric (5).

**Question 4** Take $a = 0.9$, $E = 0.95$ and $L_z = 3$, and use the effective potential to find the allowed range of $r_0$ for which the initial conditions $\theta = \pi/2$, $r = r_0$ and $\mathrm{d}r/\mathrm{d}\tau = 0$ lead to bound motion. Plot a Poincaré map as described above for a range of initial conditions of this type. Is the result similar to what you saw for the Schwarzschild metric?

**Question 5** Show that the quantity

$$Q = (aE\sin\theta - L_z \operatorname{cosec}\theta)^2 + (r^2 + a^2\cos^2\theta)^2 \left(\frac{\mathrm{d}\theta}{\mathrm{d}\tau}\right)^2 + \delta\, a^2\, \cos^2\theta \tag{8}$$

is conserved for geodesic motion in the Kerr metric, where $\delta$ is a numerical constant that should be determined. You may use a CAS to help demonstrate this, but should include evidence of the calculation. What does $Q$ become in the limit $a = 0$, i.e., for the Schwarzschild metric? Provide a physical interpretation if possible.

## References

[1] Chandrasekhar, S.; *The Mathematical Theory of Black Holes*; Clarendon Press: Oxford; 1992.

[2] D'Inverno, R.; *Introducing Einstein's Relativity*; Clarendon Press: Oxford; 1992.

[3] Goldstein, H., Poole, C. & Safko, J.; *Classical Mechanics*, third edition, Pearson Education International: New Jersey; 2002.

# 15 Number Theory

## 15.1 Primality Tests (9 units)

*This project is related to material in the Part II course Number Theory.*

A primality test is an algorithm used to determine whether or not a given integer is prime. In this project we consider several different primality tests, and apply them to numbers in the range from 3 up to $10^{10}$.

### 1 Trial division

The simplest primality test is *trial division*: $N$ is prime if and only if it is not divisible by any integer $t$ with $1 < t \leqslant \sqrt{N}$.

> **Question 1** Write a program to test for primality using trial division. Use your program to list the primes in the intervals $[188000, 188200]$ and $[10^9, 10^9 + 200]$.

### 2 The Fermat test

Fermat's Little Theorem states that if $p$ is prime then $a^{p-1} \equiv 1 \bmod p$ for any $a$ coprime to $p$. The *Fermat test base a* for $N$, if $1 < a < N$, is to compute $a^{N-1} \bmod N$: if this is not $\equiv 1 \bmod N$ then $N$ is certainly composite. A *Fermat pseudoprime base a* is a composite $N$ which passes the Fermat test base $a$.

> **Question 2** Write a program to carry out the Fermat test base $a$, capable of working for $N$ up to $10^{10}$. Run it on the intervals in Question 1, say for $a$ up to 13. Check that the output is consistent with your earlier answer, and make a note of any Fermat pseudoprimes that you find.
>
> You should take care to devise a good algorithm for computing $a^b \bmod N$, ensuring that there is no possibility of integer overflow during the calculation, in view of the possibility that the integers in your chosen language may be limited to $10^{15}$ or similar. (See the note on programming at the end of the project.) If the integers in your language are large enough for there to be no chance of overflows, then you should still comment briefly on how you might manage if you were limited to $10^{15}$. (*Hint*: A multiplication modulo $N$ can be done in two pieces.)
>
> Briefly discuss the complexity of your algorithm, *i.e.* the time theoretically taken. To do this, you should consider (without coding it) how your program might extend to arbitrarily large $N$, and work out roughly how many basic operations are needed as $N \to \infty$, where a basic operation could be addition or multiplication of two numbers of similar size to $N$. (A more sophisticated analysis might also take into account the time required to add and multiply large numbers on a finite machine, but it isn't necessary to go into such details here.)

An *absolute Fermat pseudoprime*, also called a *Carmichael number*, is a composite number which passes the Fermat test for any base $a$ with $(a, N) = 1$.

**Question 3** Find the Fermat pseudoprimes base 2 and the absolute Fermat pseudoprimes up to $10^6$. Can you think of any very simple methods (not involving any elaborate theory) to speed up the checking of the absolute pseudoprimes?

How many values of $a$, starting from $a = 2$, are necessary to determine the primality or compositeness of those $N$ in this range which are not absolute Fermat pseudoprimes? Note that if $a$ has a non-trivial common factor with $N$ then we regard $a$ as having determined that $N$ is composite.

The existence of infinitely many absolute Fermat pseudoprimes (proven in 1994) means that the Fermat test cannot be relied on to prove the primality of $N$ any faster than trial division, although it can usually detect compositeness quickly.

## 3 The Euler test

We can improve the Fermat test by using Euler's theorem, that if $p$ is an odd prime then $a^{(p-1)/2} \equiv (a/p) \bmod p$ where $(a/p)$ is the Legendre symbol. The *Euler test base a*, for an odd integer $N$, is to compute $a^{(N-1)/2} \bmod N$ and check if this is $\pm 1$ and equal to the *Jacobi symbol* $(a/N)$: for $N$ odd and positive this satisfies the properties

$$\left(\frac{a}{N}\right) = \left(\frac{a \bmod N}{N}\right),$$

$$\left(\frac{ab}{N}\right) = \left(\frac{a}{N}\right)\left(\frac{b}{N}\right),$$

$$\left(\frac{-1}{N}\right) = (-1)^{(N-1)/2},$$

$$\left(\frac{2}{N}\right) = (-1)^{(N^2-1)/8},$$

and, if $M$ and $N$ are odd positive and coprime,

$$\left(\frac{M}{N}\right)\left(\frac{N}{M}\right) = (-1)^{(M-1)(N-1)/4}.$$

If $N$ is prime then $(a/N)$ is just the Legendre symbol. As before, an *Euler pseudoprime base a* is a composite $N$ which passes the Euler test base $a$, and an *absolute Euler pseudoprime* is a composite number which passes the Euler test for any base $a$ with $(a, N) = 1$.

**Question 4** Write a procedure to evaluate the Jacobi symbol and modify your previous program to carry out the Euler test. Find the Euler pseudoprimes base 2 and the absolute Euler pseudoprimes up to $10^6$. How many values of $a$ are necessary to determine the primality or otherwise of those $N$ in this range which are not absolute Euler pseudoprimes?

## 4 The strong test

If $p$ is prime then the multiplicative group $(\mathbb{Z}/p\mathbb{Z})^*$ is cyclic and so $-1$ is the only element of order 2. If we put $p - 1 = 2^r s$ where $s$ is odd then either $a^s \equiv 1 \bmod p$ or there is $j$, $0 \leqslant j < r$ such that $a^{2^j s} \equiv -1 \bmod p$. The *strong*, or *Miller–Rabin test base a* is to put $N - 1 = 2^r s$ with $s$ odd and to compute the sequence $a^s$, $a^{2s}$, $\ldots$, $a^{2^{r-1}s}$: if the sequence begins with 1 or contains $-1$ then $N$ passes the strong test.

**Question 5** Modify your program to carry out the strong test. As before, find the strong pseudoprimes base 2 and the absolute strong pseudoprimes up to $10^6$. How many values of $a$ are required to determine the character of the integers $N$ in this range?

**Question 6** Use your programs to tabulate the number of Fermat/Euler/strong pseudoprimes base 2 in the intervals $[10^k, 10^k + 10^5]$ for $5 \leqslant k \leqslant 9$. For comparison you should also list the numbers of primes in these intervals. Repeat for base $a = 3$, and also for numbers passing the corresponding tests for both $a = 2$ and $a = 3$.

Briefly comment on the relation between the Fermat, Euler and strong tests.

**Question 7** Combining all your previous results, devise the most efficient algorithm you can for determining the primality or otherwise of a number $N$ in the range from 3 up to $10^{10}$. (This may be a mixture of the above methods.)

Experiment on a suitable number of random numbers, say 10000, in the range to compare the time taken by your algorithm and trial division. Comment also on the time required theoretically for these algorithms for large $N$, making clear any assumptions you are making.

An important question in applications is the reliability of these tests for determining the primality of randomly chosen numbers in some range.

**Question 8** Fix a value of $k$ and suppose that $N$ is chosen uniformly at random from all odd integers of exactly $k$ bits, that is, between $2^{k-1}$ and $2^k - 1$. (You should be able to take $k$ at least 15.) Investigate the probability that if $N$ passes $t$ rounds of the strong test with randomly chosen $a$, then $N$ is in fact composite.

## Programming note

The quantities of interest in this project are natural numbers. You might be aware that computer languages typically like to be told if a number is an integer rather than a general real, because it means there is no need to save a fractional part, and it avoids concerns over rounding errors.

MATLAB, though, is incorrigibly real-minded. Hence, although MATLAB allows you to tell it a number is an integer with the `int32` command (see the CATAM manual), it is better for this project if you don't do so. In other words, you need not worry about distinguishing between reals and integers.

You should, however, be aware that MATLAB can handle integers in this way only up to about 15 digits. (Try `eps(10^15)` and `eps(10^16)`. Try also `10^8 * 10^8 - (10^16-1)`.) So you must avoid calculations involving integers exceeding $10^{15}$.

You may use the MATLAB functions `conv` and `deconv` to multiply and divide polynomials.

Obviously no credit can be given in this project for using inbuilt functions such as `isprime` or `factor` — you are expected to write and analyse your own programs. You may however use the inbuilt MATLAB function `gcd`.

