



UNIVERSITY OF  
CAMBRIDGE

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

Computational Projects  
2024-25

*CATAM*

*Mathematical Tripos Part II*  
*Astrophysics Tripos Part II*  
**Computational Projects**

July 2024

*Edited by the Computational Projects Assessors Committee*

*Course Director: Prof J.R. Taylor*

*Assistant Course Director: Dr D. Frank*

**Computer-Aided Teaching of All Mathematics**

**Faculty of Mathematics**  
**University of Cambridge**

# Contents

	<i>Units</i>
<b>Introduction</b>	
<b>1 Numerical Methods</b>	
1.3 Parabolic Partial Differential Equations	7
1.8 Hyperbolic Partial Differential Equations	8
<b>2 Waves</b>	
2.7 Soliton Solutions of the KdV Equation	7
2.10 Phase and Group Velocity	8
<b>3 Fluid and Solid Mechanics</b>	
3.1 Boundary-Layer Flow	6
3.8 Wind-forced Ocean Currents	10
<b>4 Dynamics</b>	
4.3 The Rotating Top	6
<b>5 Quantum Mechanics</b>	
5.1 Band Structure	8
<b>6 Electromagnetism</b>	
6.1 Diffraction Pattern due to a Current Strip	7
<b>7 Mathematical Methods</b>	
7.5 Padé Approximants	8
7.6 Insulation	10
<b>9 Operational Research</b>	
9.3 Protein Comparison in Bioinformatics	8
9.5 The Google PageRank Algorithm	5
<b>10 Statistics</b>	
10.3 Bootstrap Estimation of Standard Error	5
10.15 Variable Selection and the Bias-Variance Tradeoff	8
<b>11 Statistical Physics</b>	
11.2 The Ising Model in One Dimension	10

<b>12</b>	<b>Nonlinear Dynamics &amp; Dynamical Systems</b>	
12.8	A Nonlinear Map and the Dynamics of Hydrogen Atoms in Electric Fields	7
12.9	Differential Equations for Nonlinear Oscillators	10
<b>14</b>	<b>General Relativity</b>	
14.1	Particle or Photon Orbits near a Black Hole	8
14.7	Gravitational Radiation from Point Masses in a Keplerian Orbit	8
<b>15</b>	<b>Number Theory</b>	
15.3	Positive Definite Binary Quadratic Forms	9
15.6	Computing Roots Modulo $p$	7
<b>16</b>	<b>Algebra</b>	
16.8	The Character Table of a Finite Group	9
16.9	Resultants and Resolvants	8
<b>17</b>	<b>Combinatorics</b>	
17.5	Matchings	6
17.7	Graph Planarity	10
<b>19</b>	<b>Communication Theory</b>	
19.2	Information Content of Natural Language	4
<b>20</b>	<b>Probability</b>	
20.1	The Percolation Model	7
20.2	Importance Sampling and Fast Simulation	5
<b>23</b>	<b>Astrophysics</b>	
23.1	Generating a Consistent Self-Gravitating System	8
23.4	Stellar Structure	8

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 2024-25. At the time of writing (July 2024) the booklet for the academic year 2023-24 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 at the start of 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,<sup>1</sup> but *you are free to write your programs in any computing language whatsoever*. Python, Julia,<sup>2</sup> R,<sup>3</sup> C, C++, Mathematica,<sup>4</sup> Maple<sup>5</sup> 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.<sup>6</sup>

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.

---

<sup>1</sup> Except where an alternative is explicitly stated, e.g. see footnotes 3 and 5.

<sup>2</sup> 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/>.

<sup>3</sup> 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/>.

<sup>4</sup> 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/>

<sup>5</sup> 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.

<sup>6</sup> 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 than 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  $M \leq N - 30$  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.<sup>7</sup>

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*.<sup>8</sup> 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

---

<sup>7</sup> Needless to say, if you submit fewer than 30 units no upwards scaling applies.

<sup>8</sup> 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  $O(f(n))$  or  $\Omega(f(n))$  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.<sup>9</sup> 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.<sup>10</sup> 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.  $\text{\LaTeX}$ , *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  $\text{\LaTeX}$  (or, if they are of an older generation,  $\text{\TeX}$ ), e.g. this document is written in  $\text{\LaTeX}$ . However, please note that although  $\text{\LaTeX}$  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

---

<sup>9</sup> Reports of projects with fewer/more units might be slightly shorter/longer.

<sup>10</sup> Recall that graphs should not as a rule be printed one to a page.

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

*L<sup>A</sup>T<sub>E</sub>X*. If you decide to use L<sup>A</sup>T<sub>E</sub>X, you will probably want to install it on your own personal computer. This can be done for free. For recommendations of T<sub>E</sub>X distributions and associated packages see

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

*Front end*. In addition to a T<sub>E</sub>X distribution you will also need a front-end (i.e. a ‘clever editor’). A [comparison of T<sub>E</sub>X editors](#) is available on WIKIPEDIA; below we list a few of the more popular T<sub>E</sub>X editors.

*T<sub>E</sub>Xstudio*. For Windows, Mac and Linux users, there is T<sub>E</sub>Xstudio. The proT<sub>E</sub>Xt distribution, based on MiK<sub>T</sub><sub>E</sub>X, includes the T<sub>E</sub>Xstudio front end.

*T<sub>E</sub>Xworks*. Again for Windows, Mac and Linux users, there is T<sub>E</sub>Xworks. The MiK<sub>T</sub><sub>E</sub>X distribution includes T<sub>E</sub>Xworks.

*T<sub>E</sub>XShop*. Many Mac aficionados use T<sub>E</sub>XShop. To obtain T<sub>E</sub>XShop and the T<sub>E</sub>XLive distribution see <http://pages.uoregon.edu/koch/texshop/obtaining.html>.

*T<sub>E</sub>XnicCenter*. T<sub>E</sub>XnicCenter 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 L<sup>A</sup>T<sub>E</sub>X as necessary.

*Learning L<sup>A</sup>T<sub>E</sub>X*. A *Brief L<sup>A</sup>T<sub>E</sub>X Guide for CATAM* is available for download from

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

- The L<sup>A</sup>T<sub>E</sub>X 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 L<sup>A</sup>T<sub>E</sub>X* page; see

[http://www.eng.cam.ac.uk/help/tpl/textprocessing/LaTeX\\_intro.html](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<sup>11</sup>

<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](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*.

---

<sup>11</sup> 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 2024 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](mailto: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](mailto: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](mailto: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 you 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 Statutes and Ordinances of the University<sup>12</sup>

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

---

<sup>12</sup>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*<sup>13</sup>. 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 16th June 2025), 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.

---

<sup>13</sup> For more information see

[https://www.plagiarism.admin.cam.ac.uk/files/investigative\\_2016.pdf](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

**Thursday 1st May 2025, 4pm.**

Self-certified extensions may be obtained for a period of up to 7 days. A form to apply for an extension will be made available on Moodle at the start of the submission period. Students must inform their college Tutor that they are applying for an extension *before* filling out the extension request form. Whenever possible, students should apply for an extension before the original submission deadline. It will not be possible to apply for a self-certified extension later than 7 days after the original submission deadline. We strongly encourage all students to complete and turn-in their work well in advance of the original submission deadline to allow time to deal with any issues arising during submission and avoid impacts on other coursework and revision.

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 (either the standard or extended deadline, as appropriate).

## 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.

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.<sup>14</sup> 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](mailto: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.

---

<sup>14</sup> See <https://password.csx.cam.ac.uk/forgotten-passwd>.

## A Appendix: Example Submission Form

### PART II MATHEMATICAL TRIPOS 2024-25

Computational Projects 2024

#### 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<sup>15</sup> 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.*

---

<sup>15</sup> 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
<b>Total Credit Units</b>		

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

## 1.3 Parabolic Partial Differential Equations (7 units)

*Part II Numerical Analysis is useful but not essential, since the required background can readily be found in references [1, 2, 3], and elsewhere.*

### 1 Formulation

For times  $0 \leq t < \infty$  we wish to solve the diffusion equation

$$\theta_t = \theta_{xx}$$

on the interval  $0 \leq x \leq 1$ , with boundary conditions

$$\theta(0, t) = f(t) \quad \text{and} \quad \theta(1, t) = 0 \quad \text{for } 0 \leq t < \infty, \quad \text{where } f(t) = t(1 - t),$$

and with initial condition

$$\theta(x, t) = 0 \quad \text{for } t \leq 0, \quad 0 \leq x \leq 1.$$

This is the (non-dimensionalised) initial-value problem for the conduction of heat down a bar when the temperature of one end varies in time. The aim is to study the performance of three simple finite-difference methods applied to this problem, for which the numerical solutions can be compared with an analytic one.

## 2 Analytic solution

### Question 1

- (i) To find an analytic solution of the problem first write

$$\theta(x, t) = f(t)(1 - x) + \phi(x, t).$$

Next find the governing equation, boundary conditions and initial condition for  $\phi(x, t)$ . Thence, with justification, solve for  $\phi$  in terms of a Fourier sine series in  $x$ .

- (ii) Deduce, either from the Fourier sine series or otherwise, that as  $t \rightarrow \infty$

$$\phi(x, t) \rightarrow \alpha(x)t + \beta(x), \tag{1}$$

where the functions  $\alpha(x)$  and  $\beta(x)$  are to be identified.

- (iii) Write a program to compute the analytic solution by summing a finite number of terms of the series, or otherwise.
- (iv) Plot  $\theta$  against  $x$  at a few judiciously chosen values of  $t$  to illustrate the evolution in time.
- (v) How have you satisfied yourself that the solution has been computed to ‘sufficient’ accuracy?
- (vi) Discuss the evolution of the temperature in terms of the physics.



### 3 Numerical Methods

Divide  $0 \leq x \leq 1$  into  $N$  intervals, each of size  $\delta x \equiv 1/N$ . The aim is to march the solution forward in time for various time steps  $\delta t$ . We consider three schemes.

- (i) Approximate  $\theta_t$  by a forward difference in time and  $\theta_{xx}$  by a spatial central difference at the current time, which gives the numerical scheme

$$\frac{\theta_n^{m+1} - \theta_n^m}{\delta t} = (\delta^2 \theta)_n^m \equiv \frac{\theta_{n+1}^m - 2\theta_n^m + \theta_{n-1}^m}{(\delta x)^2},$$

where  $\theta_n^m$  is an approximation to  $\theta(n\delta x, m\delta t)$ .

- (ii) Approximate  $\theta_t$  instead by a central difference in time, so that

$$\frac{\theta_n^{m+1} - \theta_n^{m-1}}{2\delta t} = (\delta^2 \theta)_n^m.$$

In this case you will need scheme (i) in order to make the first step.

- (iii) Modify scheme (i) to

$$\frac{\theta_n^{m+1} - \theta_n^m}{\delta t} = \rho (\delta^2 \theta)_n^{m+1} + (1 - \rho) (\delta^2 \theta)_n^m$$

with  $0 < \rho \leq 1$ . This is now an *implicit* method, and at each step ( $N + 1$ ) simultaneous equations have to be solved for the  $\theta_n^{m+1}$ .

*Remarks*

- (a) The matrix of the simultaneous equations is tridiagonal. Therefore the system may be solved quickly and efficiently by exploiting the sparsity. Your code *should make use of the sparsity*, e.g. the matrix should be stored in an efficient way, and needless multiplications by zero avoided. If you are using MATLAB then `help sparse`, `help spdiags` and `help speye` should help.
- (b) You can check that aspects of your program are working by setting  $\rho = 0$  and comparing with the output of scheme (i).

**Question 2** It is convenient to introduce the Courant number  $\nu = \delta t / (\delta x)^2$ .

- (i) First run each finite-difference scheme with  $N = 5$  and  $\nu = \frac{1}{2}$  and, in the case of scheme (iii),  $\rho = \frac{1}{2}$ . Plot the solution for representative times. In particular, tabulate and plot the numerical solution  $\theta_n^m$ , the analytic solution  $\theta(n\delta x, m\delta t)$  and the error  $\theta_n^m - \theta(n\delta x, m\delta t)$  at  $t = 0.1$ .
- (ii) Next investigate a range of values of your choice for the parameters  $\nu$  (for all schemes) and  $\rho$  (for scheme (iii)) and describe the results. You might like to start by considering  $\nu = \frac{1}{12}, \frac{1}{6}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}$  and 1, and  $N = 5, 20, 80$ . In the case of scheme (iii) also consider  $\delta t = \mu\delta x$  (i.e.  $\nu = \mu/\delta x$ ) for appropriate values of the constants  $\mu$  and  $\rho$ .
- (iii) Discuss the accuracy and the stability of each scheme, and how these properties vary with  $N$ ,  $\nu$  and  $\rho$ . For instance, are your results consistent with the theoretical order of accuracy of each scheme, e.g. see [1, 2, 3]? Statements about accuracy and stability should be supported by *selective* reference to your numerical results, displayed as short tables and/or graphs. Relevant theoretical results should be cited *briefly*.

Comment on, and explain, any interesting features, e.g. do you notice anything about the error in the case of scheme (i) with  $\nu = \frac{1}{6}$ , scheme (iii) with particular choices of  $\rho$  and  $\nu$ , and scheme (iii) with  $\rho = \frac{1}{2}$  and  $\delta t = \mu\delta x$ ?

- (iv) Explain, *with justification*, which scheme and parameter values you would recommend to achieve a given level of accuracy using the *least* computing resources. In particular, you should consider the total operation count to achieve a given level of accuracy.
- (v) For your recommended scheme and parameter values, demonstrate that the numerical solution tends to the asymptotic limit (1) as  $t \rightarrow \infty$ .

## References

- [1] Ames, W.F. *Numerical Methods for Partial Differential Equations*, Academic Press.
- [2] Iserles, A. *A First Course in the Numerical Analysis of Differential Equations*, CUP.
- [3] Smith, G.D. *Numerical Solution of Partial Differential Equations: Finite Difference Methods*, OUP.

# 1 Numerical Methods

## 1.8 Hyperbolic Partial Differential Equations (8 units)

This project concerns shock formation and propagation in nonlinear hyperbolic equations. While this project is largely self-contained, knowledge of the Part II Waves and Part II Numerical Analysis courses is helpful. You might also find it helpful to refer to one or more of the following textbooks: Billingham & King [1], Lighthill [2], Renardy & Rogers [3] or Whitham [5].

### 1 Background

The Euler equations of compressible fluid dynamics allow for the development of some interesting non-linear features; for example *shocks* (or sonic booms) and *rarefaction fans*. A useful model equation both for furthering our understanding of such solutions and for developing numerical methods for the Euler equations is

$$u_t + f(u)_x = 0, \quad f(u) = \frac{1}{2}u^2, \quad (1)$$

known variously as the kinematic wave equation or the inviscid Burger's equation. This equation's interest lies in the fact it possesses a non-linear flux term proportional to the square of the basic variable  $u$ , identical to the convection term present in the Euler equations.

**Question 1** Show analytically that  $u$  is constant along the characteristic curves of (1). A 'shock' develops at the point in space and time where characteristic curves intersect. In the context of (1) and with the aid of characteristic diagrams (sketch only\*), clearly state the condition required for a shock to form and provide examples of initial conditions with qualitatively different behaviour.

Consider the following discontinuous initial condition:

$$u(x, 0) = u_0(x) = \begin{cases} u_1 & x < 0, t = 0 \\ u_2 & x > 0, t = 0 \end{cases}$$

with  $u_1 < u_2$ . Obtain and sketch a solution to (1) satisfying this initial condition and discuss the possibility for shock formation in this case.

In the numerical work to follow we wish to solve (1) subject to the following initial condition

$$u(x, 0) = \begin{cases} -1 & x < 1, \\ 1/2 & 1 \leq x < 2, \\ 0 & x \geq 2, \end{cases} \quad (2)$$

on the domain  $x \in [0, 3]$ . The boundary conditions at  $x = 0$  and  $x = 3$  should be taken to be those of *out-flow*.

---

\* Whereas almost all graphs, including labels, annotations, etc., need to be computer-generated, this is one of the relatively few cases where a scanned hand-drawing is acceptable for electronic submission.

## 2 Numerical Fundamentals

We divide our domain of interest into  $J$  cells each of size  $\Delta x$ . Our basic variable  $u$  is conserved under the action of equation (1). It is, therefore, important that this is reflected in any numerical method we employ. A particular class of methods with this conservation property update the solution at the  $i$ th cell via

$$u_i^{n+1} = u_i^n + \frac{\Delta t}{\Delta x} (F(u_{i-1}^n, u_i^n) - F(u_i^n, u_{i+1}^n)). \quad (3)$$

In this equation  $n$  is the current time level,  $n+1$  the next time level and  $\Delta t$  the time step between the two;  $F(u_L, u_R)$  is the *numerical flux* through the *interface* between two neighbouring cells,  $u_L$  and  $u_R$  the states of the left and right cells respectively. The update is conservative no matter how we define the function  $F$ .