## References

[1] Riesel, H., *Prime numbers and computer methods for factorisation*, 2nd edition, Progress in Mathematics 126, Birkhauser, 1994

[2] Koblitz, N., *A Course in Number Theory and Cryptography*, Graduate Texts in Mathematics 114, Springer, 1987.

# 15 Number Theory

## 15.10 The Continued Fraction Method for Factorization (8 units)

*This project is related to material in the Part II course Number Theory.*

### 1 Factor bases

In this project $N$ will be a (usually large) integer, that we would like to factor, and $B$ will be a finite set of (usually small) primes. We call $B$ the factor base. Sometimes it is convenient to allow $-1$ as an element of $B$.

> **Question 1** Write a program, using trial division, to test whether $N$ is a product of primes in $B$ (we say that $N$ is $B$-smooth), and if so to give the prime factorization. Use your program to estimate, for a suitable range of $d$, the probability that a $d$-digit integer is $B$-smooth where $B$ is the set of primes less than 50.

The integers that may be accurately represented in your chosen language may be limited to $10^{15}$ or similar. For example in MATLAB numbers are represented by default as doubles, meaning that they are stored to 16 significant (decimal) figures. For integers larger than $10^{15}$ functions such as `mod` and `int2str` may give incorrect answers. If your language is able to handle larger integers, then you are still expected to comment where appropriate on how you would manage if you were restricted to $10^{15}$.

On a modern computer it is practical to factor integers of the size considered in this project by trial division. We study a factoring method that remains practical for much larger values of $N$. To enable comparisons, we therefore make the following artificial restriction: the factor base $B$ is only allowed to contain primes that are less than 50.

### 2 Continued fractions

The continued fraction algorithm applied to a real number $x_0 = x$ forms a sequence of partial quotients $a_n$ by the transformation

$$\begin{aligned} a_n &= \lfloor x_n \rfloor \\ x_{n+1} &= \frac{1}{x_n - a_n} \end{aligned}$$

where as usual $\lfloor x \rfloor$ denotes the greatest integer $\leqslant x$. The algorithm terminates if $x_n = a_n$; this happens if and only if the initial $x$ is rational.

The convergents $P_n/Q_n$ are defined for $n \geqslant 0$ by

$$\begin{aligned} P_n &= a_n P_{n-1} + P_{n-2} \\ Q_n &= a_n Q_{n-1} + Q_{n-2} \end{aligned}$$

with initial conditions $P_{-2} = 0$, $P_{-1} = 1$ and $Q_{-2} = 1$, $Q_{-1} = 0$.

**Question 2** Show that if $x = \sqrt{N}$ for some positive integer $N$ then each $x_n$ may be written in the form $(r + \sqrt{N})/s$ with $r$, $s$ integers and $s \mid (r^2 - N)$.

Write a program to develop the continued fraction expansion of $\sqrt{N}$. Your program should work with integers as far as possible, so that there is no risk of rounding errors. In some programming languages it is best to use an integer type, but there is no particular advantage in doing this if you are using MATLAB.

Tabulate the partial quotients of $\sqrt{N}$ for $1 \leqslant N \leqslant 50$ and comment on the results. Investigate how large $r$ and $s$ can become (in terms of $N$).

For a fixed value of the positive integer $N$, the equation $x^2 - Ny^2 = 1$, in integer unknowns $x$ and $y$, is called Pell's equation. The negative Pell equation is $x^2 - Ny^2 = -1$.

**Question 3** Tabulate the quantities $P_n^2 - NQ_n^2$ for some values of $N$ and hence comment on the use of continued fractions to solve Pell's equation, and the negative Pell equation. Can you see any simple condition on $N$ which ensures that the negative Pell equation is insoluble?

Write a procedure that given $x, y, N \leqslant 10^{15}$ tests whether $x^2 - Ny^2 = \pm 1$, being careful to avoid integer overflow. (Hint: try working mod $p$ for several primes $p$.)

Use your observations to write a program to find non-trivial solutions to Pell's equation. Tabulate the solutions found for each $N$ in the ranges $1 \leqslant N \leqslant 100$ and $500 \leqslant N \leqslant 550$, when such a solution exists. You may find a few values of $N$, such as 509, beyond the capacity of your program; you do not need to correct for this. You should however make sure that all answers you do give are correct.

## 3 Factorization

By "factorization" we mean the task of finding a non-trivial factor of a composite integer $N$. We will not be concerned with finding the complete factorization of $N$ into primes. You should assume from now on that $N$ is odd.

**Question 4** Suppose we are given a supply of integers $x$, $y$ with $x^2 \equiv y^2 \bmod N$. How might this help us factor $N$? Estimate the complexity of the steps involved. If $N$ is composite, can we be sure that suitable $x$ and $y$ exist?

One source of integers $x$, $y$ with $x^2 \equiv y^2 \bmod N$ arises from the convergents of the continued fraction expansion of $\sqrt{N}$.

**Question 5** Modify your programs to compute $P_n \bmod N$ and $P_n^2 \bmod N$ for $N$ up to $10^{10}$. Explain how you avoid integer overflow. (Hint: a multiplication mod $N$ can be done in two pieces.) Run your program for $N = 2012449237$, $2575992413$ and $3548710699$.

**Question 6** Let $A$ be a matrix over the field $\mathbb{F}_2 = \mathbb{Z}/2\mathbb{Z}$. Write a program using Gaussian elimination to determine whether there exists a non-zero column vector $\mathbf{v}$ with $A\mathbf{v} = 0$, and to find one if there is.

**Question 7** Implement the continued fraction method for factorization, and discuss briefly how it works. Your program should be capable of giving details of the intermediate steps involved. Give some detailed worked examples, including the values of $N$ in Question 5. Investigate the number of convergents typically required for factorization.

Discuss the choice of factor base $B$. What improvements could be made to your method to make it more efficient for very large values of $N$?

## References

[1] Davenport, H., *The higher arithmetic.*

[2] Koblitz, N., *A course in number theory and cryptography.*

[3] Riesel, H., *Prime numbers and computer methods for factorization.*

# 16 Algebra

## 16.1 The Galois Group of a Polynomial (7 units)

*This project is related to material in the Part II course Galois Theory.*

## 1 Introduction

The *Galois group* $G(f)$ of a polynomial $f$ defined over a field $K$ is the group of $K$-automorphisms of the field generated over $K$ by the roots of $f$ (the Galois group of the splitting field for $f$ over $K$).

We shall consider Galois groups over the rationals and polynomials $f$ which are monic and have coefficients in $\mathbb{Z}$. Assume that $f$ has no repeated factor. Let $f$ have degree $\partial f = n$ with Galois group a subgroup of the symmetric group $S_n$ acting on the roots of $f$. The *decomposition group of $f$ modulo $p$* is the Galois group $G_p(f)$ of $f$ regarded as a polynomial over the finite field $\mathrm{GF}(p)$, provided that $f$ modulo $p$ does not have a repeated factor. The key result we shall use is that the decomposition group (when defined) is always cyclic and isomorphic to a subgroup of the Galois group of $f$. Furthermore, it is isomorphic in a way which preserves cycle types, so the cycle type of the generator of the decomposition group will also occur in the Galois group.

We shall use the decomposition groups to derive information about the Galois group of a polynomial $f$. For example, if $G(f)$ contains a 2-cycle, a $(n-1)$-cycle and an $n$-cycle, then it must be $S_n$. As the decomposition group is always cyclic, this information does not distinguish between groups with the same abelian subgroups, but computing a sufficient number of decomposition groups will usually determine the cyclic subgroups and hence often determine $G(f)$, although there is always the possibility that our answer will be too small if we do not compute enough.

## 2 The algorithms

To find the decomposition group of $f$ modulo $p$, we need information about the factorisation of $f$ over $\mathrm{GF}(p)$. There is a repeated factor in $f$ iff $f$ has a factor in common with its formal derivative $f'$ and this can be determined by applying the Euclidean algorithm. Since $\mathrm{GF}(p^r)$ is the splitting field of any irreducible polynomial of degree $r$ over $\mathrm{GF}(p)$ and the Galois group of $\mathrm{GF}(p^r)$ over $\mathrm{GF}(p)$ is the cyclic group $C_r$ generated by $x \mapsto x^p$, it is only necessary to find the degrees of the irreducible factors in $f$ in order to find its Galois group over $\mathrm{GF}(p)$. We let $f_r$ be the product of all the irreducible factors of degree $r$ in $f$: then there are $n_r = \partial f_r / r$ factors of degree $r$ in $f$ and the Galois group $G_p(f)$ of $f$ over $\mathrm{GF}(p)$ is cyclic, where the generator has $n_r$ $r$-cycles for each $r$.

We determine $f_r$ by the observation that the elements of $\mathrm{GF}(p^r)$ all satisfy the equation $\phi_r(X) = X^{p^r} - X = 0$ and hence, if we proceed by successively removing the factors $f_1, \ldots, f_{r-1}$ then at the $r^{\text{th}}$ stage we can obtain $f_r$ by taking the highest common factor of the residue with $\phi_r$.

**Question 1** Write procedures to compute the quotient and remainder from dividing two polynomials over $\mathrm{GF}(p)$ and use them to write a procedure to find the highest common factor of two polynomials over $\mathrm{GF}(p)$. Include in your report some test output from all three procedures. Describe an efficient way of using your procedures to compute a large power of one polynomial modulo another polynomial.