The time step  $\Delta t$  need not necessarily be constant throughout our numerical calculation. One approach is to let it be given by the formula

$$\Delta t = \frac{\Delta x C_{cfl}}{S_{\max}^n}, \quad (4)$$

wherein  $0 < C_{cfl} \leq 1$  is the *Courant-Friedrichs-Lewy (CFL) coefficient* and  $S_{\max}^n$  the maximum wave speed at time level  $n$ . In the case of equation (1) this is simply

$$S_{\max}^n = \max_{0 \leq i \leq J} \{|u_i^n|\}.$$

## 3 Numerical Methods

A simple numerical method is the Lax-Friedrichs scheme wherein the numerical flux is defined as

$$F^{\text{LF}}(u_L, u_R) = \frac{1}{2} (f(u_L) + f(u_R)) + \frac{1}{2} \frac{\Delta x}{\Delta t} (u_L - u_R). \quad (5)$$

**Question 2** Given the initial condition (2) and the boundary conditions specified in Section 1, write a program that marches equation (1) forward to a time  $t = \frac{1}{2}$  using the Lax-Friedrichs scheme. Derive the exact solution analytically and compare your numerical solution with it. What is the order of accuracy of the scheme?

Another numerical method is the Richtmyer scheme. The numerical flux in this case is calculated via

$$u^{\text{Ri}} = \frac{1}{2} (u_L + u_R) + \frac{1}{2} \frac{\Delta t}{\Delta x} (f(u_L) - f(u_R)), \quad F^{\text{Ri}}(u_L, u_R) = f(u^{\text{Ri}}). \quad (6)$$

**Question 3** Using the same initial and boundary conditions as in Question 2, write a program that marches equation (1) forward to a time  $t = \frac{1}{2}$  using the Richtmyer scheme. Compare your numerical solution with the exact solution commenting on the order of accuracy and any interesting features you observe. How do these results compare with those obtained previously? Your remarks should include an outline discussion of the property known as *monotonicity* and a statement of its relevance to both the Richtmyer and Lax-Friedrichs schemes. Reference to the textbook by Toro [4] may be helpful.

In order to eradicate the inaccuracies associated with each of the above schemes, the following numerical flux was proposed:

$$F = F^{\text{LO}} + \phi[F^{\text{HI}} - F^{\text{LO}}], \quad (7)$$

which contains both a high (HI) and low (LO) order flux and a *flux limiter*  $\phi$ . The limiter lies in the range  $0 \leq \phi \leq 1$  and acts to vary the overall flux,  $F$ , locally between low and high order.

There are many different limiter functions. We will consider here

$$\phi = \begin{cases} 0 & r \leq 0, \\ r & 0 \leq r \leq 1, \\ 1 & r \geq 1, \end{cases}$$

wherein  $r$  is defined locally as

$$r_i^n = \min \left\{ \frac{u_i^n - u_{i-1}^n}{u_{i+1}^n - u_i^n}, \frac{u_{i+2}^n - u_{i+1}^n}{u_{i+1}^n - u_i^n} \right\} \quad (8)$$

and is a measure of the local change in gradient of the solution.

**Question 4** Is the overall flux,  $F$ , predominantly high or low order accurate in the neighbourhood of:

- (i) sharp changes in gradient?
- (ii) slight changes in gradient?

Outline your reasoning given the above formulae and why, in light of earlier results, such variation of  $F$  is desirable.

A particular method is the Flux Limited Centred (FLIC) scheme. In this case the high and low order fluxes are defined as

$$\begin{aligned} F^{\text{HI}} &= F^{\text{Ri}}, \\ F^{\text{LO}} &= \frac{1}{2}[F^{\text{LF}} + F^{\text{Ri}}]. \end{aligned}$$

**Question 5** Using the same initial and boundary conditions as in Questions 2 and 3, write a program that marches equation (1) forward to a time  $t = \frac{1}{2}$  using the FLIC scheme. Compare your numerical results with the exact solution and with the results obtained earlier. To what extent has the scheme eliminated the undesirable features of the previous methods?

## References

- [1] Billingham, J. & King, A.C. *Wave Motion: Theory and application*. Cambridge University Press, 2000.
- [2] Lighthill, M.J. *Waves in Fluids*. Cambridge University Press, 1978.
- [3] Renardy, M. & Rogers, R.C. *An Introduction to Partial Differential Equations*. Springer-Verlag, 1993.
- [4] Toro, E.F. *Riemann Solvers and Numerical Methods for Fluid Dynamics*. Springer-Verlag, 1997.
- [5] Whitham, G.B. *Linear and Nonlinear Waves*. Wiley, 1999.

## 2 Waves

### 2.7 Soliton Solutions of the KdV Equation (7 units)

This project assumes only basic knowledge of wave equations and does not rely directly on any Part II lecture course. The Part II(D) course *Integrable Systems* may be useful but the project should be quite accessible to those who have not taken this course, using *Drazin & Johnson [1]* as a suitable reference.

#### 1 Theory

The Korteweg-de Vries (KdV) equation,

$$u_t + uu_x + \delta^2 u_{xxx} = 0, \quad (1)$$

arises in many branches of physics as a model for the evolution and interaction of nonlinear waves. It is well-known that the equation possesses single-soliton solutions of the form  $u(x, t) = f(x - ct)$ , where

$$f(x) = A \operatorname{sech}^2 \left( \frac{x - x_0}{\Delta} \right), \quad \Delta^2 = 12\delta^2/A, \quad c = A/3, \quad (2)$$

and where  $A$  and  $x_0$  are arbitrary constants representing the amplitude and initial location of the soliton respectively. It is supposed that  $u(x, t)$  obeys the cyclic boundary conditions,

$$u(x + 1, t) = u(x, t), \quad (3)$$

so that only the region  $0 \leq x \leq 1$  need be considered.

#### 2 Questions

**Question 1** Verify that the single soliton (2) is indeed a [non-periodic] solution of the KdV equation. For a periodic solution satisfying (3), prove that the mass,  $M$ , and the energy,  $E$ , of the motion, defined by

$$M \equiv \int_0^1 u(x, t) dx \quad \text{and} \quad E \equiv \int_0^1 \frac{1}{2} u^2(x, t) dx, \quad (4)$$

are independent of time.

**Question 2** A leap-frog scheme first employed by Zabusky & Kruskal for solving the KdV equation is given on page 184 of *Drazin & Johnson [1]*. Let  $u_m^n$  be the solution at  $x = hm$  and  $t = kn$  with  $n, m = 0, 1, \dots$ , then the KdV equation can be discretised as

$$u_m^{n+1} = u_m^{n-1} - \frac{k}{3h} (u_{m+1}^n + u_m^n + u_{m-1}^n) (u_{m+1}^n - u_{m-1}^n) - \frac{k\delta^2}{h^3} (u_{m+2}^n - 2u_{m+1}^n + 2u_{m-1}^n - u_{m-2}^n). \quad (5)$$

What is the order of this scheme? For  $\delta = 1$ , this scheme will be stable provided that

$$k \leq \frac{h^3}{4 + h^2 |u_{\max}|}, \quad (6)$$

where  $|u_{\max}|$  is the maximum modulus of  $u$ . By rescaling the KdV equation, derive the stability condition for  $\delta \neq 1$ .

Write a program to implement (5). Note that you will need an alternative method for making the first step; explain your choice carefully in your write-up. As a check on the accuracy of your program, use it to calculate  $u(x, 0.5)$  for the initial data

$$u(x, 0) = A \operatorname{sech}^2 \left( \frac{x - x_0}{\Delta} \right), \quad (7)$$

with  $\delta = 0.03$ ,  $A = 2$  and  $x_0 = 0.25$ .

Estimate the error in your results, and clearly explain the reasoning behind your choice of the values of  $k$  and  $h$ . Plot on the same axes graphs of both your numerical solution and the exact solution at  $t = 0.5$ . Comment on any differences. Does the propagation speed of the numerical solution agree exactly with that of the analytical solution?

**Question 3** Describe and comment on the evolution of the initial data corresponding to the sum of two solitons, one with  $A = 2$  and  $x_0 = 0.25$ , and a second with  $A = 1$  and  $x_0 = 0.75$ , again with  $\delta = 0.03$ . Illustrate your answer with plots of the numerical solution at several instants. In addition, use the trapezium rule to calculate the mass and energy associated with your numerical solution at each time step, and comment on their variation with time (you need not output these quantities at every time step).

**Question 4** Consider now

$$u(x, 0) = \sin(2\pi x). \quad (8)$$

In the case  $\delta = 0$  give a very brief qualitative description of the evolution of this initial data. Use your program to investigate the case  $\delta = 0.03$  numerically, plotting graphs of the solution at suitable instants. With particular reference to the relative magnitudes of the various terms in the KdV equation, explain why your numerical solution changes character at a certain time; estimate this time. Describe how the solution evolves after this time.

Try different values of  $\delta$ , both larger and smaller than 0.03, and comment on the results that you obtain.

## References

- [1] Drazin, P.G. & Johnson, R.S. (1989) *Solitons: An Introduction*, CUP.
- [2] Zabusky, N. & Kruskal, M (1965) Interaction of “solitons” in a collisionless plasma and the recurrence of initial states. *Physical Review Letters*, volume 15, number 6, pages 240-3.

## 2 Waves

### 2.10 Phase and Group Velocity

(8 units)

*Part II Waves is helpful but not essential. The main prerequisite is an elementary knowledge of the method of stationary phase, as taught in Part II Waves and Part II Asymptotic Methods. However, all the required material can be found in [1], or any of the books listed in the Asymptotic Methods schedule, or those by Billingham & King, Lighthill and Whitham in the Waves schedule.*

The Klein-Gordon equation,

$$u(x, t) : \quad u_{tt} - c_0^2 u_{xx} = -q^2 u \quad \text{with } c_0 > 0 \text{ and } q \geq 0 \text{ constants ,} \quad (1)$$

arises, for example, when looking for solutions of ‘the’ wave equation  $p_{tt} - c_0^2 \nabla^2 p = 0^*$  in the form  $p = u(x, t) \cos(m\pi y/a) \cos(n\pi z/b)$  to describe the propagation of, say, sound waves in a rectangular tube (‘waveguide’) occupying  $0 < y < a$ ,  $0 < z < b$ . It is a simple example of a wave equation which is both *hyperbolic* (there is an upper bound on the speed at which waves can propagate) and *dispersive* (different Fourier components propagate at different speeds).

#### 1 Initial-value problem

The goal of this part of the project is to solve (1) for  $x \in (-\infty, \infty)$  with initial conditions

$$u(x, 0) = f(x) , \quad u_t(x, 0) = 0 , \quad (2)$$

where  $f(x)$  is some specified (real) function with  $f(x) \rightarrow 0$  as  $|x| \rightarrow \infty$ , and which for simplicity we shall assume to be even in  $x$ , in which case the solution  $u(x, t)$  will be even in  $x$  for all  $t$ . Without loss of generality we set  $c_0 = 1$ . (Why?)

When  $q = 0$ , (1) reduces to ‘the’ one-dimensional wave equation familiar from IB Methods, and the solution subject to (2) is

$$u(x, t) = \frac{1}{2} [f(x - t) + f(x + t)] . \quad (3)$$

**Question 1** Show that for general  $q$ , (1)–(2) may be solved by a Fourier transform in  $x$  to give

$$u(x, t) = \frac{1}{4\pi} \int_{-\infty}^{\infty} \tilde{f}(k) e^{ikx - i\Omega(k)t} dk + \text{complex conjugate} \quad (4)$$

where

$$\tilde{f}(k) = \int_{-\infty}^{\infty} f(x) e^{-ikx} dx \quad (5)$$

and

$$\Omega(k) = \sqrt{q^2 + k^2} \quad (\text{the ‘dispersion relation’}). \quad (6)$$

Note that if  $f(x)$  is even in  $x$ ,  $\tilde{f}(k)$  is real (and even in  $k$ ).

Sketch graphs<sup>†</sup> of ‘phase velocity’  $\Omega(k)/k$ , and ‘group velocity’  $\Omega'(k)$  against  $k$  (on the same axes). Give a physical interpretation of phase velocity.

---

\*More properly, this should be called ‘the linear non-dispersive wave equation’, for there are many other equations which describe wave propagation, the Klein-Gordon equation being one such.

<sup>†</sup> Whereas almost all graphs, including labels, annotations, etc., need to be computer-generated, this is one of the relatively few cases where a scanned hand-drawing is acceptable for electronic submission.



An alternative representation of the solution, which you are *not* asked to verify<sup>‡</sup>, is

$$u(x, t) = \frac{1}{2} \left[ f(x - t) + f(x + t) - qt \int_{-\pi/2}^{\pi/2} J_1(qt \cos \theta) f(x + t \sin \theta) d\theta \right] \quad (7)$$

where  $J_1(x)$  is the Bessel function of the first kind [which could be computed by using the MATLAB function `besselj(1,x)`]. Neither analytic form is particularly suitable for calculating  $u(x, t)$  at a large number of points. Nevertheless, the Fourier representation is convenient in allowing a simple approximation for large  $t$  to be obtained by the *method of stationary phase*.

**Question 2** Giving only a brief outline of the theory, show by the method of stationary phase applied to the integral (4) that for  $q > 0$ , in the limit  $t \rightarrow \infty$  with  $V \equiv x/t$  fixed and  $|V| < 1$

$$u(x, t) \sim [2\pi\Omega''(k_0)t]^{-1/2} |\tilde{f}(k_0)| \cos \left[ k_0x - \Omega(k_0)t + \arg \tilde{f}(k_0) - \frac{1}{4}\pi \right] \quad (8)$$

where  $k_0$  is specified by

$$\Omega'(k_0) = V, \quad (9)$$

and hence (assuming that  $\tilde{f}(k)$  is real)

$$u(x, t) \sim \frac{q^{1/2}t}{(2\pi)^{1/2}(t^2 - x^2)^{3/4}} \tilde{f} \left( \frac{qx}{\sqrt{t^2 - x^2}} \right) \cos \left( q\sqrt{t^2 - x^2} + \frac{1}{4}\pi \right) \quad \text{for } |x| < t. \quad (10)$$

Explain how this provides a physical interpretation of group velocity.

**Question 3** Write a program to solve (1)–(2) numerically as follows: take a uniform grid with steplengths  $\Delta t$  in  $t$  and  $\Delta x$  in  $x$ , and use the centred-difference approximation

$$u_{tt}(x, t) = \frac{u(x, t + \Delta t) - 2u(x, t) + u(x, t - \Delta t)}{(\Delta t)^2} + O((\Delta t)^2), \quad (11)$$

and a similar one for  $u_{xx}$ , to generate the finite-difference scheme

$$\frac{u_i^{j+1} - 2u_i^j + u_i^{j-1}}{(\Delta t)^2} - \frac{u_{i+1}^j - 2u_i^j + u_{i-1}^j}{(\Delta x)^2} = -q^2 \left( \frac{u_{i+1}^j + u_{i-1}^j}{2} \right) \quad (12)$$

where  $u_i^j$  is an approximation to  $u(i\Delta x, j\Delta t)$ . (It is possible to use the more natural  $-q^2 u_i^j$  on the right-hand side, but then a smaller  $\Delta t$  may be required for numerical stability: see [2], p.588.) The solution at  $t$ -level  $j + 1$  can thus be found from those at  $j$  and  $j - 1$ . Clearly the first  $t$ -level needs special treatment: for the initial conditions (2) it is sufficient, to the order of accuracy of the scheme, to set  $u_i^{-1} = u_i^1$ . (Why?) Further, since we are restricting attention to solutions even in  $x$ , we need only solve in  $0 \leq x \leq L \equiv N\Delta x$  with the additional boundary condition  $u_x(0, t) = 0$ , which can be implemented in the finite-difference scheme by setting  $u_{-1}^j = u_1^j$ . Note that the  $u_0^j$  are determined as part of the solution, but the outer-boundary values  $u_N^j$  need to be specified.