**Question 2** Write a procedure to compute the decomposition group of $f$ modulo $p$. You should check first that the group is defined, that is, that $f$ and $f'$ have no common factor, and then decompose $f$ into factors $f_r$. You should try to make this procedure reasonably efficient computationally. Use your procedures in a program that will read in a monic polynomial $f$ with integer coefficients, and print out its decomposition groups for $p$ up to, say, 97.

**Question 3** Run your program for the polynomials

$$\begin{aligned}
&X^2 + X + 41,\\
&X^3 + 2X + 1,\\
&X^3 + X^2 - 2X - 1,\\
&X^4 - 2X^2 + 4,\\
&X^4 - X^3 - 4X + 16,\\
&X^4 - 2X^3 + 5X + 5,\\
&X^4 + 7X^2 + 6X + 7,\\
&X^4 + 3X^3 - 6X^2 - 9X + 7,\\
&X^5 + 36,\\
&X^5 - 5X + 3,\\
&X^5 + X^3 - 3X^2 + 3,\\
&X^5 - 11X^3 + 22X - 11,\\
&X^6 + X + 1,\\
&X^7 - 2X^6 + 2X + 2,\\
&X^7 + X^4 - 2X^2 + 8X + 4,\\
&X^7 + X^5 - 4X^4 - X^3 + 5X + 1.
\end{aligned}$$

Your program should tabulate its output in columns, so that the results for this question take only a few pages in total.

**Question 4** Discuss the Galois groups of these polynomials in the light of your output, with special reference to the reducible polynomials. Assuming, in each case, that the group is the smallest possible, formulate a conjecture as to the relative frequencies of the various cycle shapes for a fixed polynomial $f$ as $p$ varies. Do any of the polynomials (especially those of smaller degree) appear to contradict this conjecture? If so, run your programs for these polynomials for higher values of $p$ and see if this rectifies the matter.

## Programming note

If you use MATLAB then you may wish to use the `DocPolynom` class that is included as an example in the help browser. To use this you should create a directory `@DocPolynom` and place `DocPolynom.m` into it. This will enable you to define and display (non-zero) polynomials and to carry out standard algebraic manipulations with them. There is no need to include the class file in your program listings (assuming you do not modify it). *[The latest version requires MATLAB 2022b or later to run.]*

If you use a computer algebra package (such as MAPLE), then you may find that some of the routines asked for in this project are included in the package. In such cases, no credit will be given for using the packaged routines — you are expected to write your own programs. You may wish to compare the answers given by your program and by the packaged routines.

## References

[1] B. L. van der Waerden, *Modern Algebra vol. 1*

# 16 Algebra

## 16.5 Permutation Groups (7 units)

*This project is self-contained, building on theory covered in the Part IA course* Groups. *Some knowledge of the groups part of the Part IB course* Groups, Rings and Modules *would be useful.*

## 1 Introduction

Suppose we are given a set of permutations of $X = \{1, \dots, n\}$. They generate a finite permutation group $G \leqslant S_n$. The aim of this project is to replace the given set of generators of $G$ with another generating set for $G$ which is of greater utility, hopefully allowing us to deal with various questions. For programming purposes you do not need to go above $n = 20$ (although you are welcome to if you so wish).

## 2 Permutations

A permutation $\pi$ of $X$ is a bijective function from $X$ to $X$. If $x$ is an element of $X$ then the image of $x$ under $\pi$ is written $\pi x$. If $\pi_1$ and $\pi_2$ are permutations then their product $\pi_1 \cdot \pi_2$ maps $x$ to $\pi_1(\pi_2 x)$. The set of all permutations of the set $X = \{1, \dots, n\}$ is the symmetric group $S_n$. If $\pi$ is a permutation and $y = \pi y$ then $y$ is called a *fixed point of* $\pi$.

**Question 1** Write procedures to compute the inverse $\pi^{-1}$ of a permutation $\pi$ and the product $\pi_1\pi_2$ of two permutations $\pi_1$ and $\pi_2$. What is the complexity of your method for computing inverses (as a function of $n$)?

## 3 Groups

Suppose the permutation group $G$ is generated by permutations $\pi_1, \dots, \pi_k$. First we reduce the number of generators with the Stripping Algorithm of Sims. Let $A$ be an $n \times n$ array of permutations which is initially empty.

Suppose we have already put the first $l - 1$ permutations into the array. If $\pi_l$ does not fix 1 and the $\pi_l(1)$th entry in the first row is still empty then put $\pi_l$ there. Suppose the $\pi_l(1)$th entry is the permutation $g$. Then modify $\pi_l$ to be $g^{-1}\pi_l$ so the new $\pi_l$ fixes 1. Go to the second row.

If $\pi_l$ does not fix 2 and the $\pi_l(2)$th entry in the second row is still empty then put $\pi_l$ there. If the entry is $g$ then modify $\pi_l$ to be $g^{-1}\pi_l$ which hence fixes 1 and 2. Go on to the third row ...

If we reach the last row then we must have produced the trivial permutation which can be omitted from the generating set.

Once a permutation is placed in the array, or deemed to be the trivial permutation, we go on to try to place the next permutation in the array.

**Question 2** Show that the modified set of permutations generates the group $G$. Give an upper bound for the size of the modified set of generators and for the number of operations needed to complete the algorithm. (As a function of $n$ and the size of the original generating set, noting that, e.g., storing a permutation is $O(n)$ operations.)

**Question 3** Write a procedure which computes the array of a permutation group given by a set of generators. It should receive a set of permutations as input and give a set of permutations as output, which generate the same group and are in the above reduced form. (Here, as elsewhere in this project, you should give some examples to demonstrate that your program is working correctly.)

## 4 Orbit and Stabilizer

Let $G$ be a permutation group of $X$. If $\alpha \in X$ then the set $A = \{\beta \in X \mid \exists g \in G, g\alpha = \beta\}$ is called the *orbit* of $\alpha$ (under $G$). If $\beta \in X$ is in the orbit of $\alpha$ then an element $g \in G$ is called a *witness* of this if $g\alpha = \beta$. It is easy to see that $\beta$ is in the orbit of $\alpha$ if and only if the orbit of $\beta$ is the same as the orbit of $\alpha$. Hence different orbits are disjoint and the orbits form a partition of $X$.

The *stabilizer* of an element $\alpha$ in $X$ is $G_\alpha = \{g \in G \mid g\alpha = \alpha\}$. It is a subgroup of $G$.

**Question 4** Write down a bijection between the set of left cosets of $G_\alpha$ in $G$ and the orbit of $\alpha$. State the orbit-stabilizer theorem.

**Question 5** Write a procedure which computes the orbit with witnesses of a given element under a permutation group $G$ generated by a given set of permutations. It should receive as input a set of permutations and an element $\alpha \in X$ and should return as a output a list of elements forming the orbit of $\alpha$, together with a witness in each case. Briefly explain how your procedure works.

## 5 Schreier's Theorem and the final algorithm

Suppose $G$ is a permutation group of $X$, given as a set of generators $Y$, $\alpha$ is an element of $X$ and $T$ is a complete set of left coset representatives of $G_\alpha$ in $G$. Let the surjective map $\varphi\colon G \to T$ be defined via $g\alpha = \varphi(g)\alpha$.

**Question 6** Let $x$ be an element of $G_\alpha$. Write $x = y_r \dots y_1$ with each $y_i$ an element of $Y$. Let $t_1$ be the element of $T$ belonging to $G_\alpha$. Let $t_{i+1} = \varphi(y_i t_i)$ for $i = 1, 2, \dots, r$. Show that $t_{r+1} = t_1$. Deduce that $G_\alpha$ is generated by the set of elements:

$$\{\varphi(yt)^{-1} \cdot y \cdot t \mid y \in Y, t \in T\}.$$

This is a special case of Schreier's Theorem.

**Question 7** Write a procedure which computes a generating set of a stabilizer of a permutation group given as a set of generators. It should receive a set of permutations and an element $\alpha$ as input and give a set of permutations as output which generate the stabilizer. Use Question 5 to obtain $T$, then use Schreier's Theorem and finally reduce the set of generators with the Stripping Algorithm. Comment on the complexity of your algorithm.

**Question 8** Write a program which computes the order of a permutation group $G$ given with a set of generating permutations. The program should receive a set of permutations as input and give a natural number as output which is the order of $G$. You should first reduce the number of generators with the Stripping Algorithm, and recursively find a nontrivial orbit and use the previous question until you reach a subgroup of order 1. Use the orbit-stabilizer theorem in the recursive part to find the order of $G$. You should make a note of the group order and number of generators (before and after stripping) for each of the subgroups computed. Give some brief output from your program.

What might happen if we forgot to use the Stripping Algorithm at every stage? (For instance, say the input was two permutations of $S_{20}$.)

**Question 9** For the group $S_n$ (throughout this question you may take $n \geqslant 5$), we consider the probability $P_n$ that a pair of elements $g, h$ picked uniformly at random generates $S_n$. In other words we have

$$P_n = \frac{|\{(g,h) \in S_n \times S_n : \langle g, h \rangle = S_n\}|}{|S_n|^2}.$$

Why do you know from IA that $P_n > 0$? Give a straightforward argument to show that there is $k < 1$ *independent of* $n$ such that $P_n \leqslant k$. What is your value for $k$? What is the value of $P_n$ for very small $n$?