---

<sup>‡</sup>The derivation is indicated in [3]

Run the program with

$$f(x) = \begin{cases} (1 - x^2)^2 & |x| \leq 1 \\ 0 & |x| \geq 1 \end{cases}, \quad (13)$$

for both  $q = 0$  and  $q = 1$ , at least up to  $t = 50$ . The  $x$ -range should be sufficiently large that the outer-boundary values  $u_N^j$  can safely be set equal to zero. Experiment with different values of  $\Delta x$  and  $\Delta t/\Delta x$ , and comment on the accuracy and stability of the numerical scheme; you may wish to compare with the exact solutions (3) and/or (7) at selected points.

Find  $\tilde{f}(k)$  and plot its graph.

Plot the finite-difference solutions against  $x$  at various representative values of  $t$ , using steplengths which are sufficiently small for graphical accuracy; for the case  $q = 1$ , superpose the stationary-phase approximation (10). Comment on the solutions, mentioning similarities and differences between those for  $q = 0$  and  $q = 1$ . What would you expect to happen if  $q$  were non-zero but very small?

## 2 Signalling Problem

**Question 4** Modify your program to solve (1) for  $x > 0$ ,  $t > 0$  with the initial conditions

$$u(x, 0) = u_t(x, 0) = 0$$

and the boundary condition

$$u(0, t) = \sin(\omega_0 t).$$

Physically, this describes a system which is undisturbed ( $u \equiv 0$ ) for  $t < 0$ , but excited for  $t > 0$  by a time-harmonic forcing applied at  $x = 0$ . As before, there is no loss of generality in taking  $c_0 = 1$ . (Why?)

For  $q = 0$ , find the solution analytically, and use it as a check on the program.

Run the program with  $q = 1$  and  $\omega_0 = 0.9, 1.1$  and  $1.5$ , at least as far as  $t = 150$ , with suitable steplengths and domain size (give brief justification for your choice). For each case, present *selected* results to illustrate the key features. Comment on the solutions, particularly in the light of phase and group velocities (see Question 1).

## References

- [1] Carrier, G.F., Crook, M. & Pearson, C.E., *Functions of a Complex Variable*.
- [2] Dodd, R.K., Eilbeck, J.C., Gibbon, J.D., & Morris, H.C., *Solitons and Nonlinear Wave Equations*.
- [3] Ockendon, J.R., Howison, S.R., Lacey, A.A., & Movchan, A.B., *Applied Partial Differential Equations*.

# 3 Fluid and Solid Dynamics

## 3.1 Boundary-Layer Flow (6 units)

*This project is based on a section of the Part II course Fluid Dynamics but the relevant material can be found in Chapter 8 of [1] or Chapter V of [2]; a brief outline is given below. Further relevant material can be found in book by Sobey [3] and the paper by Stewartson [4].*

### 1 Background Theory

Consider flow of an incompressible viscous fluid of constant density  $\rho$  and kinematic viscosity  $\nu$ , whose velocity  $\mathbf{u}(\mathbf{x}, t)$  and hydrodynamic pressure  $p(\mathbf{x}, t)$  satisfy the incompressibility condition

$$\nabla \cdot \mathbf{u} = 0, \quad (1)$$

and the Navier-Stokes momentum equation

$$\rho (\mathbf{u}_t + (\mathbf{u} \cdot \nabla) \mathbf{u}) = -\nabla p + \rho \nu \nabla^2 \mathbf{u}. \quad (2)$$

It follows that the vorticity  $\boldsymbol{\omega} \equiv \nabla \times \mathbf{u}$  satisfies

$$\boldsymbol{\omega}_t + (\mathbf{u} \cdot \nabla) \boldsymbol{\omega} - (\boldsymbol{\omega} \cdot \nabla) \mathbf{u} = \nu \nabla^2 \boldsymbol{\omega}. \quad (3)$$

If the flow is two-dimensional, it can be expressed in terms of a streamfunction  $\psi(x, y, t)$ , with

$$\mathbf{u} \equiv (u, v, 0) = \left( \frac{\partial \psi}{\partial y}, -\frac{\partial \psi}{\partial x}, 0 \right), \quad \boldsymbol{\omega} = (0, 0, -\nabla^2 \psi), \quad (4)$$

in terms of which the vorticity equation (3) becomes

$$\frac{\partial}{\partial t} (\nabla^2 \psi) - \frac{\partial (\psi, \nabla^2 \psi)}{\partial (x, y)} = \nu \nabla^2 (\nabla^2 \psi). \quad (5)$$

Suppose there is a stationary rigid boundary on  $y = 0$ ,  $x > 0$  on which must be satisfied the conditions of no slip,  $u = 0$ , and no penetration,  $v = 0$ . If the viscosity is ‘small’, then away from the boundary the flow *may*, to a good approximation, be irrotational: in this case  $\mathbf{u} \approx \nabla \phi$  for a velocity potential  $\phi$  satisfying  $\nabla^2 \phi = 0$  (incompressibility) and  $\partial \phi / \partial y = 0$  on  $y = 0$ ,  $x > 0$  (no penetration). These conditions, together with corresponding ones on other boundaries and/or at infinity, are sufficient to determine  $\nabla \phi$  uniquely. As a result, it is not possible to specify the tangential (‘slip’) velocity component on the boundary, i.e.

$$\frac{\partial \phi}{\partial x}(x, 0, t) \equiv U_e(x, t) \quad (x > 0).$$

Instead,  $U_e(x, t)$  is determined as part of the irrotational potential-flow solution, and is necessarily non-zero. There must therefore be a ‘boundary layer’ near  $y = 0$  where the potential-flow approximation is not valid and satisfaction of the no-slip condition implies viscous diffusion of vorticity away from the boundary.

If the boundary layer is ‘thin’, in the sense that within it  $\partial / \partial x \ll \partial / \partial y$  (i.e. variations with respect to  $y$  are much more rapid than variations with respect to  $x$ ), then the approximations

$$\nabla^2 \psi \approx \psi_{yy}, \quad \nabla^2 (\nabla^2 \psi) \approx \psi_{yyyy},$$

may be made in equation (5), which can then be integrated once with respect to  $y$  to give

$$\psi_{yt} + \psi_y \psi_{xy} - \psi_x \psi_{yy} = G(x, t) + \nu \psi_{yyy} , \quad (6)$$

This is in fact the  $x$ -component of the Navier-Stokes momentum equation (2) with the  $\rho \nu u_{xx}$  term neglected (since it is small compared to  $\rho \nu u_{yy}$ ). The forcing term, or more precisely  $-\rho G$ , may be identified with the pressure gradient  $\partial p / \partial x$  (which in this approximation is independent of  $y$ , i.e. uniform across the ‘thin’ boundary layer). Equation (6) is to be solved in conjunction with the conditions of no slip and no penetration at the boundary, i.e.

$$\psi_y = 0 , \quad -\psi_x = 0 \Rightarrow \psi = \psi_0(t) , \quad \text{on } y = 0 , \quad (7)$$

where  $\psi_0(t) = 0$  without loss of generality. In addition the solution must ‘match’ to the outer potential flow, i.e.

$$u \equiv \psi_y \rightarrow U_e(x, t) \quad \text{as } y \rightarrow \infty . \quad (8)$$

It follows from (6) and (8) that

$$G(x, t) = \frac{\partial U_e}{\partial t} + U_e \frac{\partial U_e}{\partial x} . \quad (9)$$

In the special case when the flow is steady,  $\partial / \partial t = 0$ , and when

$$U_e(x) = Ax^m \quad \text{for } x > 0 , \quad (10)$$

with  $A$  and  $m$  constants, the problem (6)–(9) admits a ‘similarity’ solution with

$$\psi(x, y) = |U_e(x)| \delta(x) f(\eta) , \quad \eta = \frac{y}{\delta(x)} \quad \text{and} \quad \delta(x) = \left( \frac{\nu x}{|U_e(x)|} \right)^{\frac{1}{2}} . \quad (11)$$

Here,  $\delta(x)$  is a measure of the local boundary-layer thickness, and the function  $f$  satisfies the *Falkner–Skan* equation

$$m (f')^2 - \frac{1}{2}(m+1) f f'' = m + f''' , \quad (12)$$

with the boundary conditions

$$f' = f = 0 \quad \text{on } \eta = 0 , \quad f' \rightarrow \text{sgn } A \quad \text{as } \eta \rightarrow \infty . \quad (13)$$

In fact, if there is no source of vorticity other than the boundary  $y = 0$ , then  $f'$  should converge to  $\text{sgn } A$  *exponentially fast* as  $\eta \rightarrow \infty$ , i.e.

$$\eta^N (f' - \text{sgn } A) \rightarrow 0 \quad \text{as } \eta \rightarrow \infty \quad \text{for any } N . \quad (14)$$

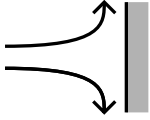
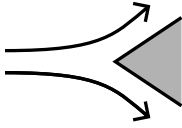

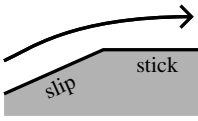
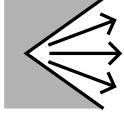
The tangential velocity component is obtained from (4) and (11) as

$$u = \frac{\partial \psi}{\partial y} = |U_e(x)| f'(\eta) . \quad (15)$$

The tangential stress (force per unit area in the  $x$ -direction) exerted by the fluid on the boundary  $y = 0$  is

$$\tau_0 \equiv \rho \nu \frac{\partial u}{\partial y} \Big|_{y=0} = \rho \left( \frac{\nu |U_e(x)|^3}{x} \right)^{\frac{1}{2}} f''(0) . \quad (16)$$

Different values of  $m$  arise for different external flows. Particular cases are (taking  $A > 0$  unless stated otherwise):

- $m = 1$  : Flow towards a stagnation point on a plane wall. 
- $0 < m < 1$  : Flow past a wedge of semi-angle  $\theta = \frac{\pi m}{m+1}$ . 
- $m = 0$  : Flow past a flat plate (the Blasius boundary layer). 
- $-\frac{1}{2} < m < 0$  : Flow around the outside of a corner, turning through an angle  $\theta = -\frac{\pi m}{m+1}$  with slip (but no penetration) upstream of the corner, and no slip (and no penetration) downstream. This is an artificial problem, but it *might* be that the solution could arise as the downstream limit of a realistic flow. 
- $m = -1$  : Flow due to a line source (for  $A > 0$ ) or line sink ( $A < 0$ ) at the intersection of two plane rigid boundaries (at arbitrary angle). 

The aim of this project is to solve the two-point-boundary-value problem (12)–(14) by ‘shooting’, finding by trial-and-error the values of  $f''(0)$  which give a solution with the required behaviour as  $\eta \rightarrow \infty$ . Except in the last question, attention is to be restricted to the case  $A > 0$ .

## 2 Analysis

**Question 1** Examine analytically the possibility that a solution of the Falkner-Skan equation (12) has one of the following terminal behaviours as  $\eta \rightarrow \infty$ :

(i) algebraic convergence,

$$f' = 1 + B\eta^{-k} + \dots \quad \text{as } \eta \rightarrow \infty;$$

(ii) exponential convergence,

$$f = \eta - \eta_0 + e^{-\sigma(\xi)} + \dots \quad \text{as } \eta \rightarrow \infty,$$

where  $\xi = \eta - \eta_0$  and

$$\sigma'(\xi) = k\xi + k' + k''\xi^{-1} + O(\xi^{-2});$$

(iii) algebraic divergence,

$$f = B\eta^{1+k} + \dots \quad \text{as } \eta \rightarrow \infty;$$

(iv) exponential divergence,

$$f = Be^{k\eta} + \dots \quad \text{as } \eta \rightarrow \infty;$$

(v) a finite-distance singularity, at  $\eta = \eta_0$  say, with

$$f = B(\eta_0 - \eta)^{-1} + \dots \quad \text{as } \eta \rightarrow \eta_0 .$$

Here  $B, \eta_0, k, k', k''$  are constants with  $B \neq 0$  and  $k > 0$ . For each case deduce by means of an asymptotic series solution, or otherwise,

- (a) which (if any) of these constants can be determined without use of conditions at  $\eta = 0$ ;
- (b) what restrictions (if any) must be placed on the value of  $m$  for this type of behaviour to be possible.

### 3 Computation

**Programming Task:** Write a program to integrate the Falkner–Skan equation (12) subject to the one-point boundary conditions

$$f(0) = f'(0) = 0, \quad f''(0) = S, \quad (17)$$

where  $S$  is a known constant. You may use a black-box numerical integrator for this problem such as the MATLAB routine `ode45`, but you should not use any *two-point-boundary-value* solver such as the MATLAB routine `bvp4c` (except possibly as a check).

**Question 2** Integrate the Falkner-Skan equation (12) with boundary conditions (17) for  $m = 0$  and  $S = 1$ . You should find that  $f'$  converges to a constant as  $\eta \rightarrow \infty$ ; comment on the nature of the convergence, and determine the constant to at least four significant figures, presenting evidence that this accuracy has been achieved.

Explain how it is possible to deduce, without further computation, a solution of the boundary-value problem (12)–(13) for  $m = 0$ , and state  $f''(0)$ .

*Hint:* consider  $af(b\eta)$  for suitable constants  $a$  and  $b$ .

**Question 3** Now apply the shooting method for  $0 \leq m \leq 1$ . You should find that for each  $m$  in this range, there is a unique value of  $S$ , call it  $S_m$ , for which  $f' \rightarrow 1$  as  $\eta \rightarrow \infty$ . For  $m = \frac{2}{5}$  and  $m = 1$ , plot graphs of  $f'$  against  $\eta$  for various values of  $S$ , both less than and greater than  $S_m$ , and comment on their terminal behaviours with reference to the terminal behaviours listed in question 1.

Write a program to determine  $S_m$ . Note:

- you may wish to use a black-box root-finder such as the MATLAB routine `fzero`;
- it may also be helpful, for the next question, to have the option of finding the inverse, i.e. determining  $m$  for given  $S$ .

Tabulate and plot  $S_m$  against  $m$  for  $0 \leq m \leq 1$  correct to at least four significant figures, explaining why you are satisfied with the accuracy.

Comment on the *physical* interpretation of your solutions, e.g. the effect of varying  $m$ .

**Question 4** Investigate solutions of the Falkner-Skan equation (12) subject to (17) for various  $m$  in the range  $-1 < m < 0$  and various  $S$  (both positive and negative), and display some representative results, illustrating the different kinds of terminal behaviour

(as enumerated in question 1) which occur. In the range  $m_c < m < 0$  (where  $m_c$  is to be determined) you should find two branches of exponentially converging solutions, one of which is the continuation of that already found for  $m \geq 0$ . Tabulate and plot  $S_m$  against  $m$  for both branches, and plot  $f'$  for both solutions against  $\eta$  for at least one value of  $m$ . Discuss the *physical* interpretation of each solution, and the form of the second solution as  $m \uparrow 0$ .

In the interval  $-1 < m < m_c$ , there are other branches of exponentially converging solutions: plot at least two of these branches in the  $m$ - $S$  plane, and present graphs of  $f'$  against  $\eta$  for a few of the solutions. Comment on their *physical* significance.

**Question 5** Investigate (numerically)  $m = -1$ , for both signs of  $A$ . What are your conclusions?

## References

- [1] Acheson, D. J., *Elementary Fluid Dynamics*. O.U.P.
- [2] Rosenhead, L. (ed.), *Laminar Boundary Layers*. Dover. (In particular Chapter V, sections 1, 12-17, 21.)
- [3] Sobey, I. J., *Introduction to Interactive Boundary Layer Theory*. O.U.P.
- [4] Stewartson, K. (1954) Further solutions of the Falkner-Skan equation. *Proceedings of the Cambridge Philosophical Society*, volume 50, issue 03, pages 454-465. See <http://dx.doi.org/10.1017/S030500410002956X>

## 3 Fluid and Solid Mechanics

### 3.8 Wind-Forced Ocean Currents (10 units)

*This project may well be attempted by someone who has attended the two Fluid Dynamics courses in Part IB and Part II.*

#### 1 Theory

Western boundary currents develop on the western sides of the ocean basins in response to wind forcing. Poleward flowing western boundary currents like the Gulf Stream and Kuroshio carry vast amounts of heat from the tropics to the mid-latitudes. Here, you will investigate wind-forced ocean circulation and the formation of western boundary currents in an idealized rectangular ocean basin.

A simple depth-independent model of the wind-forced ocean circulation is described by the governing equation for the streamfunction  $\psi(x, y, t)$ ,

$$\zeta_t + J(\psi, \zeta) + v = -\epsilon\zeta + R\tau \quad (1)$$

in  $0 \leq x \leq 1$ ,  $0 \leq y \leq 1$  with  $\psi = 0$  on the boundaries.  $x$  and  $y$  are Cartesian coordinates representing eastward and northward directions respectively. The vorticity  $\zeta$  is related to  $\psi$  through the Poisson equation

$$\nabla^2\psi = \zeta, \quad (2)$$

and the  $x$  and  $y$  components of the velocity, respectively  $u$  and  $v$ , may be written in terms of  $\psi$  as

$$u = -\frac{\partial\psi}{\partial y}, \quad v = \frac{\partial\psi}{\partial x} \quad (3a, b),$$

$J(\psi, \zeta)$  is the Jacobian with respect to  $x$  and  $y$  and is an alternative way of writing the advective derivative term  $\mathbf{u} \cdot \nabla\zeta$ . The  $-\epsilon\zeta$  term on the right-hand side of (1) is a simple representation of the effect of bottom friction on the flow. The constant  $\epsilon$  is a nondimensional frictional damping rate. The term  $R\tau(x, y)$ , representing the wind forcing, is equal to the curl of the wind stress. It is convenient to take  $\tau$  to be a prescribed function of  $x$  and  $y$  and, when investigating the behaviour of the model, to vary the strength of the forcing by varying the constant  $R$ .

The steady state form of (1) may be written in the form

$$\mathbf{u} \cdot \nabla(\zeta + y) = R\tau - \epsilon\zeta, \quad (4)$$

implying that in the absence of wind-forcing and bottom friction the quantity  $\zeta_a = \zeta + y$  would be conserved following the fluid motion.  $\zeta_a$  is known as the ‘absolute vorticity’ and is the vertical (i.e. perpendicular to the Earth’s surface) component of the vorticity measured with respect to an inertial frame (i.e. including the Earth’s rotation as well as the motion of fluid relative to the Earth). The  $y$  contribution to the absolute vorticity is a simple representation of the variation of the vertical component of the rotation vector with latitude.

**Question 1** Use incompressibility of  $\mathbf{u}$  to rewrite the left hand side of (4). By integrating (4) over a region enclosed by a streamline and, using the divergence theorem on the left-hand side, deduce that if  $\tau$  is one-signed then no steady state is possible if  $\epsilon = 0$ , i.e. friction is essential in the steady-state balance.



In this project you will be concerned with steady-state solutions to (1), and their variation as  $\epsilon$  and  $R$  are changed. However, a convenient way to find the steady-state solution is to integrate (1) in time, say from initial conditions in which  $\psi = 0$  for all  $x$  and  $y$ , until the steady state is achieved.

## 2 Numerical solution of (1)

Define a rectangular grid covering the domain, with points

$$(x_i, y_j) = \left( \frac{i}{N_x}, \frac{j}{N_y} \right) \quad 0 \leq i \leq N_x, 0 \leq j \leq N_y.$$

The grid spacings are  $\delta_x = 1/N_x$  and  $\delta_y = 1/N_y$  in the  $x$  and  $y$  directions respectively. The variables  $\zeta$  and  $\psi$  are defined at each point on the grid and it is helpful to use the notation,  $\psi_{i,j}^t = \psi(x_i, y_j, t)$ ,  $\zeta_{i,j}^t = \zeta(x_i, y_j, t)$ . (The superscripts denote the time at which a particular quantity is to be evaluated.)

In order to integrate (1) in time it is sensible to use the  $\zeta_{i,j}^t$  as the working independent variable and derive all the other quantities by solving (2) and then using finite-difference approximations for spatial derivatives in (1). You are recommended to use the expressions

$$v_{i,j}^t = \frac{\psi_{i+1,j}^t - \psi_{i-1,j}^t}{2\delta_x} \quad (5a)$$

for  $v$  and

$$J_{i,j}^t = [(\psi_{i+1,j+1}^t - \psi_{i-1,j+1}^t)\zeta_{i,j+1}^t - (\psi_{i+1,j-1}^t - \psi_{i-1,j-1}^t)\zeta_{i,j-1}^t - (\psi_{i+1,j+1}^t - \psi_{i+1,j-1}^t)\zeta_{i+1,j}^t + (\psi_{i-1,j+1}^t - \psi_{i-1,j-1}^t)\zeta_{i-1,j}^t]/4\delta_x\delta_y, \quad (5b)$$

for the Jacobian, both evaluated at the point  $(x_i, y_j)$  and time  $t$ .

To begin, it is recommended that to integrate (1) in time using the explicit Euler scheme in the form

$$\zeta_{i,j}^{t+\Delta t} - \zeta_{i,j}^t + J_{i,j}^t\Delta t + v_{i,j}^t\Delta t = -\frac{1}{2}\epsilon \left( \zeta_{i,j}^{t+\Delta t} + \zeta_{i,j}^t \right) \Delta t + R\tau_{i,j}\Delta t. \quad (6)$$

for  $1 \leq i \leq N_x - 1$ ,  $1 \leq j \leq N_y - 1$ . Note that the boundary condition on  $\psi$  means that evaluating  $J_{i,j}$  via (5b) at points immediately adjacent to the boundary does not require knowledge of  $\zeta$  on the boundary itself. There is no need to impose or determine  $\zeta$  on the boundary at any stage.

**Question 2** Write a program to integrate the above. Take  $\tau = -\sin \pi x \sin \pi y$ . You may use a library routine for the solution of Poisson's equation (see Appendix below). Try using a grid size  $N_x = N_y = 32$ . Note that numerical accuracy of the time integration is not particularly important here because it is only the final steady state that is of interest. You should experiment to find the largest possible time step  $\Delta t$  for which the integration remains stable and approaches a steady state.

Concentrate first on the case where  $R$  is very small. For  $\epsilon = 0.2$  and  $\epsilon = 0.05$  produce a plot to verify that your solution approaches a steady state. Plot contour maps of the streamfunction and vorticity fields for the steady state solution. Describe your results in qualitative terms.

In this regime you may assume that  $\zeta$  and  $\psi$  scale with  $R$  and thus you might find it helpful to redefine  $\zeta$  and  $\psi$  as  $\hat{\zeta} = \zeta/R$  and  $\hat{\psi} = \psi/R$ . Then in the limit  $R \rightarrow 0$  the

nonlinear term involving the Jacobian may be neglected, and the steady state form of (1) can be approximated as

$$\hat{v} = -\epsilon\hat{\zeta} + \tau. \quad (7)$$

You should see that the solution for  $\epsilon = 0.05$  is highly asymmetric in the  $x$  direction with a strong narrow flow close to the  $x = 0$  boundary and a broad weaker flow in the interior. Which term in the equation (1) leads to this asymmetry? For the case  $\epsilon = 0.05$ , indicate which terms in the equation play a dominant role in the balance in different parts of the flow. Provide an argument for why a strong current near the  $x = 1$  boundary cannot exist. In this linear (i.e. small  $R$ ) case, estimate the maximum value for  $\psi$  in the small- $\epsilon$  limit. (Consider where a boundary layer may lie, and which boundary conditions you can discard.) Repeat your integrations at higher resolution with  $N_x = N_y = 64$ . What are your conclusions? What resolution do you think would be needed to adequately resolve the boundary current near  $x = 0$ ?

**Question 3** Now, for  $\epsilon = 0.05$  investigate the steady-state behaviour as  $R$  increases through the range  $5 \times 10^{-4}$  to  $10^{-1}$ . Continue to use  $N_x = N_y = 32$ . You will find that the timestep  $\Delta t$  must be reduced as  $R$  increases, firstly to suppress numerical instability and secondly to allow a steady state to be achieved. For  $R = 0.1$  you will need to run your code for a while in order for a steady state to be achieved. You might find it helpful to replace (6) with a more accurate time-stepping scheme such as a 3rd order accurate Runge-Kutta or Adams-Bashforth scheme. This should allow you to take larger time steps, but it is not necessary and full marks can be obtained using (6).

Describe how the pattern of flow changes as  $R$  is increased. You may find it useful to look at contour plots of  $\psi$  and  $\zeta$  and also of the quantity  $y + \zeta$ . Include sufficient plots in your report to illustrate your main points. By considering plots of  $\psi$  and  $\zeta$  and  $y + \zeta$ , for large  $R$  identify the terms involved in the dominant balance of equation (1). Plot a graph of the maximum value of the streamfunction in the domain,  $\psi_{\max}$  against  $R$ , and try to explain its form. How would you expect  $\psi_{\max}$  to depend on the frictional damping rate  $\epsilon$  in the large- $R$  and small- $R$  limits?

### 3 Reference

This topic is discussed in some detail in Chapter 19 of the book by Vallis (Vallis, G.K., 2017: *Atmospheric and Oceanic Fluid Dynamics*. Cambridge University Press; reference copies and an online version are available from the Betty and Gordon Moore Library) but it is certainly not necessary to understand all of this Chapter in detail in order to complete this project.

### Appendix

A solver of the Poisson equation in (2) for  $\psi$  given  $\zeta_{i,j}^t$ , is provided on the CATAM website, located among the data files. After copying these matlab files into your working directory, you should be able to call the function `poisson` which is described below.

If you are using python, you can call the matlab script using `oct2py`. To do this, first install `oct2py` (e.g. `pip install oct2py`). Then, in your python script use the following commands: `from oct2py import Oct2Py, oc = Oct2Py()`, and call the function `poisson.m` using, e.g. `psi=oc.poisson(x,5,zeta)`.

[psi]=poisson(x,N,f)

The solver assumes a square grid and that  $\psi$  is 0 on all 4 boundaries.

- x: vector with grid locations in the x direction (and equivalently in the y direction). The first and last gridpoints in x should correspond to the boundary locations.
- N: Type of integration scheme. For 5 point N=5, 9 point N=9, modified 9 point N=10. For more information see reference below.
- f: Function to be integrated, in this case  $\zeta$ . Note that f should be a matrix of size (length(x),length(x)).
- psi: The solution to the Poisson equation on the *interior* gridpoints (excluding the values on the boundary which are set to 0). psi will be a matrix of size (length(x)-2,length(x)-2).

For more information on Poisson.m go to <https://cs.nyu.edu/~harper/poisson.htm>. Note that while the mathematics involved in the version described at the link are the same, the inputs of the Poisson solver provided have been modified slightly.

## 4 Dynamics

### 4.3 The Rotating Top (6 units)

This project assumes knowledge of the relevant material from the Classical Dynamics course, which may also be found in the reference listed.

This project concerns the familiar problem of the axisymmetric top, rotating about a fixed point on its axis of symmetry. By choosing units such that  $mgh = A$ , the Lagrangian (scaled by  $A$ ) is

$$L = \frac{1}{2}[\dot{\theta}^2 + \dot{\phi}^2 \sin^2 \theta] + \frac{1}{2}C[\dot{\psi} + \dot{\phi} \cos \theta]^2 - \cos \theta.$$

Here  $\theta$ ,  $\phi$ ,  $\psi$  are the usual 3 Eulerian angles.  $C$  is the ratio of the principal moments of inertia (in usual, dimensional, notation, equal to  $C/A$ ). The first Lagrangian equation of motion is then

$$0 = \frac{d}{dt} \left( \frac{\partial L}{\partial \dot{\theta}} \right) - \frac{\partial L}{\partial \theta} = \ddot{\theta} - \dot{\phi}^2 \sin \theta \cos \theta + C\dot{\phi} \sin \theta [\dot{\psi} + \dot{\phi} \cos \theta] - \sin \theta.$$

The other two can immediately be integrated once to give

$$\frac{\partial L}{\partial \dot{\psi}} = \text{constant} \quad \Rightarrow \quad C[\dot{\psi} + \dot{\phi} \cos \theta] = \alpha, \quad (1)$$

$$\frac{\partial L}{\partial \dot{\phi}} = \text{constant} \quad \Rightarrow \quad \alpha \cos \theta + \dot{\phi} \sin^2 \theta = \beta. \quad (2)$$

The energy integral is

$$E = \frac{1}{2}[\dot{\theta}^2 + \dot{\phi}^2 \sin^2 \theta] + \frac{1}{2}C[\dot{\psi} + \dot{\phi} \cos \theta]^2 + \cos \theta = \text{constant}. \quad (3)$$

Further details of the theory can be found in §5.7 of [1].

The equations can be rewritten in a form suitable for numerical solution as

$$\dot{\phi} = \frac{\beta - \alpha \cos \theta}{\sin^2 \theta}, \quad (4)$$

$$\ddot{\theta} = [\dot{\phi}(\dot{\phi} \cos \theta - \alpha) + 1] \sin \theta. \quad (5)$$

**Programming Task:** Write a program to investigate the motion numerically and to display the motion graphically. You should use MATLAB's 64-bit (8-byte) double-precision floating-point values or the equivalent in other programming languages. You may use any numerical method which you consider suitable, or library routines. For each of the questions below, you should consider the most appropriate and informative form of graphical output; possibilities include (but are not limited to) either 3D plots or 2D plots such as  $\theta$  vs.  $\phi$ ,  $\theta$  vs.  $t$  or  $\phi$  vs.  $t$ .

Comment on the numerical method chosen. Check the accuracy of your program by printing out  $E$  during each run, and comment on your results.

What problem can arise when  $\sin \theta$  becomes small? Your program will need to include a method for dealing with this difficulty, and you should check that it produces accurate results.

In later questions you may find that  $\theta$  sometimes leaves its usual domain ( $0 \leq \theta \leq \pi$ ). Should this occur, explain the physical meaning of your results and how your computed values for  $\theta$  are related to the proper values.

**Question 1** Explain the general theory underlying the motion of a rotating top, starting from equations (1)–(3). Demonstrate that your program can simulate the three main types of motion below and explain what initial conditions are required.

1. Normal precession ( $\dot{\phi}$  does not change sign)
2. Retrograde motion ( $\dot{\phi}$  changes sign)
3. Motion with cusps (the border between 1 and 2)

You should obtain one hard copy of each type of motion.

**Question 2** Choose values for  $\alpha$ ,  $\theta$  such that there are 2 values of  $\dot{\phi}$  that give a solution with constant  $\theta$ , and show that your program can replicate the motion. Explain how these results fit the general theory.

**Question 3** Investigate the stability of a sleeping top (i.e., one that spins with  $\theta = 0$ ) by giving the motion a small disturbance; specifically, use initial conditions  $\theta = 0$  and  $\dot{\theta}$  non-zero but small. Explain clearly the possible types of subsequent motion and give an appropriate criterion for stability. Obtain a rough estimate for the critical value of  $\alpha$ , which should be independent of  $C$ , and show examples of the motion just above and below the critical value. Is your estimate consistent with the theoretical predictions?

**Question 4** Take  $\alpha$ ,  $\theta$  very small (both 0.01 say) with initial conditions  $\dot{\theta} = \dot{\phi} = 0$ . What happens? Give a physical interpretation. Now change  $\alpha$  to zero, and explain your results.

## References

- [1] H. Goldstein, *Classical Mechanics*, 2nd edition. Addison-Wesley, 1980.

# 5 Quantum Mechanics

## 5.1 Band Structure (8 units)

*This project relies on a knowledge of material covered in the Part II course Applications of Quantum Mechanics.*

### 1 Introduction

In suitable units, the Schrodinger equation for a particle moving in a potential  $V$  is

$$\frac{d^2\psi}{dx^2} + (E - V(x))\psi = 0, \quad (1)$$

where  $\psi$  is the wavefunction and  $E$  is the energy. In this project, we suppose that  $V$  is periodic, in which case the Schrodinger equation may be considered to be a model for an electron in a crystal lattice. You will seek solutions to the Schrodinger equation such that  $\psi$  remains finite as  $|x| \rightarrow \infty$ : these will be called “allowed solutions”. You will verify that the energy values  $E$  for these solutions form a band structure.

If  $V(x)$  is periodic with period  $l$ , the solutions to (1) where  $\psi$  remains finite as  $|x| \rightarrow \infty$  can always be written as linear combinations of Bloch functions. These are functions such that

$$\psi(x) = e^{ikx}v(x), \quad v(x+l) = v(x), \quad (2)$$

where  $k$  is a real number. Hence, “allowed solutions” to (1) are those which can be expressed as linear combinations of Bloch functions. One sees that  $v(x)$  is a periodic function with the same period as  $V$ , but  $\psi(x)$  does not necessarily have this property.

### 2 Numerical Work

Consider a specific choice for  $V(x)$ , which is an even function [ $V(x) = V(-x)$ ], formed from an infinite series of nearly parabolic sections, each of width  $2a$ . For positive  $x$ ,

$$V(x) = 1 - \cosh(x - (2r + 1)a) \quad (3)$$

where  $r = \text{floor}(x/2a)$ , i.e. the greatest integer less than or equal to  $(x/2a)$ .