For each of a few moderate values of $n$, generate 100 or so random pairs of permutations. Describe how you generate a random permutation. (To do this, you may assume you have a random number generator which, with input an integer $N$ from 1 to say 100, will output an integer uniformly at random between 1 and $N$ inclusive.)

Using your previous program, what sort of estimates do you obtain for $P_n$?

# 17 Combinatorics

## 17.1 Graph Colouring (7 units)

*This project is based on the material found in the Part II Graph Theory course.*

*In this project you will need to be able to generate graphs from $\mathcal{G}(n,p)$ and $\mathcal{G}_k(n,p)$. The space $\mathcal{G}(n,p)$ is that of graphs with $n$ labelled vertices, edges appearing independently and at random with probability $p$. The space $\mathcal{G}_k(n,p)$ differs from $\mathcal{G}(n,p)$ only in that $ij$ is never an edge if $i - j \equiv 0 \pmod k$.*

## 1 A simple colouring algorithm

A *colouring* of a graph $G$ is an assignment of a colour to each vertex of $G$, so that no two adjacent vertices receive the same colour. The *chromatic number* of $G$, denoted by $\chi(G)$, is the smallest number of colours for which it is possible to produce a colouring. It is believed that finding the chromatic number of a graph $G$ is, in general, very hard. At no stage in this project are you required to implement a procedure for the exact evaluation of $\chi(G)$. You will, however, develop procedures for finding upper and lower bounds for $\chi(G)$.

The *greedy algorithm* colours a graph whose vertex set is *ordered* by colouring vertices one at a time in the order given, using colours from $\{1, 2, 3, \ldots\}$. The colour chosen for a vertex is the least colour from among those not already assigned to any previously coloured neighbours.

**Question 1** Write a procedure which applies the greedy algorithm to a graph with a given ordering of the vertices. Test your program on ten members of $\mathcal{G}(70, 0.5)$, and compare the number of colours used when the vertices are ordered in the following ways: (i) by increasing degree, (ii) by decreasing degree, (iii) where $v_j$ has minimum degree in the graph $G - \{v_{j+1}, \ldots, v_n\}$, (iv) at random.

Do the same for $\mathcal{G}_3(70, 0.75)$.

**Question 2** What ordering will guarantee that the greedy algorithm uses no more than 3 colours for $\mathcal{G}_3(70, 0.75)$? Why do you think the probability 0.75 was chosen here? For each $n$ give an example of a graph $G$ of order $3n$ such that $\chi(G) = 3$ but on which greedy might need $n + 2$ colours.

## 2 Cliques

A *clique* in a graph $G$ is a complete subgraph of largest order in $G$. (This definition differs from some in the literature.) Notice that $\chi(G)$ is at least as large as the order of a clique.

A greedy-type algorithm for finding a complete subgraph in $G$ would start with a subgraph of order one (a vertex) and repeatedly try to find a vertex joined to all vertices of the subgraph selected so far, until no further such vertex could be found.

**Question 3** Give an argument to suggest that it is unlikely the greedy-type algorithm will find a complete subgraph of order 14 in a graph from $\mathcal{G}(2000, 0.5)$. How large do you think a clique is likely to be in a graph from $\mathcal{G}(2000, 0.5)$?

**Question 4** Write a procedure to find a clique in a graph $G$. [Note: this procedure may be time-consuming but should not be excessively so on the examples here.] Compare, for several graphs, the resulting lower bound you get on $\chi(G)$ with the upper bounds obtained previously.

## 3 Colouring

An *independent set* in a graph is a subset of the vertex set which spans no edges. A colouring is thus just a partition of the vertices into independent sets.

**Question 5** Convert your clique procedure to find an independent set of maximum order in a graph. Hence write a procedure to colour a graph by the following method. First find a largest independent set $I_1$. Then find a largest independent set $I_2$ in $G - I_1$, then $I_3$ in $G - I_1 - I_2$, and so on until nothing remains. Compare the upper bounds on $\chi(G)$ so obtained with previous bounds. Try your program on ten members of $G_7(70, 0.5)$ also. Is there a change in behaviour as $p$ is increased, say from 0.4 to 0.6? If your program can handle larger graphs in a finite time, obtain further data of interest.

None of the above methods for bounding $\chi(G)$ is guaranteed to find $\chi(G)$ exactly.

**Question 6** Estimate (crudely) the theoretical running times of all the algorithms used above as functions of $n$ when the input is a typical member of $\mathcal{G}(n, 0.5)$. Describe in outline, but do not implement, a procedure for colouring a graph with exactly $\chi(G)$ colours, and estimate its running time.

## References

[1] Bollobas, B., Modern Graph Theory, Springer 1998.

# 17 Combinatorics

## 17.3 Hamiltonian cycles (5 units)

*This project is based on the material found in the Part II Graph Theory course.*

*In this project you will need to be able to generate graphs from $\mathcal{G}(n, p)$, the space of graphs with $n$ labelled vertices, edges appearing independently and at random with probability $p$.*

A *Hamiltonian cycle* in a graph is a cycle which contains every vertex.

**Question 1** Describe a simple algorithm to check whether a graph has a Hamiltonian cycle, and implement it. Test your program on a few particular graphs; and then use it on a selection of graphs from $\mathcal{G}(n, p)$ with $n$ up to 21 and $p$ firstly varying from 0.1 to 0.9 and then varying from $0.1 \ln n/n$ to $1.9 \ln n/n$. Tabulate your results, showing for each $n$ and $p$ the number of graphs from your selection which had a Hamiltonian cycle.

**Question 2** Estimate the theoretical running time of your algorithm as best you can. Compare the answers for the worst case and an average case.

**Question 3** Find a simple property possessed by many of your non-Hamiltonian examples that is sufficient (though maybe not necessary) to force a graph to be non-Hamiltonian. Why do you think the second range of values of $p$ was chosen?

You will notice that your algorithm rapidly becomes prohibitively expensive as the order of the graph increases. In this case an "approximation algorithm" can be useful. An approximation algorithm for the Hamiltonian cycle problem would seek to make a very good attempt at finding a cycle in a short space of time. If it succeeds, well and good. If it fails, there may have been a cycle it missed, but it is hoped that the probability of this will be small.

Here is a simple algorithm to search for a Hamiltonian cycle. Construct a sequence of paths $P_1, P_2, \ldots$ , where $P_1$ is just a single vertex $v_0$. Given a path $P_j$ from $v_0$ to $v_k$, proceed as follows:

1. If $P_j$ has length $n-1$ and $v_0v_k \in E(G)$, output a Hamiltonian cycle;
2. if $P_j$ has length less than $n-1$ and $v_k$ is joined to a vertex not in $P_j$, extend the path $P_j$ to a path $P_{j+1}$. If there are several neighbours not in $P_j$, pick one of them at random;
3. otherwise construct a new path of the same length as $P_j$ in this way: select a neighbour $v_i$ of $v_k$ in $P_j$ at random. Then $P_{j+1}$ is the path $v_0 \ldots v_{i-1}v_iv_kv_{k-1} \ldots v_{i+1}$.

**Question 4** Implement this algorithm, and try it on your earlier examples. You should set a stopping time $T$for the procedure so that, if it has constructed $P_T$ and still found no cycle, it quits. What functions work well in practice (i.e. fairly reliably find a cycle but aren't too expensive)?

**Question 5** In general, the stopping time $T = T(n, p)$ should be a function of both $n$ and $p$. In the case that $p$ is fixed and $n$ is large, what do you think would be a good choice of $T$? How do you think the running time might vary with $p$.

## References

[1] Bollobas, B., Modern Graph Theory, Springer 1998.

# 19 Communication theory

## 19.1 Random codes (5 units)

*Background material for this project is given in the Part II course Coding and Cryptography.*

The (binary) Hamming space $\{0,1\}^n$ consists of all possible $n$-tuples of 0s and 1s. We define a *code* $C$ of length $n$ and size $r$ to be a subset $C$ of the Hamming space $\{0,1\}^n$ with $r$ elements. The *Hamming distance* between two elements $\mathbf{x} = (x_i)$, $\mathbf{y} = (y_i)$ of $\{0,1\}^n$ is the number of places in which $\mathbf{x}$, $\mathbf{y}$ differ. The *minimum distance* of a code $C$ is the minimum Hamming distance $d(C)$ between distinct elements of $C$. The *information rate* of $C$ is $\frac{1}{n}\log_2 r$. We define the *error-control rate* to be $(d-1)/n$ (note: elsewhere it is often defined as $d/n$).

In this project we investigate how high the information rate of a randomly-generated code can be, subject to constraints on its error-control rate. In the first study, any randomly chosen code may be considered, whereas in the second study only randomly generated linear codes are allowed. In each study we try two approaches: specify the information rate of the random code and see what error-control rate can be achieved, or specify the error-control rate and see what information rate can be achieved.

**Question 1** Write a procedure to find the minimum distance of a code. Use your procedure to write a program which generates random codes of length $n$ and size $r$ and then computes the minimum distance.

Run your program several times with various values of $n$ and $r$, for each choice finding the best (i.e., largest) $d$ that you can.