**Programming Task:** Given some  $x_{\max}$ , write a program which solves (1) for  $0 < x < x_{\max}$ . The solution depends on the value of  $E$ , and on the boundary values  $\psi(0)$  and  $\psi'(0)$ .

Your program should

- (i) input the values for  $E$  and  $x_{\max}$ , and suitable boundary conditions for  $\psi$ ;
- (ii) solve the Schrodinger equation for  $x$  between 0 and  $x_{\max}$ .

Your program should be able to plot the solution  $\psi$  as a graph. When running the program, you will need to select a suitable parameters for your numerical integration.

**Question 1** Consider the case

$$a = 2, \quad \psi(0) = 1, \quad \psi'(0) = 0, \quad x_{\max} = 150$$

Run your program for some energies in the range  $-1.5 < E < 2.0$  and report the results for  $\psi$ . You should choose the values of  $E$  to illustrate the different kinds of solution that you find. Your report should discuss the accuracy of the numerical methods which you use.

Note: In this question and throughout this project, you should provide graphs that illustrate clearly the behaviour that you observe. Note that very large numbers of graphs are *unlikely to be effective* in communicating this information.

**Question 2** Remember, the physically-relevant energies  $E$  are those for which  $\psi$  remains finite as  $|x| \rightarrow \infty$ . These “allowed” values of the energy are grouped into bands.

For the parameters of question 1, use trial and error to find five band boundaries in the range  $-1.5 < E < 2.0$ . Evaluate the energies of the band boundaries to two decimal places. In order to determine which energy values are allowed, you may want to consider the effect of increasing  $x_{\max}$ , it should be adequate to consider  $x_{\max} \leq 500$  but you may consider higher values if you wish.

Make graphs showing representative solutions for  $\psi$ . You should show examples for energies that are near to all band boundaries, and examples taken from near the middle of two different bands. Comment on your results. Are the band boundaries affected by choosing different initial conditions for  $\psi$ , for example  $\psi(0) = 0$ ,  $\psi'(0) = 1$ ?

**Question 3** The allowed solutions can be expressed as linear combinations of Bloch functions, as defined in (2). Given a solution for  $\psi$ , show how you can extract  $k$  from your numerical results. How does the energy  $E$  depend on  $k$ ? (It may be useful to make a sketch.)

Make a graph in which you compare one of your numerical solutions with a suitably chosen linear combination of Bloch functions.

**Question 4** Compare your numerical results with those expected from the ‘nearly free’ and ‘tightly bound’ models of electrons in solids. Which is the most appropriate model for the different energy bands that you have found?

# 6 Electromagnetism

## 6.1 Diffraction pattern due to a current strip (7 units)

*Knowledge of material covered in the Part IB course Electromagnetism is useful as background.*

This project investigates the magnetic field generated by an oscillating current. The field is given in terms of an integral whose behaviour is analysed numerically.

### 1 Theory

Consider an infinite two-dimensional strip of conductive material in the plane  $y = 0$  that covers the area defined by  $-d < x < d$  and  $-\infty < z < \infty$ . A time-dependent current flows in the  $z$ -direction, and it emits electromagnetic (radio) waves with wavelength  $\lambda$ . We assume that  $d = n\lambda/2$  where  $n$  is a positive integer. The time-dependent current is independent of  $x, z$ , and is given by

$$j_z(t) = j_0 e^{i\omega t}.$$

where  $j_0$  is a parameter and  $\omega = 2\pi c/\lambda$ . In the following, all length scales are normalised so that  $\lambda = 1$ , hence for example  $d = n/2$ .

Now consider the component of the magnetic field in the  $x$ -direction. It is independent of  $z$ . For this particular form of  $j_z(t)$ , it can be derived from Maxwell's equations of electromagnetism as  $H_x(x, y, t) = j_z(t)h_x(x, y)$  with

$$h_x(x, y) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} e^{2\pi i u x} A(u, y) du \quad (1)$$

where

$$A(u, y) = \frac{\sin(n\pi u)}{u} \times \begin{cases} \exp\left(2\pi i y \sqrt{1 - u^2}\right), & |u| \leq 1 \\ \exp\left(-2\pi |y| \sqrt{u^2 - 1}\right), & |u| > 1 \end{cases} \quad (2)$$

To avoid ambiguity, it is convenient to specify  $A(0, y) = \lim_{u \rightarrow 0} A(u, y)$ .

It can be shown that for large  $y$ , the complex modulus of the magnetic field asymptotically approaches

$$|h_x| \simeq \left| \frac{\sin n\pi v}{2\pi v} \right| \sqrt{\frac{v(1 - v^2)}{x}}, \quad (3)$$

where

$$v = \frac{x}{\sqrt{x^2 + y^2}}.$$

This project investigates numerical approximations to  $h_x(x, y)$ , as defined in (1).

### 2 Numerical method

The right-hand side of (1) is a Fourier integral. Numerical estimation of this function has some tricky features: for example, if  $x$  is large then the integrand oscillates rapidly in  $u$ . This project uses a specialised method for integrals of this type, called the fast Fourier transform (FFT). It is a very efficient method, in particular it allows simultaneous estimation of  $h_x(x, y)$  at  $N$  distinct values of  $x$ .



To apply the method, note first that  $A$  decays rapidly for large  $u$ , so it is reasonable to introduce a (large) parameter  $U$  and approximate  $h_x(x, y)$  as

$$h_x(x, y) \approx \frac{1}{2\pi} \int_{-U}^U e^{2\pi i u x} A(u, y) du \quad (4)$$

This approximation is accurate for sufficiently large  $U$ .

Now define a periodic function  $A^{\text{per}}$  with period  $2U$  by taking  $A^{\text{per}}(u, y) = A(u, y)$  for  $|u| \leq U$  and  $A^{\text{per}}(u + 2mU, y) = A^{\text{per}}(u, y)$  for any integer  $m$ . The integral in (4) is unchanged on replacing  $A$  by  $A^{\text{per}}$ . The domain of integration can then be replaced by  $[0, 2U]$ , and it is natural to estimate the integral by a (Riemann) sum. Define

$$\hat{h}_x(x, y) = \frac{\Delta u}{2\pi} \sum_{k=0}^{N-1} e^{2\pi i k x \Delta u} A^{\text{per}}(k\Delta u, y) \quad (5)$$

with  $\Delta u = 2U/N$ .

Under certain conditions, this allows  $h_x(x, y)$  to be approximated by  $\hat{h}_x(x, y)$ , but the accuracy of this approximation requires some care. For example  $\hat{h}_x$  exhibits rapid oscillations as a function of  $x$ , which are not present in  $h_x$ . Also, the right hand side of (5) can be recognised as a Fourier series (or discrete Fourier transform, DFT). Hence  $\hat{h}_x(x, y)$  is periodic in  $x$ , specifically  $\hat{h}_x(x, y) = \hat{h}_x(x + 2mX, y)$  with  $X = 1/(2\Delta u)$ . However,  $h_x$  is not periodic.

To understand the relation of  $\hat{h}_x$  to  $h_x$ , define a periodic function  $h_x^{\text{per}}$  by taking  $h_x^{\text{per}}(x, y) = h(x, y)$  for  $|x| \leq X$  and  $h_x^{\text{per}}(x + 2mX, y) = h_x^{\text{per}}(x, y)$  for any integer  $m$ . Define also  $\Delta x = 2X/N$ . Then for integer  $m$  and sufficiently large values of  $N$  and  $U$ , one has

$$\hat{h}_x(m\Delta x, y, t) \approx h_x^{\text{per}}(m\Delta x, y, t) . \quad (6)$$

Under these conditions,  $h_x$  can be approximated by  $\hat{h}_x$  as long as  $|x| \leq X$  and  $x = m\Delta x$ . This construction relies on the fact that  $\Delta x \Delta u = 1/N$  so that the exponential factors in (5) are the  $N$ th roots of unity.

The FFT method is an efficient algorithm for computing sums of the form (5), for  $x = m\Delta x$  and  $m = 0, 1, 2, \dots, N - 1$ . This allows accurate estimation of  $h_x^{\text{per}}(m\Delta x, y, t)$  for  $x \in [0, 2X]$  and hence of  $h_x$ . The method is described in the Appendix. For cases where  $N$  is an integer power of 2, the FFT is much faster than computing the sum (5) individually for each value of  $m$  in turn. For this project, it is not necessary to understand any of the details, you only need to invoke an FFT routine to compute the relevant quantities. You may use a Matlab routine such as `fft` or `ifft`, or an equivalent routine in any other language, or you may write your own (but you should not compute (5) directly).

Finally, note that we have defined the method by taking  $N$  and  $U$  as parameters, from which  $\Delta u, \Delta x, X$  are derived. From a practical point of view it is more natural to take  $N$  and  $X$  as parameters, from which one may derive  $U$  and the other relevant quantities.

### 3 Numerical work

**Programming Task:** Given values of  $n, y, N, X$ , write a program to compute (5) by FFT, for  $x = m\Delta x$  and  $m = 0, 1, \dots, N - 1$ . It is sufficient to restrict to  $N = 2^p$  for integer  $p$ . The program should also use (6) to estimate the real and imaginary parts of  $h_x$  for  $x \in [-X, X]$ . Also estimate its complex modulus  $|h_x|$ . It will be necessary to plot these estimates.

**Question 1** Take

$$n = 2, \quad y = 0.2, \quad X = 5, \quad N = 256 .$$

Plot your estimates of the real and imaginary parts of  $h_x$ , and its modulus, for  $|x| < X$ . Derive the relationships between  $h_x(x, y)$  and  $h_x(-x, y)$  and  $h_x(x, -y)$ . Verify that your results are consistent with these relationships.

**Question 2** Keeping  $n = 2$  and  $y = 0.2$ , compute estimates of  $h_x(x, y)$  for  $|x| < 5$ , using different values of  $X$  and  $N$  (always with  $X \geq 5$ ). Analyse the behaviour of your estimates, as  $N$  and  $X$  are varied.

Note: In this question and throughout this project, you should provide graphs that illustrate clearly the effect of the parameters on your results. Note that large numbers of graphs are *very unlikely to be effective* in communicating this information.

**Question 3** For  $n = 2$ , produce a single graph that shows  $|h_x|$  as a function of  $x$  for  $y = 0.12, 0.6, 1, 6, 12$ . Fix  $N = 256$  and choose suitable values of  $X$  (dependent on  $y$ ). Justify the values that you have chosen. Are there some values of  $y$  for which larger (or smaller) values of  $N$  would be appropriate?

Compare your numerical results for large  $y$  with the asymptotic formula (3). This comparison must be presented in a way that illustrates clearly any differences between the numerical estimates and the asymptotic formula. It may be useful to consider additional values of  $y$ , as well as those listed above.

**Question 4** Perform a similar analysis to question 3 but now for  $n = 3, 4$ . Justify your choices of  $N, X$ . Combining these results with those of question 3, discuss how the approximation of  $h$  by  $\hat{h}$  depends on both  $n, y$  and  $N, X$ .

**Question 5** Comment on the physical significance of your results. In particular, how do your results demonstrate the phenomenon of diffraction?

## Appendix: The Fast Fourier Transform

Given a vector of complex numbers  $\mu = (\mu_0, \mu_1, \dots, \mu_{N-1})$ , define

$$\lambda_r = \sum_{k=0}^{N-1} \mu_k e^{-2\pi ikr/N} . \quad (7)$$

The FFT is an efficient (fast) method of evaluating the vector  $\lambda = (\lambda_0, \lambda_1, \dots, \lambda_{N-1})$ , which is the discrete Fourier transform. The same algorithm can also be used to evaluate similar vectors where the factor  $e^{-2\pi ikr/N}$  in the definition of  $\lambda_r$  is replaced by  $e^{2\pi ikr/N}$ , this is sometimes called the inverse FFT.

Note that (7) corresponds to multiplication of the vector  $\mu$  by a particular  $N \times N$  matrix that we denote by  $\Omega^{(N)}$ . Its elements are taken from the set of  $N$ th roots of unity. It follows that  $\lambda$  can be computed using approximately  $N^2$  multiplication operations. (There would be a similar number of addition operations, it is assumed here that the multiplication operations take the greater part of the computational effort.) If  $N = 2^p$  for integer  $p$ , the FFT can compute  $\lambda$  much more quickly, it requires approximately  $(N/2) \log_2 N$  multiplication operations.

To see this, divide  $\mu$  into even and odd subsequences, that is  $\mu^E = (\mu_0, \mu_2, \dots, \mu_{N-2})$  and  $\mu^O = (\mu_1, \mu_3, \dots, \mu_{N-1})$ . Their Fourier transforms are given by matrix multiplication as

$$\lambda^E = \Omega^{(N/2)} \mu^E, \quad \lambda^O = \Omega^{(N/2)} \mu^O. \quad (8)$$

Then it may be shown that

$$\left. \begin{aligned} \lambda_r &= \lambda_r^E + e^{2\pi i r/N} \lambda_r^O \\ \lambda_{r+N/2} &= \lambda_r^E - e^{2\pi i r/N} \lambda_r^O \end{aligned} \right\} \quad r = 0, 1, \dots, \frac{N}{2} - 1 \quad (9)$$

Hence if  $\lambda^E$  and  $\lambda^O$  are known, it requires  $(N/2)$  multiplications to evaluate  $\lambda$ .

Moreover, since  $\lambda^E$  is itself the Fourier transform of a particular sequence  $\mu^E$ , it can be estimated efficiently by further splitting  $\mu^E$  into even and odd subsequences. For  $N = 2^p$ , this decomposition is repeated  $p$  times, leading to an FFT in  $p$  stages.

In stage 1, each element  $\mu_k$  of  $\mu$  is treated as a sequence  $\mu^{(k,1)}$  of length 1. Their Fourier transforms are simply  $\lambda_0^{(k,1)} = \mu_0^{(k,1)}$ . These sequences are labelled as even/odd, and are combined in pairs using a rule similar to (9), which generates  $N/2$  sequences each of length 2. These are denoted as  $\lambda^{(k,2)}$  for  $k = 0, 1, 2, \dots, (N/2) - 1$ . In stage 2, these new sequences are again labelled as even/odd and combined in pairs using the generalised (9), to obtain  $N/4$  sequences of length 4, denoted by  $\lambda^{(k,4)}$  for  $k = 0, 1, 2, \dots, (N/4) - 1$ . The procedure repeats until stage  $p$  ends with a single sequence  $\lambda^{(0,2^p)}$  of length  $2^p$ .

The detailed rules that explain how the sequences are combined can be found in the original paper [1] or in standard textbooks such as [2]. These are chosen such that  $\lambda^{(0,2^p)} = \lambda$ , the vector of interest.

For efficiency, the key point is that each step requires  $N/2$  multiplication operations and there are  $p = \log_2 N$  stages. Hence the algorithm only requires  $(N/2) \log_2 N$  multiplication operations, as advertised above.

## References

- [1] JW Cooley, and JW Tukey, *An algorithm for the machine calculation of complex Fourier series*, Math. Comput. 19: 297-301 (1965)
- [2] TH Cormen, CE Leiserson, RL Rivest, and C Stein, Chapter 30 of *Introduction to Algorithms*, MIT press, 3rd edition (2009)

# 7 Mathematical Methods

## 7.5 Padé Approximants

(8 units)

*This project is essentially self-contained, though the Part II course Asymptotic Methods provides background to Question 4.*

### 1 Introduction

A Padé approximant is a rational function, i.e., a function expressed as a fraction whose numerator and denominator are both polynomials, whose power series expansion agrees with a given power series to the highest possible order.

The primary application of Padé approximants is to problems where it is possible to derive the solution formally as a power series expansion in some parameter. The corresponding Padé approximants often turn out to be much more useful than the power series itself (in a sense to be explored in this project).

Given the power series

$$f(x) = \sum_{k=0}^{\infty} c_k x^k \quad (1)$$

the  $[L, M]$  Padé approximant  $R_{L,M}(x)$  is defined by

$$R_{L,M}(x) = \frac{\sum_{k=0}^L p_k x^k}{1 + \sum_{k=1}^M q_k x^k} \quad (2)$$

such that

$$f(x) - R_{L,M}(x) = O(x^{L+M+1}), \quad (3)$$

i.e., the first  $L + M + 1$  terms of the power series of  $R_{L,M}(x)$  match the first  $L + M + 1$  terms of the power series of  $f(x)$ .

Equations for the coefficients  $p_k$ ,  $k = 0, \dots, L$  and  $q_k$ ,  $k = 1, \dots, M$  can be obtained by multiplying (3) by the denominator of  $R_{L,M}(x)$  and equating coefficients of  $x^k$  for  $k = 0, \dots, L + M$ .

The result is  $M$  simultaneous equations for the  $q_k$ ,  $k = 1, \dots, M$ ,

$$\sum_{k=1}^{\min(r,M)} q_k c_{r-k} = -c_r \quad (r = L + 1, \dots, L + M) \quad (4)$$

and  $L + 1$  expressions for the  $p_k$ ,  $k = 0, \dots, L$ ,

$$p_k = c_k + \sum_{s=1}^{\min(k,M)} q_s c_{k-s} \quad (k = 0, \dots, L) \quad (5)$$

In many cases it is convenient to consider only ‘diagonal’ Padé approximants with  $L = M$ . But sometimes this may not be possible, e.g., for special forms of the power series the simultaneous equations (4) corresponding to diagonal approximants may not have a solution. In that case it may be convenient to choose  $M = L + 1$ , or something similar.

**Programming Task:** You will need to write two general purpose programs for this project. You should use MATLAB's 64-bit (8-byte) double-precision floating-point values or the equivalent in other programming languages.

Program A should solve equations (4) and (5) given the coefficients  $c_k$  and the values of  $L$  and  $M$  and evaluate the resulting Padé approximant  $R_{L,M}(x)$  for a specified set of values of  $x$ . You may use a library routine to solve the simultaneous equations (4). For example, if using MATLAB you can use the built-in matrix division routines such as `mldivide`. If you are not using MATLAB it may be worth using iterative improvement. (See Appendix.)

Program B should find the (possibly complex) roots of a polynomial, given the coefficients. If using MATLAB, the `roots` routine makes this program particularly easy to write. Alternatively, two straightforward possibilities are discussed in section 9.5 of [3]. If a (possibly complex) roots of a polynomial routine is available with the programming language you choose, just use it.

## 2 Estimating functions defined by power series

Consider the function  $f_1(x) = (1+x)^{1/2}$ .

**Question 1** Derive the power series expansion for  $f_1(x)$  about  $x = 0$ , deducing a formula for the coefficients  $c_k$  for arbitrary  $k$ . What is the radius of convergence of the power series and what limitations does this put on using the power series to estimate  $f_1(x)$ ? Noting that the power series converges for  $x = 1$  investigate the convergence of the partial sums  $\sum_{k=0}^N c_k$  as  $N$  increases and display selected results. Regarding the partial sum as an estimate for  $\sqrt{2}$ , how does the error vary with  $N$  as  $N$  increases?

**Question 2** Use your program to determine the Padé approximant  $R_{L,L}(x)$  and evaluate this for  $x = 1$ . Again regarding this as an estimate for  $\sqrt{2}$ , how does the error vary with  $L$  as  $L$  increases? What is the smallest value to which the error can be reduced? What determines this smallest value? Does iterative improvement to the solution of (4) make any difference?

Compare the results for the power series and for the Padé approximant. Which method would you recommend to give an estimate for  $\sqrt{2}$  to specified accuracy?

**Question 3** Now consider  $x$  in the range  $1 < x \leq 100$ . Compare power series estimates and Padé approximant estimates for  $f_1(x)$  for a few choices of  $N$  and  $L$ . Display the results graphically and discuss. For two chosen values of  $x$  (e.g.,  $x = 10$  and  $x = 100$ ) investigate carefully how the error in the Padé approximant estimates varies as  $L$  increases, display the results graphically and discuss. What are the implications for using the Padé approximant to estimate  $f_1(x)$  for large  $x$ ?

Now consider the function  $f_2(x) = \int_0^\infty e^{-t}(1+xt)^{-1}dt$ , which is defined for all real  $x \geq 0$  (in fact, everywhere in the complex  $x$ -plane except the negative real axis). Replacing  $(1+xt)^{-1}$  by its Maclaurin expansion and integrating term-by-term gives the *asymptotic expansion*

$$1 - 1!x + 2!x^2 - 3!x^3 + 4!x^4 - 5!x^5 + \dots \quad (6)$$

This diverges for all  $x \neq 0$ , which is hardly surprising since the Maclaurin expansion of  $(1+xt)^{-1}$  diverges for  $t \geq x^{-1}$ . Nevertheless, *when truncated at a finite number of terms*, the series gives

a “good” approximation to  $f_2(x)$  when  $x$  is “small”. (A more precise statement of this result, and its justification by Watson’s Lemma, can be found in [1] or [2], or any of the books listed in the schedule for the Part II *Asymptotic Methods* course.)

**Question 4** Regarding the asymptotic series (6) as a power series, use program A to generate Padé approximants for  $f_2(x)$ . Compare the truncated power series and the Padé approximants as a basis for calculating  $f_2(x)$  on the range  $0 \leq x \leq 20$ . Note that numerical integration gives the following values, correct to eight decimal places:

$x$	$f_2(x)$
0.1000	0.91563334
0.2000	0.85211088
0.3000	0.80118628
0.4000	0.75881459
0.5000	0.72265723
0.6000	0.69122594
0.7000	0.66351027
0.8000	0.63879110
0.9000	0.61653779
1.0000	0.59634736
2.0000	0.46145532
3.0000	0.38560201
4.0000	0.33522136
5.0000	0.29866975
6.0000	0.27063301
7.0000	0.24828135
8.0000	0.22994778
9.0000	0.21457710
10.0000	0.20146425
11.0000	0.19011779
12.0000	0.18018332
13.0000	0.17139800
14.0000	0.16356229
15.0000	0.15652164
16.0000	0.15015426
17.0000	0.14436271
18.0000	0.13906806
19.0000	0.13420555
20.0000	0.12972152

### 3 Zeros and poles

**Question 5** Use Program B to determine (in the complex  $x$ -plane) the poles and zeros of the Padé approximant  $R_{L,L}(x)$  for  $f_1(x)$ . Investigate carefully how the positions of the poles and zeros change as  $L$  is increased.

Carry out the same investigation for the functions  $f_3(x) = (1+x)^{-1/2}$ ,  $f_4(x) = e^x$ ,  $f_5(x) = e^x/(1+x)$  and  $f_6(x) = (1+x+x^2)^{1/2}$ . [For  $f_5(x)$  and  $f_6(x)$  your program will have to do some straightforward calculation to evaluate the coefficients in the power series.]

On this basis can you suggest how the positions of poles and zeros of the Padé approximants correspond to any poles, zeros, branch points and branch cuts of the approximated functions? Relate your comments carefully to specific properties of each of the functions considered. Provide a selection of results in the form of plots or short tables to support your comments.

Do you find ‘anomalous’ poles and zeros of the approximants that do not match poles, zeros, branch points or branch cuts of the approximated function? You will find many such cases for  $f_4(x)$  and  $f_5(x)$ , but should also find cases for  $f_1(x)$  and  $f_3(x)$ , particularly when  $L$  is large. What do you notice about the anomalous poles and zeros in these latter cases?

Comment on any problems that might be encountered in using Padé approximants to estimate  $f_6$  along the real  $x$ -axis. Display one or two relevant graphs.

## Appendix: Iterative improvement of the solution of linear simultaneous equations

Consider the set of equations  $A\mathbf{x} = \mathbf{b}$ , where  $A$  is a square matrix,  $\mathbf{b}$  is the column vector of right-hand sides and  $\mathbf{x}$  is the column vector of unknowns.

Suppose that numerical solution has generated the approximate solution  $\mathbf{y}$ . Now suppose that the true solution is given by  $\mathbf{x} = \mathbf{y} + \delta\mathbf{y}$ . Multiplying by  $A$  implies that  $A\delta\mathbf{y} = \mathbf{b} - A\mathbf{y}$ . This is a set of equations for the correction  $\delta\mathbf{y}$  to the approximate solution and solving gives an estimate for the correction, and hence a refinement to the solution. This procedure may be repeated until no further improvement is found.

Note that at each refinement the set of simultaneous equations to be solved has the same associated matrix  $A$ . Only the right-hand sides change. Therefore there is advantage in using an approach such as  $LU$  decomposition, since once the  $LU$  decomposition of  $A$  has been calculated it may be used repeatedly to solve the simultaneous equations occurring at each refinement.

## References

- [1] Hinch, E.J., *Perturbation Methods*.
- [2] Bender, C. & Orszag, S.A., *Advanced Mathematical Methods for Scientists and Engineers*.
- [3] Press *et al.*, *Numerical Recipes in C*.

# 7 Mathematical Methods

## 7.6 Insulation (10 units)

*There are no prerequisites for this project.*

### 1 Introduction

When sheets of plastic and of other insulating materials are used in the construction of building walls and ceilings, a balance is sought between the need to minimise the loss of heat and the need to include (non-insulating) holes to allow gases, especially water vapour, to pass. In this project we will model a two dimensional analogue of this problem. A perfectly insulating sheet containing equally spaced holes is placed between two constant temperature surfaces and the resulting steady-state temperature distribution is computed. The steady-state temperature distribution is computed by solving Laplace's equation using a so-called relaxation method.

We start from the non-steady-state (Poisson's) equation appropriate to heat conduction,

$$\kappa \nabla^2 T = \rho s \frac{\partial T}{\partial t} , \quad (1)$$

where  $T$  is temperature,  $t$  is time,  $\kappa$  is the thermal conductivity of the medium,  $\rho$  is its density and  $s$  is its specific heat. We then impose the steady-state condition  $\partial T / \partial t = 0$ , to obtain

$$\nabla^2 T = 0 , \quad (2)$$

that is, Laplace's equation.

### 2 Empty unit square

In this section we will consider a unit square or "box"  $0 \leq x \leq 1$  and  $0 \leq y \leq 1$  in which the left side  $x = 0$  is held at constant temperature 0, the right side  $x = 1$  at constant temperature 1, and in which the top and bottom boundaries  $y = 0, 1$  are perfect insulators, so that there is no temperature flux across the top and bottom. Thus the boundary conditions are

$$\begin{aligned} T(0, y) &= 0 \\ T(1, y) &= 1 \\ \frac{\partial T}{\partial y}(x, 0) &= \frac{\partial T}{\partial y}(x, 1) = 0 \end{aligned}$$

The analytic solution to this problem is easy to find; this therefore allows us to check the accuracy of our numerical method and confirm that it is working correctly.

Consider the following  $N_x \times N_y$  discretisation of the unit square:

$$\begin{aligned} x_i &= i\Delta x, & i &= 0, 1, \dots, N_x, & \Delta x &= \frac{1}{N_x} , \\ y_j &= j\Delta y, & j &= 0, 1, \dots, N_y, & \Delta y &= \frac{1}{N_y} . \end{aligned}$$



Let us adopt the notation  $T_{i,j} = T(x_i, y_j)$  for the numerical solution to (2). We approximate  $\partial^2 T / \partial x^2$  and  $\partial^2 T / \partial y^2$  to second order at a general internal (i.e., away from the boundary) point  $(x_i, y_j)$  with

$$\begin{aligned}\frac{\partial^2 T}{\partial x^2} &= \frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{\Delta x^2}, \\ \frac{\partial^2 T}{\partial y^2} &= \frac{T_{i,j+1} - 2T_{i,j} + T_{i,j-1}}{\Delta y^2}.\end{aligned}$$

By choosing the special case  $\Delta x = \Delta y = \Delta$ , which implies  $N_x = N_y$ , Laplace's equation can therefore be written in discretised form to second order as

$$0 = -4T_{i,j} + T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1}. \quad (3)$$

In this project (3) is solved using a relaxation method. So-called multigrid methods are faster, but their extra complications are not necessary here (see Press et al., 1992 for details). The relaxation method iteration is started by choosing *any* initial values for the  $T_{i,j}$ ; initial values that are close to the solution to (3) will converge quicker than initial values that are very different from the solution. Each subsequent iteration step consists of computing new values for  $T_{i,j}$  successively for each internal point using the relaxation algorithm:

$$T_{i,j}^{\text{new}} = (1 - \sigma)T_{i,j}^{\text{old}} + \frac{\sigma}{4} (T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1}), \quad (4)$$

where  $\sigma$  is a pre-determined constant (see below).

New values for  $T_{i,j}$  on the boundary are then computed by applying the appropriate boundary conditions. Applying the first two boundary conditions is simply a matter of not changing the boundary points  $T_{0,j}$  and  $T_{N_x,j}$ . The third boundary condition can be obtained by using a central difference approximation to  $\partial T / \partial y$  at a general boundary point  $(x_i, y_j)$ :

$$\frac{\partial T}{\partial y} = \frac{T_{i,j+1} - T_{i,j-1}}{2\Delta} = 0. \quad (5)$$

This approximation is used with (4) to compute a new value for  $T_{i,j}$  on the boundary in terms of points that are either on the boundary or internal. For example the bottom  $j = 0$  boundary points are computed using

$$T_{i,0}^{\text{new}} = (1 - \sigma)T_{i,0}^{\text{old}} + \frac{\sigma}{4} (T_{i+1,0} + T_{i-1,0} + 2T_{i,1}), \quad (6)$$

and an analogous formula is used for the top  $j = N_y$  boundary points.

Further iterations are carried out until the  $T_{i,j}$  have converged at which time they are a solution to (3). The special case  $\sigma = 1$ , obtained by rearrangement of (3), is called the Jacobi relaxation method. However, in practice, convergence can be considerably faster if an appropriate value  $\sigma > 1$  is chosen (called over-relaxation). For all the cases studied in this project,  $\sigma \approx 1.9$  is a good first choice.

**Programming Task:** Write a program to solve (2) for the unit square  $0 \leq x \leq 1$  and  $0 \leq y \leq 1$  with the given boundary conditions. You should use MATLAB's 64-bit (8-byte) double-precision floating-point values or the equivalent in other programming languages. Your program should plot contours of  $T$  and should include some mechanism for deciding whether the  $T$  array has converged. You should choose  $N_x = N_y$  that is large enough for the contours of  $T$  to be well resolved but small enough that the  $T$  array converges in a reasonable time.

Note that equation (4) does not specify whether some of the values of  $T$  on the right-hand side are the “old” or the “new” values. Either choice will work, but it is most efficient to “sweep” through the matrix updating each value of  $T_{i,j}$  immediately as you go along; so in fact, some of the  $T$ -values on the right-hand side will be “old” (because they haven’t yet been updated) and some will be “new” (because the sweeping process has already updated them). This improvement to the Jacobi method, which surprisingly increases its rate of convergence, is called the Gauss–Seidel method.

**Question 1** Describe how you decided whether the  $T$  array had converged. Experiment with values of  $\sigma$  and comment on the effect of changing  $\sigma$  on the number of iterations required for convergence. Include in your write-up a contour plot of the steady-state temperature distribution of  $T(x, y)$ .

### 3 Unit square and vertical insulating wall with holes

In this section we place a vertical insulating “wall” across the unit square of section §2 with holes of size  $\epsilon$  spaced a distance  $\delta$  apart. Figure 1 shows a schematic example for the case  $\epsilon = 2\Delta$ ,  $\delta = 5\Delta$ . The top of the wall is shown expanded to show more clearly new types of boundary gridpoint labelled A–H. New boundary conditions (similar to (6)) will be needed for these types of gridpoint and for I and J at the bottom of the wall. At corners of the wall (e.g., gridpoint F) you should assume that the corner is smoothed off (rounded) so that the normal is diagonal to the grid there.

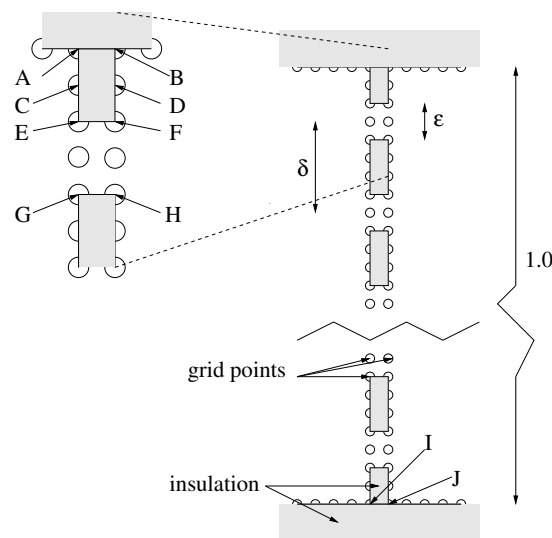


Figure 1: Insulating wall with holes

**Programming Task:** Modify the program of section §2 to include a vertical insulating wall at the middle of the box (or as close as possible) with holes of size  $\epsilon$ , spaced  $\delta$  apart. Your program should arrange the holes and the insulator pieces as symmetrically as possible about the centre of the box so that there is a length of insulator at the top and bottom of the wall as in Fig. 1. Explain *briefly* how your program assigns the locations of the insulating sections and how it deals with special cases. Note that it is not always possible to obtain perfect symmetry.