**Question 2** Now generate codes of length $n$ and minimum distance $d$ by starting with an initial code vector, say $(0,\ldots,0)$ and randomly generating further vectors, adding a new vector to the code if it has distance at least $d$ from all the vectors already in the code.

Run your program several times for each choice of parameters $n$ and $d$, finding the best (i.e., largest) $r$ that you can.

**Question 3** Take the output from the two previous questions and plot the corresponding points on a graph with information rate and error-control rate as the two axes. Comment on your results.

We call a code *linear* if it forms a subspace of the Hamming space, regarded as a vector space over the field $F$ of 2 elements. The *weight* $w(\mathbf{x})$ of a vector $\mathbf{x}$ is the number of non-zero components, that is, the Hamming distance $d(\mathbf{x},\mathbf{0})$. The minimum distance of a linear code is just the minimum non-zero weight. The *rank* $k$ of a linear code is the dimension of the code as a subspace, and the size of a linear code is $r = 2^k$.

**Question 4** Write a procedure to find the minimum non-zero weight of a code generated over $F$ by a set $g_1, \ldots, g_k$ of $k$ generators. Use your procedure to find linear codes of given length $n$ and either given rank $k$ or given minimum distance $d$ by considering random sets of generators. As before, run your programs several times to plot the information and error-control rates and comment on the results.

**Question 5** Comment on your results in relation to known constraints on the design of codes and the Shannon coding theorems.

Comment on these methods as a way of designing effective error-control codes.

## References

[1] C.M. Goldie and R.G.E. Pinch, *Communication theory*, CUP, 1991.

# 20 Probability

## 20.5 Percolation and the Invasion Process (9 units)

*This project requires general knowledge of probability theory, at the level of IA Probability. It also requires competency in programming.*

## 1 Introduction

This project concerns certain probability models for bond percolation. The book by Grimmett [1] is a good source to learn more about percolation.

We work on a connected graph $G = (V, E)$, that is, a collection of nodes $V$ connected by edges $E$. To each edge $e \in E$, we assign, independently, a uniform random variable $U_e \sim U[0, 1]$. We decide on a value $p \in [0, 1]$; we declare the edge $e$ to be *p-open* if $U_e < p$, and we declare it to be *p-closed* otherwise.

When $p$ is very small, very few edges are open; but as we increase $p$, there appear open clusters, i.e. sets of nodes connected by open edges.

Percolation theory is the study of the geometry of the open clusters. In particular, important questions are whether or not there exists an infinite cluster of open edges; and if one does exist, how many infinite clusters there are. Clearly if $p = 0$ there is none and if $p = 1$ there is one open cluster, namely the graph $G$ itself.

## 2 The binary tree

Let $V$, the set of nodes of the graph, consist of finite strings, as follows: $V$ contains the empty string '' (also known as *Eve*), and the three strings '1', '2' and '3' (also known as *Eve's daughters*), and also every string that is one of Eve's daughters followed by a finite sequence of '1's and '2's. Two nodes are connected by an edge if one can be obtained by appending one digit to the other. For example, '3221' is connected to '322' (its mother) and to '32211' and '32212' (its two daughters). As before, each edge $e$ is assigned a random variable $U_e \sim U[0, 1]$.

(We can use this as a crude model to describe the propagation of a defective gene in a population.)

**Question 1** Let $\phi_p$ be the probability that Eve's daughter '1' is in an infinite open cluster consisting of her own descendents. Show that

$$\phi_p = 2p(1-p)\phi_p + p^2\big(\phi_p^2 + 2\phi_p(1-\phi_p)\big).$$

It can be shown that $\phi_p$ is the maximal solution to this equation. Find $\phi_p$. (One way to obtain a 'merit' mark in this project, though not the only way, is to show that $\phi_p$ *is* the maximal solution.)

Now let $\theta_p$ be the probability that Eve is in an infinite open cluster. Find $\theta_p$ and draw its graph as a function of $p$.

**Question 2** Show that, for $p \leqslant \frac{1}{2}$, there are almost surely no infinite clusters.
How many infinite clusters are there if $\frac{1}{2} < p < 1$? Justify your answer.

This model exhibits a property which is general to percolation models: there exists a critical probability $p_c$ such that for $p < p_c$ there is no infinite cluster and for $p > p_c$ we find at least one infinite cluster.

The region $p > p_c$ is called the *supercritical* region. We can ask questions like: what is $\theta_p$, the probability that a node chosen arbitrarily lies in an infinite open cluster? (In Question 1, in calculating $\theta_p$, we could of course have designated any node to be Eve.)

The region $p < p_c$ is called the *subcritical* region. We ask questions like: how likely are we to observe an open cluster of size $n$? if there is an open cluster of size $n$, what shape is it?

Most of the interesting (unsolved) problems relate to the geometry of open clusters when $p$ is near $p_c$. For example, there are many graphs (e.g. $\mathbb{Z}^3$) where it is not known (but strongly conjectured) that there is no infinite $p_c$-open cluster.

## 3 The square lattice

The square lattice in two dimensions $\mathbb{L}^2$ is a graph with $V = \mathbb{Z}^2 = \{(m,n) : m,n \in \mathbb{Z}\}$. If the distance function is

$$d\big((k,l),(m,n)\big) = |k-m| + |l-n|,$$

the edges of the graph are straight lines connecting nodes which are distance 1 apart.

Let us look at two techniques which will help us estimate the critical probability $p_c$ above which there is an infinite cluster and below which there is none.

### Lower bound

Let us start at the origin. Let $\sigma_n$ be the number of self-avoiding paths (i.e., paths which traverse each edge at most once) of length $n$ leading away from the origin.

**Question 3** Let $\lambda = \limsup_{n\to\infty} \sigma_n^{1/n}$. Show that $\lambda \leqslant 3$. Show further that $p_c \geqslant \lambda^{-1}$.

The actual value of $\lambda$ is an open problem.

### Upper bound

The general behaviour of large clusters in the subcritical region $p < p_c$ is described in the following result. Some notation first: We say $x \leftrightarrow y$ if there exists an open connected path between $x$ and $y$. Define the open sphere $S_n$ to be

$$S_n = \big\{x \in \mathbb{Z}^2 : d(x,0) \leqslant n\big\}.$$

The boundary $\partial S_n$ consists of the nodes where $d(x,0) = n$. Let $P_p(0 \leftrightarrow \partial S_n)$ be the probability that there exists a $p$-open path connecting the origin to some node in $\partial S_n$. It can be shown that, for $p < p_c$, there exists $\psi_p > 0$ such that

$$P_p(0 \leftrightarrow \partial S_n) < e^{-n\psi_p} \quad \text{for all } n. \tag{1}$$

The proof is beyond the scope of this project but can be found in [1, Sections 5.2 and 6.1].

Armed with the above result, we aim to show that $p_c \leqslant \frac{1}{2}$. We do this as follows. Consider the following subgraph $G_n = (V_n, E_n)$ of the square lattice.

$$V_n = \{(k,l) \,:\, 0 \leqslant k \leqslant n,\ 0 \leqslant l \leqslant n-1\}.$$

Let $E_n$ be all the edges in $E$ connecting these nodes. We call $\{(k,l) \in V_n : k = 0\}$ the left boundary, and $\{(k,l) \in V_n : k = n\}$ the right boundary. Let $A$ be the event that some node in the left boundary is connected to the right boundary via a path consisting of open edges.

**Question 4** Show that $P_{1/2}(A) = \frac{1}{2}$. *Hint. You may find it useful to consider the dual graph $\bar{G}_n$, which has nodes at*

$$V_n' = \{(k+\tfrac{1}{2}, l-\tfrac{1}{2}) \,:\, 0 \leqslant k \leqslant n-1,\ 0 \leqslant l \leqslant n\},$$

*and edges joining those nodes which are distance 1 apart, and whose edges are open or closed depending on whether the edges of $G$ are open or closed, in a manner which you should specify.*

**Question 5** By constructing $n$ events $\{A_i\}$ such that $A = \cup A_i$, and using (1), prove that $p_c \leqslant \frac{1}{2}$.

## 4 The Invasion Process

We can try and use computing power to estimate $p_c$ and $\theta_p$. Suppose that at time $n = 0$ you are an invading force standing at the origin. We will call $I_n$ the set of nodes you have invaded by time $n$. At time $n+1$ you invade another node by looking at the edge-boundary of your territory and walking along the edge with the least value of $U_e$ attached to it. Formally, we define

$$\partial I_n = \{e \in E : I_n \overset{e}{\leftrightarrow} \mathbb{Z}^2 \setminus I_n\}.$$

We use the notation $x \overset{e}{\leftrightarrow} y$ to mean that the edge $e$ connects $x$ and $y$. You walk from $I_n$ along the edge $e_n \in \partial I_n$ which satisfies

$$U_{e_n} = \min\{U_f : f \in \partial I_n\},$$

so that $I_{n+1}$ is $I_n$ with the node at the other side of $e_n$ added. It can be shown that, almost surely,

$$\limsup_{n\to\infty} U_{e_n} = p_c. \tag{2}$$

(The proof is not hard, and is outlined at the end of this project.) The advantage of using the sequence $U_{e_n}$ to estimate $p_c$ is that the amount of memory required to store $U_{e_n}$ and to calculate $U_{e_{n+1}}$ is $O(n)$. This is true whether we are working in $\mathbb{L}^2$ or $\mathbb{L}^{47}$.