**Question 2** Write down and explain the formulae you used to compute  $T_{i,j}^{\text{new}}$  at the new types of boundary grid point A–J. Include in your write-up a contour plot of the steady-state temperature distribution of  $T(x, y)$  for the case  $\epsilon = 4\Delta$ ,  $\delta = 8\Delta$ ,  $N_x = N_y = 129$ . Why is 129 a better choice than 128? What effect does putting holes in the wall have on the temperature distribution in the unit square? Examine what happens to temperature along horizontal cross-sections through the temperature distribution by plotting  $T(x, y_0)$  against  $x$  for a few values of  $y_0$ . Choose values of  $y_0$  that are near the centre of the box and make sure you include cases that go through the centre of a hole and through the the centre of an insulating section.

The total heat flux across the boundary  $x = 1$  is a gauge of the quality of the insulating layer. Given that the heat flux at any point  $(x, y)$  is given by  $-\kappa \nabla T(x, y)$ , where  $\kappa$  is the thermal conductivity from equation (1), define a suitable measure  $Q$  of the insulator's quality. How would  $Q$  differ for a good insulator versus a good conductor? Comment on the insulating properties of the wall with  $\epsilon = 4\Delta$ ,  $\delta = 8\Delta$ .

**Question 3** Investigate what happens to the insulator quality  $Q$  when you vary  $\epsilon$  and  $\delta$  (but keep  $N_x = N_y$  constant) according to

$$\epsilon = k\delta^\alpha, \tag{7}$$

where  $k$  and  $\alpha$  are suitable real constants. In our discrete model, since  $\epsilon$  and  $\delta$  must be multiples of  $\Delta$ , you would need to choose the nearest integer multiple of  $\Delta$  for  $\epsilon$  for a given  $\delta$  or vice-versa. Include in your write-up a few plots that illustrate what happens to  $Q$  as you decrease  $\delta$ : choose relationships that show interesting behaviour. Comment on your plots and on the physical significance of (7).

## 4 Infinite length vertical insulating wall with holes

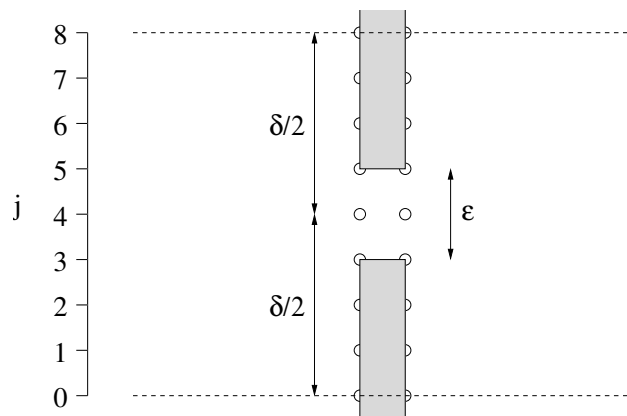


Figure 2: Insulating wall with holes, periodic boundary

In this section we simulate an infinitely long vertical insulating wall with holes by taking a single, finite section of the wall and imposing a periodic boundary condition at the top and bottom of the region. A periodic boundary condition means  $T(x, y_{\text{bot}}) = T(x, y_{\text{top}})$ . With this boundary condition only one central hole is needed. Figure 2 shows a schematic example for the case  $\epsilon = 2\Delta$ ,  $\delta = 8\Delta$  so that  $T_{i,8} = T_{i,0}$ . Thus for the case shown in Fig. 2, the  $j = 0$

(boundary) points are computed using (4) but with the periodic boundary taken into account by replacing  $j$  with  $j$  modulo 8, thus

$$T_{i,0}^{\text{new}} = (1 - \sigma)T_{i,0}^{\text{old}} + \frac{\sigma}{4} (T_{i+1,0} + T_{i-1,0} + T_{i,1} + T_{i,7}) . \quad (8)$$

Similarly, the  $j = 7$  points are computed using

$$T_{i,7}^{\text{new}} = (1 - \sigma)T_{i,7}^{\text{old}} + \frac{\sigma}{4} (T_{i+1,7} + T_{i-1,7} + T_{i,0} + T_{i,6}) . \quad (9)$$

**Programming Task:** Write a program to implement a relaxation method algorithm for the situation illustrated in Fig. 2 but for general  $N_x$ ,  $\delta$  and  $\epsilon$ .

**Question 4** Investigate what happens to  $Q$  when you vary  $\epsilon$  and  $\delta$  (but keep  $N_x$  constant) in this new model. Include in your write-up a couple of illustrative plots. Is there any need to change your definition of the quality  $Q$ ? How does this periodic boundary condition model relate to the model in Question 3?

**Question 5** Adapt the infinite length vertical insulating wall with holes program for the case  $\delta = 8\epsilon$  to investigate what happens to the steady-state temperature distribution near the hole(s) as the wall thickness is varied. Try values of wall thickness in the range  $\frac{\epsilon}{4}$  to  $4\epsilon$ . To do this you will need to use as large a number of gridpoints as is consistent with a not unreasonable run time of your program.

Note that you may find it helpful to start with the smallest wall thickness, save the gridpoint temperature distribution, use it to construct a first approximation for the next wall thickness and repeat the process.

**Question 6** The original task was to investigate, for insulating sheets, the balance between the need to minimise the loss of heat and the need to include (non-insulating) holes to allow gases, especially water vapour, to pass. In the light of what you have learned from this model, what advice would you give a manufacturer of insulating sheets? What are the limitations of the model and what steps could be taken towards greater realism?

## Reference

Press, W. H., Teukolsky, S. A., Vetterling, W. T., Flannery, B. P., 2002: *Numerical recipes in C: The art of Scientific computing*, Cambridge University Press.

## 9 Operational Research

### 9.3 Protein Comparison in Bioinformatics (8 units)

*This project is computationally intensive but mostly self-contained mathematically. Some understanding of random variables (covered in the Part IA Probability course) is required.*

#### Introduction

Sequence comparison and alignment, combined with the systematic collection and search of databases containing biomolecular sequences, both DNA and protein, has become an essential part of modern molecular biology. Molecular sequence data is important because two proteins with similar sequences often have similar functions or structures. This means that we can learn about the function of a protein in humans by studying the functions of proteins with similar sequences in simpler organisms such as yeast, fruit flies, or frogs.

In this project we will examine methods for comparison of two sequences.

We will work with two strings  $S$  and  $T$  of lengths  $m$  and  $n$  respectively, composed of characters from some finite alphabet. Write  $S_i$  for the  $i$ th character of  $S$ , and  $S[i, j]$  for the substring  $S_i, \dots, S_j$ . If  $i > j$  then  $S[i, j]$  is the empty string. A *prefix* of  $S$  is a substring  $S[1, k]$  for some  $k \leq m$  (possibly the empty prefix). Similarly, a *suffix* of  $S$  is a substring  $S[k, m]$  with  $k \geq 1$ .

#### 1 The edit distance

This section follows work originally done by Needleman and Wunsch [4], although the notation we use is slightly different.

Suppose  $S = \text{fruit}$  and  $T = \text{berry}$ . We can transform  $S$  into  $T$  by

1. **R**Replacing f with b,
2. **I**nserting e,
3. **M**atching r with r,
4. **R**eplacing u with r,
5. **R**eplacing i with y,
6. **D**eleting t.

We call RIMRRD the *editing transcript*. The *alignment* of  $S$  and  $T$  is read vertically character by character and is given by:

```
RIMRRD
f ruit
berry
```

There are, of course, many possible ways to transform one string into another; but from an evolutionary point of view there must be a cost associated with any action other than matching. The *optimal* edit transcripts are those that involve the least number of edit operations (**R**eplace, **I**nsert, and **D**elete). We define the *edit distance*  $d(S, T)$  to be the minimal number of edits between  $S$  and  $T$ . Let  $D(i, j) = d(S[1, i], T[1, j])$ . Observe that  $d(S, T) = D(m, n)$ .

**Question 1** Prove that for all  $i, j > 0$

$$D(i, j) = \min\{D(i-1, j) + 1, D(i, j-1) + 1, D(i-1, j-1) + s(S_i, T_j)\},$$

where  $s(a, b)$  is some suitable function which you should determine. Explain your reasoning carefully. What boundary conditions  $D(0, 0)$ ,  $D(i, 0)$ , and  $D(0, j)$  did you use?

**Question 2** Write a program to find the edit distance between two strings. Use it to find the edit distance between `shesells` and `seashells`. What is the complexity of your algorithm?

Usually, we are interested in finding an optimal editing transcript and alignment of  $S$  and  $T$  rather than just  $d(S, T)$ . This can be done by assigning a pointer when calculating  $D(i, j)$ , pointing to one of  $D(i-1, j)$ ,  $D(i, j-1)$ , or  $D(i-1, j-1)$ .

**Question 3** Modify your algorithm so as to produce one possible optimal alignment between two strings. Take proteins  $A$  and  $B$  from the file `proteins.txt` on the CATAM website. (Both proteins are myoglobin, protein  $A$  is for the duckbill platypus and protein  $B$  for yellowfin tuna.) Find the edit distance between them, and give the first 50 steps of an optimal alignment.

## 2 Scoring matrix

A protein is essentially a long sequence of amino acids. Approximately twenty types of amino acid (the exact number is species dependent) are involved in the construction of each protein. A gene is a sequence of DNA which can be translated into a sequence of amino acids, i.e., a protein. Mutations in DNA will lead to changes in the sequence of amino acids, and some mutations are more likely than others. In this section we adjust our scoring algorithm in order to capture some of these biological considerations.

The adjustment is achieved by replacing the scoring function  $s(a, b)$  which you found in Section 1. There are various schemes for assessing the probability of a mutation from amino acid  $a$  to amino acid  $b$ ; currently the two dominant schemes are the PAM matrices introduced by Dayhoff [2], and the BLOSUM matrices of Henikoff and Henikoff [3].

For historical reasons, we will talk about maximising a score rather than minimizing a distance. Let  $v(S, T)$  be the maximum score of all edit transcripts from  $S$  to  $T$ .

**Question 4** Using the BLOSUM matrix `blosum.txt` from the CATAM website for the scoring function  $s$ , and scoring  $-8$  for each **I**nsert or **D**elate, find the score  $v$  between proteins  $A$  and  $B$  and give the first 50 steps of the optimal alignment.\*

## 3 Scoring for gaps

Some mechanisms for DNA mutations involve the deletion or insertion of large chunks of DNA. Proteins are often composed of combinations of different domains from a relatively small repertoire; so two protein sequences might be relatively similar over several regions, but differ in other regions where one protein contains a certain domain but the other does not.

---

\*If your version of MATLAB has the *Bioinformatics Toolbox* installed, the appropriate BLOSUM matrix can be generated using the command `blosum(62, 'order', 'CSTPAGNDEQHRKMILVFW')`.

At some computational cost, we can still align two protein strings taking gaps into account. Let  $w(l) < 0$ ,  $l \geq 1$ , be the score of deleting (or inserting) a sequence of amino acids of length  $l$  from (or into) a protein. Let  $v_{\text{gap}}(S, T)$  be the gap-weighted score between  $S$  and  $T$ , and write  $V_{\text{gap}}(i, j)$  for  $v_{\text{gap}}(S[1, i], T[1, j])$ . Then

$$\begin{aligned} V_{\text{gap}}(i, j) &= \max \{E(i, j), F(i, j), G(i, j)\}, \\ E(i, j) &= \max_{0 \leq k \leq j-1} \{V_{\text{gap}}(i, k) + w(j - k)\}, \\ F(i, j) &= \max_{0 \leq k \leq i-1} \{V_{\text{gap}}(k, j) + w(i - k)\}, \\ G(i, j) &= V_{\text{gap}}(i - 1, j - 1) + s(S_i, T_j). \end{aligned}$$

Iterating the above equations on the  $n$  by  $m$  grid has complexity of  $O(mn^2 + nm^2)$ . Happily, if  $w(l)$  takes some fixed value  $u$  for all  $l \geq 1$ , then there exists an algorithm for finding  $v_{\text{gap}}$  which has complexity  $O(mn)$ .

**Question 5** Find and implement such an algorithm. Explain how your algorithm works, and why it has complexity  $O(mn)$ . What boundary conditions do you use?

**Question 6** Take proteins  $C$  and  $D$  from the file *proteins.txt* on the CATAM website. (Both proteins are keratin structures in humans.) Using the BLOSUM matrix from Section 2 for the scoring function  $s$ , and  $u = -12$  as the fixed score of insertion/deletion, find the gap-weighted score  $v_{\text{gap}}(C, D)$  and give the first 50 steps of the optimal alignment.

## 4 Statistical significance

We may now ask at what threshold a score  $v_{\text{gap}}(S, T)$  should be declared to have biological significance?

Let us simplify the problem slightly. Suppose there are only two letters in our alphabet,  $a$  and  $b$ , corresponding, say, to hydrophobic and hydrophilic amino acids. Let  $s(a, a) = s(b, b) = 1$  and  $s(a, b) = s(b, a) = -1$ . Let  $U^n$  be a random protein of length  $n$ : all the amino acids  $U_1^n, \dots, U_n^n$  are independent and identically distributed, with  $P(U_i^n = a) = p$  and  $P(U_i^n = b) = 1 - p$ .

**Question 7** Consider two random proteins  $U^n$  and  $V^n$ , independent and identically distributed. Let the score of inserting/deleting a sequence of length  $l$  be fixed:  $w(l) = u$  for all  $l \geq 1$ . Prove that for all  $0 \leq p \leq 1$ , and for all  $u \leq 0$ ,

$$\liminf_{n \rightarrow \infty} \frac{\mathbb{E}(v_{\text{gap}}(U^n, V^n))}{n} > 0.$$

(Note: if  $x_n$  is a sequence of real numbers, then  $\liminf_{n \rightarrow \infty} x_n$  is defined by

$$\liminf_{n \rightarrow \infty} x_n = \lim_{n \rightarrow \infty} \inf_{m > n} x_m.$$

Note that the limit is always guaranteed to exist, though it may be  $+\infty$  or  $-\infty$ . This is discussed further in most textbooks on analysis.)

In fact,  $\lim_{n \rightarrow \infty} n^{-1} \mathbb{E}(v_{\text{gap}}(U^n, V^n))$  exists and is strictly positive. One way to obtain an excellence mark in this project, though not the only way, is to show that this limit exists.

**Question 8** Let  $u = -3$ ,  $p = \frac{1}{2}$ . Write a program to estimate  $n^{-1} \mathbb{E}(v_{\text{gap}}(U^n, V^n))$ . Now vary  $n$  and estimate the limit of  $n^{-1} \mathbb{E}(v_{\text{gap}}(U^n, V^n))$ . Explain how you arrive at your estimate of the limit.

## 5 Local Alignment

Full alignment of proteins is meaningful when the two strings are members of the same family. For example, the full sequences of the oxygen-binding proteins myoglobin and haemoglobin are very similar. Often, though, only a small region of the protein is critical to its function and only this region will be conserved throughout the evolutionary process. When we identify two proteins which perform similar functions but look superficially different, it is useful to identify these highly conserved regions.

We aim to find a pair of substrings  $S'$  and  $T'$  of  $S$  and  $T$  with the highest alignment score, namely,

$$v_{\text{sub}}(S, T) = \max\{v(S', T') : S' \text{ a substring of } S, T' \text{ a substring of } T\}.$$

(For simplicity, we will use the same scoring as in Section 2. We will also write  $s(-, a) = s(a, -) < 0$  for the score of an insertion or deletion.)

Finding  $v_{\text{sub}}(S, T)$  seems to be of much higher complexity than solving the global alignment problem, as there are  $\Theta(n^2m^2)$  combinations of substrings of  $S$  and  $T$ . Amazingly, we will solve it using an algorithm whose complexity is still only  $O(mn)$ .