**Question 6** Implement the invasion process. Describe your algorithm.
Explain in particular why it only requires $O(n)$ storage space to calculate the first $n$ values of $U_{e_n}$, and what it does to ensure it never revisits a vertex.
Comment also on the complexity of the algorithm (the number of time steps needed).

**Question 7** Use your program to estimate $p_c$ for $\mathbb{L}^2$, and explain your method. Estimate $p_c$ for $\mathbb{L}^3$ (which is defined like $\mathbb{L}^2$, but using $\mathbb{Z}^3$ rather than $\mathbb{Z}^2$). Include in your report any appropriate plots.

It is desired to plot $\theta_p$, the density of the infinite cluster.

> **Question 8** Explain how the invasion process can be used to estimate $\theta_p$ simultaneously for all $p$. Produce a plot of $\theta_p$ against $p$ for $\mathbb{L}^2$ by running the simulation for large $n$ several times (at least $n = 5000$, at least 500 times). For what values of $p$ do you expect your plot to be inaccurate? Why?

## Appendix

Here is an outline of the proof of (2).

Let $p > p_c$. Then there exists an infinite $p$-open cluster. Let $T_p$ be the first time that the invasion process hits the cluster (i.e. the first time that the vertex $I_{T_p} \setminus I_{T_p - 1}$ is in the infinite $p$-open cluster). It can be shown that $P(T_p < \infty) = 1$. For all $n \geqslant T_p$, there will always be an edge in $\partial I_n$ with $U_e \leqslant p$. It follows that

$$\limsup_{n\to\infty} U_{e_n} \leqslant p.$$

Since $p > p_c$ was arbitrary, $\limsup_{n\to\infty} U_{e_n} \leqslant p_c$.

Now let $p < p_c$. Suppose that with some probability $\alpha > 0$,

$$\limsup_{n\to\infty} U_{e_n} \leqslant p < p_c.$$

Then we have (with a positive probability) an infinite cluster, with all but finitely many of its edges $p$-open. It follows that (with a positive probability) there exists an infinite $p$-open cluster. This contradicts the definition of $p_c$ as the critical probability.

## References

[1] G.R. Grimmett. *Percolation.* Springer-Verlag, Berlin 1989 and 1999

# 20 Probability

## 20.6 Loss Networks (9 units)

*This project requires knowledge of discrete- and continuous-time Markov chains, covered in the Part IB Markov Chains and Part II Applied Probability courses respectively.*

## Introduction

Nearly a century ago the mathematician Erlang, working for the Copenhagen Telephone Company, devised the first mathematical theories for telecommunications networks. The technology we have now would seem like science fiction to Erlang, yet his insight into the essential structure of networks means that his theorems are just as useful for designing an optically-routed backbone for the Internet as they were for early Danish telephony.

In this project we will deal with certain models for telephone networks, arising from Erlang's work. We consider a network to be a collection of links (cables). Each telephone call occupies a certain amount of space on certain links; for example, a telephone call from a Cambridge college to a London house might occupy 8 kbit/sec of space on the link from the college to the university exchange, and on the link from the university exchange to BT's Cambridge exchange, and on the link from there to a London exchange, and so on. This space is occupied for the duration of the call. The links which comprise the national telephone network only have limited capacity, and when they are full we get a busy signal. Some interesting questions are: What is the probability of a busy signal? How does it depend on the volume of traffic? Can we reduce this probability by strategies such as offering multiple routes for a call?

For further reading see [1].

## 1 The Erlang link

Consider a single link with the capacity to carry $C$ simultaneous calls. Suppose that new calls arrive as a Poisson process of rate $\nu$, that each call lasts for a duration which is exponential with mean 1, and that all call durations are independent of each other and of the arrival process. If a new call arrives when the link is already carrying $C$ calls, then the new call is *blocked.* This system is known as the "Erlang link".

> **Question 1** Set up a continuous-time Markov model for the Erlang link. Calculate the equilibrium probability that there are $i$ calls in progress.

Define $E(\nu, C)$ to be the equilibrium probability that there are $C$ calls in progress.

We say that an arriving call "sees" the system in state $s$ if the system is in state $s$ just before it arrives. The PASTA property (Poisson arrivals see time averages) says that the long-run proportion of arrivals which see the system in state $s$ is equal to the equilibrium probability that the system is in state $s$.

> **Question 2** Show that the PASTA property holds for the Erlang link. *Hint. One approach is to consider the discrete-time Markov chain which records the state of the system after each event, where an event is an attempted arrival or a departure, and to*

*express the equilibrium distribution of this chain in terms of the equilibrium distribution for the continuous-time chain.* Deduce that the long-run proportion of calls which are blocked is $E(\nu, C)$.

**Question 3** Write a program to simulate the Erlang link and to measure the blocking probability. Compare the empirical blocking probability to that given by $E(\nu, C)$ for a range of values of $C$ up to 600 and an appropriate range of values of $\nu$.

## 2 Alternative Routing

Consider now a network of links. For simplicity, suppose that the network is a complete graph on $K$ nodes, i.e., there is a link between every pair of nodes $\{1, \dots, K\}$, and that each link has capacity $C$. Suppose that for every pair of nodes $(a, b)$, calls between $a$ and $b$ arise as a Poisson process of rate $\nu$. It might seem reasonable, in order to reduce the blocking probability, to offer an alternative route if the direct link is full. Specifically, suppose that calls between $a$ and $b$ are routed as follows:

1. If there is spare capacity on the direct link $a \leftrightarrow b$, route the call over that link.
2. Otherwise, pick a new node $c$ uniformly at random from the other $K-2$ nodes. If there is spare capacity on $a \leftrightarrow c$ and on $c \leftrightarrow b$, route the call over these two links.
3. Otherwise, the call is blocked.

We will call this the "Alternative Routing" system. One way (but not the only way) to obtain a 'merit' mark in this project is to prove that the Alternative Routing system satisfies the PASTA property.

As you can see, it is possible in principle to set up a Markov process model for this system, and thereby to calculate the equilibrium distribution; but the number of states is so large that, even for moderate $K$, it is not computationally practical to do so. Instead, we can use a famous approximation called the *Erlang fixed point approximation*, which is that blocking occurs independently on different links. This leads to the formula

$$B = E(\nu + 2\nu B(1-B), C) \tag{1}$$

where $B$ is the probability that an incoming call cannot be routed on its chosen direct link.

**Question 4** Give a careful intuitive explanation for (1). In what sense could $\nu + 2\nu B(1-B)$ be called the "offered load" on a link?

**Question 5** Demonstrate numerically (e.g., by plotting an appropriate graph) that for some values of $C$ and $\nu$ this equation has a unique solution, and that for other values it has multiple solutions.

Now pick $C = 600$ and choose $\nu$ such that (1) has multiple solutions.

**Question 6** Write a program to simulate the Alternative Routing system. Explain clearly and concisely the algorithm you have used. Run your simulation, for $K = 5$, and record the cumulative count of the number of calls blocked. Plot a graph which shows this count as a function of time. *Programming hint. Run your simulation for at least a million transitions of the Markov process; this should take less than 10 minutes on a modern computer. You may find it convenient to write a small subsample of your simulation output to a file, then use Excel or some other program to plot the result.*

You should find, from your simulation, that the system spends some of the time in a high-blocking regime and some of the time in a low-blocking regime, corresponding to solutions of (1).

**Question 7** Give an intuitive explanation for why there are multiple solutions. It is a standard result from Markov chain theory that this Markov process has a unique equilibrium distribution; comment briefly on how this result relates to the existence of multiple solutions.

**Question 8** You should observe that there is one solution to (1) which is *not* reflected in your simulations. By considering fixed points of the map $B \leftarrow E(\nu + 2\nu B(1-B), C)$, suggest why this is so.

**Question 9** Use the Erlang fixed-point approximation to find the probabilities that an incoming call is blocked in the high-blocking regime and in the low-blocking regime. How do these compare to a network without alternative routing, i.e., in a network in which a call may only be routed on the direct link?

## 3 Trunk reservation

A telecoms operator would be alarmed at the situation you investigated in the last section, and would seek to control the network so that it stays in the low-blocking state. One way to do this is with a technique called "trunk reservation". We modify the call admission procedure, so that calls arising between $a$ and $b$ are routed as follows:

1. Consider the direct link $a \leftrightarrow b$. If this link has spare capacity, route the call over it.
2. Otherwise, pick a new node $c$ uniformly at random from the other $K-2$ nodes, and consider the two links $a \leftrightarrow c$ and $c \leftrightarrow b$. If on each of these links the number of calls in progress is less than $C-s$, then route the call over these two links.
3. Otherwise, the call is blocked.

The parameter $s > 0$ is known as the *trunk reservation parameter.*

**Question 10** Develop a fixed-point approximation for this system. *Hint. First set up a suitable Markov model for the number of calls in progress on a single link with two classes of traffic.*

**Question 11** Compare the blocking probability to those you found in Question 9. Has trunk reservation improved matters? How large should the trunk reservation parameter be?

## References

[1] F. P. Kelly, *Network Routing*, Philosophical Transactions of the Royal Society series A, 1991. `http://www.statslab.cam.ac.uk/~frank/loss/`

# 23 Astrophysics

## 23.5 Ionization of the Interstellar Gas near a Star (8 units)

*No knowledge of Astrophysics is assumed or required: all relevant equations are defined and explained in the project itself.*

## 1 Introduction

The interstellar medium surrounding a hot star is ionized by the radiation from the star. In this project we calculate the size of the ionized region, and gain some insight into how its structure depends on the nature of the radiation from the star.

Assuming that there is a uniform, static, constant temperature gas surrounding a spherically symmetric star provides a good approximation, which keeps the essentials of the situation without allowing unnecessary distractions. Each gas element is assumed to be in ionization equilibrium, so the ionization rate for each atom is balanced by recombinations. We assume that the sole source of ionization is by absorption of radiation from the star giving a bound electron enough energy to escape from the atom. Recombination occurs when a free electron is captured by an ion with the creation of a photon. Since hydrogen is the most common element in the Universe, its behaviour will dominate in most cases, so we consider only a pure hydrogen interstellar medium.

## 2 Radiation from a star

The radiation from the star is specified as $L_\nu$, the total energy output from the star per unit frequency $\nu$ per unit time, so the luminosity per unit frequency interval $L_\nu$ is expressed in W Hz$^{-1}$. The total energy output radiated from the star is the integral of this quantity over all frequencies, or $L = \int_0^\infty L_\nu \mathrm{d}\nu$. In some cases a star spectrum is reasonably well approximated by a black-body, so the radiation flux in a frequency interval $\mathrm{d}\nu$ at frequency $\nu$ emerging per unit area from the surface of the star where the temperature is $T_*$ is given by

$$I_\nu \mathrm{d}\nu = \frac{2\pi h}{c^2}\frac{\nu^3}{\exp(\frac{h\nu}{kT_*}) - 1}\mathrm{d}\nu,$$

where $h$ is Planck's constant, $c$ is the velocity of light, and $k$ is Boltzmann's constant (given below). Thus for a star of radius $R$

$$L_\nu = 4\pi R^2 \frac{2\pi h}{c^2}\frac{\nu^3}{\exp(\frac{h\nu}{kT_*}) - 1}.$$

**Question 1** The sun has radius $R = 6.96 \times 10^8$ metres, and the total luminosity $L = 3.90 \times 10^{26}$ W. Show, using the above equation, that its surface temperature is close to 5800 K.

A 7 solar mass star has a surface temperature $T_* = 20,000$K and a luminosity $L = 4.0 \times 10^{29}$W. What is its radius? What is the radius of a 12 solar mass star which has a surface temperature $T_* = 25,000$K and a luminosity $L = 4.0 \times 10^{30}$W.

## 3 Equations for ionization and recombination

An element of gas at a distance $r$ from the (centre of the) star will receive $\frac{L_\nu}{4\pi r^2}$ W m$^{-2}$ Hz$^{-1}$ if the radiation is not attenuated by any material between it and the star. The number of photons received per second per unit frequency interval is $L_\nu$ divided by the energy, $h\nu$ per photon. If these photons have frequency $\nu \geqslant \nu_0 = 3.29 \times 10^{15}$Hz they have enough energy to separate a hydrogen atom into a proton and an electron. The rate at which this happens depends on the frequency of the radiation, and is given by the photon rate per second $\times$ an absorption coefficient $a_\nu$ per hydrogen atom. For an energy flux $I_\nu(r)$, so photon flux $I_\nu(r)/h\nu$, the rate of ionization per unit volume is

$$n_{\mathrm{H}^0} \int_{\nu_0}^{\infty} \frac{I_\nu(r)}{h\nu} a_\nu \mathrm{d}\nu,$$

where $n_{\mathrm{H}^0}$ is the number density of neutral hydrogen atoms (i.e. number of neutral hydrogen atoms per cubic metre). The coefficient $a_\nu$ depends only on the atomic species being considered, and for neutral hydrogen

$$\begin{aligned} a_\nu &= a_{\nu_0}\left(\frac{\nu_0}{\nu}\right)^3 \quad \text{for } \nu \geqslant \nu_0 \\ a_\nu &= 0 \ \text{ for } \nu < \nu_0, \end{aligned}$$

where $a_{\nu_0} = 6.3 \times 10^{-22}$ m$^2$. If absorption can occur at the gas element, it can also occur in all the elements between the star and the one under consideration. As we go along a path d$r$ the radiation is attenuated, so at a given frequency $\nu$,

$$\frac{\mathrm{d}I_\nu}{\mathrm{d}r} = -n_{H^0} a_\nu I_\nu,$$

and so

$$I_\nu(r) = I_\nu(R) e^{-\tau_\nu},$$

where $\tau_\nu$ is defined by

$$\frac{\mathrm{d}\tau_\nu}{\mathrm{d}r} = n_{H^0}(r) a_\nu$$

and $\tau_\nu(R) = 0$. $\tau_\nu$ is referred to as the *optical depth* at the frequency $\nu$. There is no redistribution of the photons in frequency (they are effectively absorbed from the point of view of this calculation), so we can determine $\tau_\nu$ from

$$\frac{\mathrm{d}\tau_{\nu_0}}{\mathrm{d}r} = n_{H^0}(r) a_{\nu_0}$$

and

$$\tau_\nu = \tau_{\nu_0}\left(\frac{\nu_0}{\nu}\right)^3.$$

This absorption of radiation and the ionization it causes must be balanced by recombination of protons and electrons at the same rate per unit volume. This rate depends on the number densities of the protons ($n_p$) and the electrons ($n_e$), their relative velocity and a velocity-dependent cross-section for the interaction which has to be integrated over the velocity distribution at whatever temperature the gas is at. These velocity-dependent terms are combined into recombination coefficients $\alpha_B$ which are tabulated for various gas temperatures $T$:

| $T$ (K) | $\alpha_B$ |
|---|---|
| 5 000 | $4.54 \times 10^{-19}$ m$^3$ s$^{-1}$ |
| 10 000 | $2.59 \times 10^{-19}$ |
| 20 000 | $2.52 \times 10^{-19}$ |

Then we are in a position to write down the ionization balance equation

$$n_{\mathrm{H}^0} \int_{\nu_0}^{\infty} \frac{L_\nu}{4\pi r^2 h\nu} a_\nu e^{-\tau_\nu} \mathrm{d}\nu = n_p n_e \alpha_B(T), \tag{1}$$

and subsidiary equations

$$\frac{\mathrm{d}\tau_{\nu_0}}{\mathrm{d}r} = n_{H^0}(r) a_{\nu_0}, \tag{2}$$

$$\tau_\nu = \tau_{\nu_0} \left(\frac{\nu_0}{\nu}\right)^3, \tag{3}$$

$$n_p = n_e, \tag{4}$$

$$n_p + n_{H^0} = n_H, \tag{5}$$

which govern the ionization balance for the hydrogen-filled interstellar medium near a star.

[We have omitted a step here, and assumed that the energy released by recombination does not give rise to an ionizing photon. Often it does, but we assume that all this does is cause another ionization nearby until recombination occurs to an upper level in the hydrogen atom, and then the energy is lost from the system through radiation at frequencies less than $\nu_0$. The net result is a change in the effective recombination coefficient, to the one quoted here. Those interested in more details will find them in Osterbrock's book (1989)]

## 4 Ionization near stars

A typical interstellar medium hydrogen number density is $n_H = 10^6$ m$^{-3}$, and a typical temperature is $T = 10^4$K.

**Question 2** Write a program to solve the ionization equations to obtain the neutral hydrogen and proton densities as a function of distance from the centre of the star, assuming that the interstellar gas has a constant temperature and density. Note that the coefficients involved have large powers of 10, so for some compilers a rescaling of variables may be desirable. Describe any transformations used. Are there any advantages to using $\tau_{\nu_0}$ instead of $r$ as the radial coordinate?

Now apply this program to some realistic cases:

**Question 3** Determine the ionization and neutral fractions of hydrogen as a function of distance from the star with surface temperature $T_* = 20,000$K and luminosity $4 \times 10^{29}$W for an interstellar gas density $n_H = 10^6$ m$^{-3}$ and $T = 10^4$K, and plot the results. Show in particular that at some radial distance, $r_1$, there is quite a sharp transition from the gas being mostly ionized to mostly neutral. What is the value of $r_1$, the distance from the centre of the star where $n_p = n_{H^0}$? Give your answers to two significant figures.

Repeat the calculation for gas of the same temperature and density but with the 12 solar mass star at the centre and again with the Sun as the central star. Also, compute the ionization fractions for all three stars in the cases that the gas temperature is 5000K and 20, 000K. Comment on any similarities or differences between the three cases.

Provide plots of the nine cases (three stellar masses and three gas temperatures) and a table containing the values of $r_1$ for each case.

### 4.1 An approximation to $r_1$

Equation (1) can be integrated over volume from $r = 0$ to $r = \infty$ by using the definition of $\tau_\nu$ to replace d$r$ and assuming that the recombination term is well approximated by $n_p = n_e = n_H$ for $r \leqslant r_1$ and $n_p = n_e = 0$ for $r > r_1$. This $r_1$ is called the Strömgren radius.

**Question 4** Show that, under these circumstances,

$$Q(H) \equiv \int_{\nu_0}^{\infty} \frac{L_\nu}{h\nu} \mathrm{d}\nu = \frac{4\pi}{3} r_1^3 n_H^2 \alpha_B,$$

where $Q(H)$ is the total number of ionizing photons emitted by the star per second.

Calculate the values of $Q(H)$ for the cases given above and compare the $r_1$ determined from this approximation with the values you have computed for $T = 10,000$K.

## 5 The effect of a quasar on the host galaxy

The energy output from a quasar is very different from that of a star, both in intensity and frequency dependence. A bright quasar has energy output of $4 \times 10^{39}$W, and resides in the centre of a galaxy of radius $3 \times 10^{20}$m (= 30,000 light years).

**Question 5** If the frequency dependence of the quasar luminosity at frequencies $\nu \geqslant \nu_0$ is given by

$$L_\nu = 10^{24} \left(\frac{\nu}{\nu_0}\right)^{-1.4} \exp\left(-\frac{\nu}{10\nu_0}\right),$$

determine whether or not any of the interstellar gas in the galaxy has neutral fraction $n_{H^0}/n_H > 0.5$. (Assume that the gas temperature is $10^4$K, and density $10^6$ hydrogen atoms per cubic metre. Assume also that there is no gas within $10^{18}$m of the quasar position, so start the computation there.)

Tabulate the results for the hydrogen neutral fraction as a function of distance from the quasar.

*Useful constants:*

| | |
|---|---|
| velocity of light | $c = 2.998 \times 10^8 \text{ms}^{-1}$ |
| Planck's constant | $h = 6.626 \times 10^{-34}$Js |
| Boltzmann's constant | $\text{k} = 1.381 \times 10^{-23}\text{JK}^{-1}$ |

## References

[1] Osterbrock, D.E., 1989. *Astrophysics of Gaseous Nebulae and Active Galactic Nuclei* University Science Books: Mill Valley, CA (Especially chapter 2).

# 23 Astrophysics

## 23.6 Accretion Discs (8 units)

*No knowledge of astrophysics is assumed or required in this project. All relevant equations are defined and explained in the project.*

## 1 Fluid Equations

Accretion discs are composed of fluid orbiting a central object. They evolve viscously so that matter falls inwards while angular momentum drifts outwards. Accretion discs are found in binary star systems, around forming stars and in active galactic nuclei. We use cylindrical polar coordinates $(R, \phi, z)$ and assume axisymmetry for this project. When the central object dominates the gravitational field the angular velocity of the matter in the disc is Keplerian so that

$$\Omega = \left(\frac{GM}{R^3}\right)^{1/2}, \tag{1}$$

where $M$ is the mass of the central object and $G$ is the gravitational constant. The disc is made up of annuli of matter lying between $R$ and $R + \Delta R$ with mass $2\pi R \Delta R \Sigma$, where $\Sigma(R, t)$ is the surface density (with dimensions $\mathrm{ML}^{-2}$) of the disc at time $t$.

The equation describing conservation of mass is

$$R\frac{\partial \Sigma}{\partial t} + \frac{\partial}{\partial R}(R\Sigma V_{\mathrm{R}}) = 0, \tag{2}$$

where $V_{\mathrm{R}}$ is the radial velocity in the disc. Conservation of angular momentum gives us

$$R\frac{\partial(\Sigma R^2 \Omega)}{\partial t} + \frac{\partial}{\partial R}(R\Sigma V_{\mathrm{R}} R^2 \Omega) = \frac{1}{2\pi}\frac{\partial \Gamma}{\partial R}, \tag{3}$$

where the viscous torque

$$\Gamma = 2\pi R \nu \Sigma R^2 \Omega' \tag{4}$$

where $\Omega' = \frac{d\Omega}{dR}$ and $\nu(R, \Sigma)$ is the viscosity in the disc.

**Question 1**

Show that, for $\Omega$ independent of time,

$$R\frac{\partial \Sigma}{\partial t} = -\frac{\partial}{\partial R}\left[\frac{1}{2\pi (R^2\Omega)'}\frac{\partial \Gamma}{\partial R}\right]. \tag{5}$$

and hence, using equation 1, for a Keplerian disc that

$$\frac{\partial \Sigma}{\partial t} = \frac{1}{R}\frac{\partial}{\partial R}\left[R^{1/2}\frac{\partial}{\partial R}(3\nu\Sigma R^{1/2})\right] \tag{6}$$

This is the basic equation that governs the evolution of the surface density of a Keplerian accretion disc, which we shall assume for the rest of this project. The viscosity may be a function of $\Sigma$, $R$ and $t$ and so this equation may be non-linear.

## Question 2

The mass accretion rate $\dot{m}$ through the disc is

$$\dot{m}(R) = -2\pi R \Sigma V_{\mathrm{R}} \tag{7}$$

Show that

$$V_{\mathrm{R}} = -\frac{3}{\Sigma R^{1/2}} \frac{\partial}{\partial R}(\nu \Sigma R^{1/2}). \tag{8}$$

In a steady state disc the accretion rate through the disc is constant. Find, analytically, the steady state solution for $\nu\Sigma$ from equation (6). Use the inner boundary condition $\Sigma = 0$ at $R = R_{\mathrm{in}}$ where $\nu$ is finite at $R = R_{\mathrm{in}}$. Plot $\nu\Sigma$ in units of $\dot{m}$ against $R/R_{\mathrm{in}}$ in the range $1 < R/R_{\mathrm{in}} < 100$.

We can make the problem dimensionless by setting $r = R/R_0$, $\tau = t/t_0$, $\sigma(r,\tau) = \Sigma/\Sigma_0$, and $\eta(r,\sigma) = \nu/\nu_0$. Find a condition for $t_0$ such that equation (6) remains the same for these dimensionless variables. Assume that the accretion rate is small enough that $M \approx$ const.

## Question 3

Use the substitution $X = r^{1/2}$ in equation 6 to derive

$$\frac{\partial f}{\partial \tau} = \frac{\partial^2 g}{\partial X^2} \tag{9}$$

where $f$ and $g$ are to be determined. We consider $X$ to be discrete with values $X_i$ where $i = 1, 2, \ldots 100$ and $X_1 = \Delta X$ where $\Delta X$ is a constant. We let $X_{i+1} = X_i + \Delta X$ and time be $\tau_n$ where $n = 1, 2, \ldots$ with $\tau_{n+1} = \tau_n + \Delta\tau$. We have $\tau_1 = 0$ and we let $f_i^n = f(X_i, \tau_n)$ and $g_i^n = g(X_i, \tau_n)$.

Show that equation (9) can be represented by the difference equation

$$f_i^{n+1} = f_i^n + \frac{\Delta\tau}{(\Delta X)^2}(g_{i+1}^n - 2g_i^n + g_{i-1}^n). \tag{10}$$

## Question 4

Write a program to solve equation (9) taking $\eta = 1$ and with the boundary conditions $\sigma(r_{\mathrm{in}}, \tau) = 0$ and $\sigma(r_{\mathrm{out}}, \tau) = 0$. Set $r_{\mathrm{in}} = 0.0004$ and $r_{\mathrm{out}} = 4$ and use 100 grid points equally spaced in $X$ between $X = 0.02$ and $X = 2$. For formal stability the timestep must satisfy

$$\Delta\tau \leqslant \frac{1}{2}(\Delta X)^2 \frac{f}{g} \tag{11}$$

at all points in the disc. However you may find that you need to use something smaller. Evolve from an initial mass distribution

$$\sigma(r, 0) = \exp\left(-\frac{(r^{1/2} - 1)^2}{0.001}\right). \tag{12}$$

Plot the initial surface density, $\sigma(r, 0)$, against $r$. Plot $\sigma$ against $r$ at times $\tau = 0.002$, 0.008, 0.032, 0.128 and 0.512 on the same axes.

**Question 5**

Adapt your program so that you can find the height and position of the peak in the surface density. Make a table of the surface density at the peak and the position of the peak at the times used in question 4. Furthermore, adapt your program to plot the time evolution of the total angular momentum in the disc (normalised to its value at $t = 0$), and the position of the peak angular momentum surface density ($R^2\Omega\Sigma$) as a function of time.

Comment on the difference in behaviour between the surface density and angular momentum.

**Question 6**

The evolution of a particle in the disk is given by $dR/dt = V_{\mathrm{R}}(R, t)$ where $V_{\mathrm{R}}$ is given by equation 8. Rewrite equation 8 in a form similar to equation 10 so that you can adapt the code written for question 4 to plot radial velocity (in dimensionless units) as a function of radius for the timesteps used in question 4 on the same figure, taking care with choice of axis to show where this is positive and negative.

**Question 7**

Use the radial velocities found by your code to follow the evolution of particles' orbits, and plot that evolution up to $\tau = 1$ for particles initially at $r_0 = 0.9, 0.95, 1.0, 1.05, 1.1$. Plot the maximum radius attained by a particle as well as the time it took to reach that distance and, for those particles that do so, the time it takes to reach a boundary as a function of its initial radius for the range $r_0 = 0.9 - 1.1$.

**Question 8**

Use the results from question 7 to work out the range of initial radii $r_0$ that have reached the inner boundary by $\tau = 0.512$ and hence, using equation 12, the fraction of the initial mass that has reached the inner boundary by this time. Also do the corresponding calculation for the outer boundary. Compare these values with the fraction of the initial mass that remains at that time that you derive using the surface density profile from question 4.

## MATLAB Specific Issues

Use of the MATLAB in-build function GRADIENT is prohibited, as its algorithm or accuracy is not documented.