We will first define a slightly easier problem. Suppose we restrict ourselves to suffixes of  $S$  and  $T$ :

$$v_{\text{sfx}}(S, T) = \max\{v(S', T') : S' \text{ a suffix of } S, T' \text{ a suffix of } T\}.$$

**Question 9** Prove carefully that

$$v_{\text{sub}}(S, T) = \max\{v_{\text{sfx}}(S', T') : S' \text{ a prefix of } S, T' \text{ a prefix of } T\}.$$

**Question 10** Write  $V_{\text{sfx}}(i, j)$  for  $v_{\text{sfx}}(S[1, i], T[1, j])$ . Prove that

$$V_{\text{sfx}}(i, j) = \max \begin{cases} 0, \\ V_{\text{sfx}}(i-1, j-1) + s(S_i, T_j), \\ V_{\text{sfx}}(i-1, j) + s(S_i, -), \\ V_{\text{sfx}}(i, j-1) + s(-, T_j), \end{cases}$$

with boundary conditions  $V_{\text{sfx}}(i, 0) = V_{\text{sfx}}(0, j) = 0$ .

**Question 11** Find  $v_{\text{sub}}$  for proteins  $C$  and  $D$ , using the BLOSUM matrix from Section 2 and  $s(a, -) = s(-, a) = -2$  for all amino acids  $a$ .

## References

- [1] Altschul, S. and Erickson, B.W. *Optimal sequence alignment using affine gap costs*. Bulletin of Mathematical Biology 48: 603-616, 1986
- [2] Dayhoff, M.O., Schwartz, R.M. and Orcutt, B.C. *A model of evolutionary change in proteins*. Atlas of Protein Sequence and Structure 5: 345-352, 1978
- [3] Henikoff, S. and Henikoff J.G. *Amino acid substitution matrices from protein blocks*. Proceedings of the National Academy of Science 89: 10915-10919, 1992
- [4] Needleman, S.B. and Wunsch, C.D. *A general method applicable to the search for similarities in the amino acid sequence of two proteins*. Journal of Molecular Biology 48: 443-453, 1970



# 9 Operational Research

## 9.5 The Google PageRank Algorithm (5 units)

*This project requires an understanding of the Part IB Markov Chains course. Familiarity with Linear Algebra is desirable.*

### 1 Introduction

*PageRank* is a link analysis algorithm, operating on a database of documents connected to each other via directional *hyperlinks*. It was developed to measure the relative importance of a webpage in the World Wide Web, and with minor variations has also been employed in the context of assigning importance to academic journal publications.

**Graph-theoretic terminology** We will represent a collection of hyperlinked documents (webpages, academic journals, etc.) as a *directed graph*  $G = (V, E)$ , where  $V$  is the set of documents  $\{d_1, \dots, d_N\}$  and the edge set  $E \subseteq V \times V$  can be represented by an  $N \times N$  *adjacency matrix*  $A$ , where  $A_{ij} = 1$  iff  $d_j \rightarrow d_i$  (i.e., iff  $(d_j, d_i) \in E$ ). The *out-degree* of a node  $i$  is the number of outgoing edges  $d_i \rightarrow d_j$ . A node with out-degree 0 is called a *dangling* node. Multiple edges can be incorporated by letting  $A_{ij} = d$  when there are  $d$  edges  $j \rightarrow i$ .

### 2 The PageRank algorithm

PageRank may be motivated as a *voting system*. Each webpage can distribute a total vote of 1 to other webpages, and votes themselves are weighted according to the importance of the respective voter, giving rise to the following recursion for the score  $w_i$  of the  $i$ th webpage:

$$w_i = \sum_{j=1:N} S_{ij} w_j, \quad \text{where } S_{ij} = \frac{A_{ij}}{\sum_{q=1:N} A_{iq}} \quad \text{and } w_i > 0. \quad (1)$$

A normalisation constraint  $\sum_i w_i = N$  is also employed, to ensure an average score of 1. Moreover, we assume that “everyone votes,” i.e., that there are no dangling nodes. We may interpret  $S$  as the transition matrix of a Markov chain that describes the behaviour of a surfer who chooses where to go next by picking one of the available outgoing links at random. Recursion (1) then characterises the score vector  $w$  as an invariant measure for this Markov chain.

**Question 1** Produce an adjacency matrix for which recursion (1) fails to converge when initialised at  $w = (1, 1, \dots, 1)$  and iterated. Assume that everyone votes.

To avoid having to enforce assumptions on the edge structure of the document collection, we may assume that the surfer occasionally gets bored following links and starts anew, selecting a random webpage from  $V$  to visit next according to some “default” distribution  $\pi$  on  $V$ . This is often referred to as *damping*. If we handle dangling nodes in a similar way, we obtain the *random surfer* model of Figure 1.

<p><b>Random Surfer</b> <math>[(V, A), \pi, d]</math></p> <p><b>At</b> <math>t = 0</math>, choose a random webpage from <math>V</math> according to <math>\pi</math>.</p> <p><b>At</b> <math>t &gt; 0</math>, <b>if there are no outgoing links</b>,              choose a random webpage from <math>V</math> according to <math>\pi</math>;</p> <p><b>else</b></p> <p>    with probability <math>d</math>                  choose an outgoing link uniformly at random among available links,</p> <p>    with probability <math>(1 - d)</math>                  choose a random webpage from <math>V</math> according to <math>\pi</math>.</p>
---

Figure 1: Description of the random surfer model for user behaviour.

**Question 2** Simulate 100 sample paths of the Markov chain of Figure 1 on the following example graph, with  $\pi$  uniform and  $d = 0.85$ :

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}. \quad (2)$$

For the  $j$ th sample path, denote the average time spent on the  $k$ th node from the beginning of that sample path until time  $t$  by  $\mu_{jt}^{(k)}$ . For a fixed sample path of your choice, and for each node, plot  $\mu_{jt}^{(k)}$  against  $t$ . For each node, and for each value of  $t$ , compute the variance of  $\mu_{jt}^{(k)}$  over different sample paths and plot it against  $t$ .

**Question 3** Modify (1) to incorporate damping and handle dangling nodes as described above. Assuming that  $1 > d > 0$ , and  $\pi_i > 0$  for all  $i$ , use standard Markov chain results to establish that the recursion you have obtained

- has a unique solution  $p$ , such that  $p$  is a distribution (i.e.,  $\sum_i p_i = 1, p_i \geq 0$ ) and  $p_i$  represents the average time the surfer spends visiting webpage  $i$ ; and
- converges to  $p$ .

The *PageRank scores* are then given by  $w = Np$ .

**Question 4** Write a procedure that implements PageRank with  $d = 0.85$  and  $\pi$  uniform. Your procedure should take as input an adjacency matrix and a maximum number of iterations, and output a column vector of PageRank scores.

- Test your procedure on  $A$  given in question 2, and compare with your results there.
- Construct an example for which node 1 has a larger number of both incoming and outgoing links than node 2, but a smaller PageRank score.

**Question 5** Write a procedure that generates a random adjacency matrix of size  $N$ , such that the out-degree of each node is an independent Poisson random variable with mean  $k$ , and conditional on the sequence of out-degrees all graphs are equiprobable.

- Generate an example with  $N = 1000$  and  $k = 100$ , and convince yourself that your implementation of PageRank is correct by inspecting the eigenvectors of the modified transition matrix. Carefully explain your reasoning. You may use the MATLAB function *eig*.

File <i>citations.dat</i> :		File <i>articlejids.dat</i> :	
...	...	...	...
9408099	9204102	9204102	82
9408099	9211097	9204103	62
9408099	9402002	9205001	65
9408099	9402005	9205002	65
...	...	...	...

Figure 2: A few entries from the two files forming the citation dataset.

- What happens to the empirical distribution of scores as  $k$  decreases?
- Describe one aspect of this model that provides an unrealistic description of real-life web networks.

### 3 Ranking academic journals.

Let us represent a collection of academic journals as a directed graph, letting  $A_{ij}$  be equal to the number of times an article published in journal  $j$  cited an article from journal  $i$ . We force  $A_{ii} = 0$ , disregarding citations within the same journal. The *Eigenfactor* (EF) score of each journal is then computed by applying PageRank to the graph described, with  $d = 0.85$  and the following choice of default distribution  $\pi$  intended to represent journal size or *popularity*:

$$\pi_i = \frac{z_i}{\sum_i z_i}, \text{ where } z_i \text{ is the number of articles in journal } i \text{ in the given time period.} \quad (3)$$

Before the introduction of the EF score, the industry standard for ranking academic journals was the *Total Citations* (TC) score, which in this representation is the in-degree of a node. To separate journal prestige from journal size or popularity, the TC score is commonly reported as an Impact Factor (IF), obtained by dividing the in-degree of node  $i$  by  $z_i$ . By analogy, the Article Influence (AI) score is obtained by dividing the EF score of a journal by  $z_i$ .

#### 3.1 Real data

The files *citations.dat* and *articlejids.dat* on the CATAM website contain citation data from the Arxiv high energy physics theory section (also see Figure 2). Each article is represented by a 7 digit identifier, leading zeros being omitted without risk of confusion. Each line of the file *citations.dat* is of the form ‘[article  $i$ ] [article  $j$ ]’, and represents a citation from article  $i$  to article  $j$ . In *articlejids.dat*, each article is assigned a *journal identifier* ranging from 1 to 272.

**Question 6** Verify that the same set of articles appears in both files, and that each article is assigned a *unique* journal identifier. Then retrieve the (multiple-edge) journal adjacency matrix  $A$ , and the vector  $z$  of articles per journal. In MATLAB, files can be loaded using the function *load*. The following functions might also be useful: *isequal*, *ismember*, *find*, *unique*.

Following the introduction of EF alongside TC scores (and of AI alongside IF scores), a debate ensued as to the relative merits of the two approaches. Central to this debate is the statistical question of whether the two offer similar information.

**Question 7** On the basis of the citations dataset, discuss the question whether EF and TC scores are practically indistinguishable. Your answer should consider

- the correlation  $\rho_{EF,TC}$  between EF and TC scores and
- the differences in journal ranking for each of the two scores.

Is the correlation  $\rho_{AI,IF}$  between AI and IF scores relevant to this question? If so, how?

## 10 Statistics

### 10.3 Bootstrap Estimation of Standard Error (5 units)

*This project is not strongly related to particular courses in Part II, though practice in statistical thinking will obviously be helpful.*

#### Background

Bootstrap methods are procedures for the empirical estimation or approximation of sampling distributions and their characteristics. Their primary use lies in estimating the accuracy (e.g., bias or variance) of parameter estimators, and in constructing confidence sets or hypothesis tests. They are applied in circumstances where the form of the population from which the observed data was drawn is unknown.

The general bootstrap method was formalized by Efron [1], [2]. In this project, the bootstrap method will be used to estimate the standard error of certain statistics derived from a sample of independent, identically distributed random variables.

Let  $\mathbf{X} = (X_1, \dots, X_n)$  be an IID sample on some sample space  $\Omega$ , drawn from a distribution  $F$ , and let  $T(\mathbf{X})$  be a statistic of interest. The standard error of  $T$  is

$$\sigma(T; F) = \sqrt{\text{Var}_F T(\mathbf{X})}.$$

The non-parametric bootstrap estimate of the standard error is

$$\sigma(T; \hat{F}) = \sqrt{\text{Var}_{\hat{F}} T(\mathbf{Y})},$$

where  $\mathbf{Y}$  is an IID sample of size  $n$  drawn from the empirical distribution

$$\hat{F}(A) = \frac{1}{n} \sum_{i=1}^n 1[X_i \in A] \quad \text{for } A \subset \Omega.$$

**Question 1** Show that  $\mathbf{Y}$  is the same as a random sample of size  $n$ , drawn with replacement from the actual sample  $\mathbf{X}$ . Comment on the reasonableness of the bootstrap estimate, such as its bias, ergodic variance, etc.

Often there will be no simple expression for  $\sigma(T; \hat{F})$ . It is, however, simple to estimate it numerically by means of simulation. The algorithm proceeds in three steps:

1. Draw a large number  $B$  of independent bootstrap samples  $\mathbf{Y}_1, \dots, \mathbf{Y}_B$ .
2. For each bootstrap sample, evaluate the statistic  $T(\mathbf{Y}_b)$ .
3. Calculate the sample standard deviation of the  $T(\mathbf{Y}_b)$  values:

$$\hat{\sigma}_B = \sqrt{\frac{1}{B-1} \sum_{b=1}^B (T(\mathbf{Y}_b) - \bar{T})^2}, \quad \text{where} \quad \bar{T} = \frac{1}{B} \sum_{b=1}^B T(\mathbf{Y}_b).$$

As  $B \rightarrow \infty$ ,  $\hat{\sigma}_B$  will approach  $\sigma(T; \hat{F})$ .

## 1 Correlation Coefficient

Suppose each sample point  $X_i$  consists of a pair  $X_i = (Y_i, Z_i)$ . The *variance-stabilized correlation coefficient* is

$$T(\mathbf{X}) = \frac{1}{2} \log \left( \frac{1 + r(\mathbf{X})}{1 - r(\mathbf{X})} \right),$$

where

$$r(\mathbf{X}) = \frac{\sum_{i=1}^n (Y_i - \bar{Y})(Z_i - \bar{Z})}{\sqrt{\sum_{i=1}^n (Y_i - \bar{Y})^2} \sqrt{\sum_{i=1}^n (Z_i - \bar{Z})^2}},$$

and  $\bar{Y} = n^{-1} \sum Y_i$  and  $\bar{Z} = n^{-1} \sum Z_i$ .

The data set `II-10-3-2020.csv` on the CATAM website contains IQ data from 120 people. Each data point  $X_i$  consists of two statistics  $X_i = (\text{VIQ}_i, \text{PIQ}_i)$ . The first measures the verbal IQ score (based on verbal questions and verbal responses); the second measures the performance IQ score (based on picture arrangement, object assembly and other nonverbal tasks).

**Question 2** Use the bootstrap method to estimate the distribution of  $T$ , i.e., plot a histogram of the bootstrap values  $T(\mathbf{Y}_b)$ . Comment on any interesting features.

**Question 3** Use the bootstrap method to estimate  $\sigma(T; F)$  by finding  $\hat{\sigma}_B$  for a reasonable value of  $B$ . Repeat this experiment several times, for the same value of  $B$ , and plot a histogram of the values of  $\hat{\sigma}_B$  you obtain. How does this histogram change with  $B$ ? What value of  $B$  would you advise?

**Question 4** Theory tells that for a bivariate normal distribution  $F$ , the standard error of  $T$  equates to

$$\sigma(T; F) = \frac{1}{\sqrt{n-3}}.$$

It has been suggested that IQ scores are normally distributed. Does your analysis provide sufficient evidence to reject the hypothesis that the Verbal and Performance IQ data are bivariate normal?

## 2 Uniform Data

Let  $\mathbf{X} = (X_1, \dots, X_n)$  be a sample of real-valued random variables. Suppose the distribution of the statistic

$$T(\mathbf{X}) = \max \{X_1, \dots, X_n\}$$

is of interest. For this very simple statistic, it is possible to calculate  $\sigma(T; \hat{F})$  exactly.

**Question 5** Calculate  $\sigma(T; \hat{F})$ . (You may wish to assess your answer by comparing it to what you obtain using the bootstrap algorithm, for some sample  $\mathbf{X}$ , but you do not need to include any such tests in your final report.)

Suppose the sample  $\mathbf{X}$  comes from the uniform distribution on  $[0, \theta]$  for some  $\theta > 0$ . (Then  $T(\mathbf{X})$  is the maximum likelihood estimator for  $\theta$ .)

**Question 6** Calculate  $\sigma(T; F)$ .

**Question 7** Generate a sample  $\mathbf{X}$  with  $\theta = 5$  and  $n = 100$ . Compare  $\sigma(T; \hat{F})$  to  $\sigma(T; F)$ . Repeat for increasing  $n$ . How well does the bootstrap method perform? Why?

## References

- [1] B. Efron,  
*Bootstrap Methods: Another Look at the Jackknife*,  
Annals of Statistics **7** (1979) 1–26.
- [2] B. Efron, R.J. Tibshirani,  
*An Introduction to the Bootstrap*,  
Chapman and Hall (1993), ISBN 0-412-44980-3.
- [3] A.C. Davison, D.V. Hinkley,  
*Bootstrap Methods and Their Application*  
Cambridge Series in Statistical and Probabilistic Mathematics, No 1,  
Cambridge University Press, 1997, ISBN 0521574714.

# 10 Statistics

## 10.15 Variable Selection and the Bias-Variance Tradeoff (8 units)

This project requires an understanding of the Part IB Statistics course.

### 1 Introduction

Consider the following linear model with a univariate response and  $p$  covariates:

$$Y = X\beta + \epsilon, \quad \epsilon \sim N(0, \sigma^2). \quad (1)$$

We assume throughout that all variables are zero mean. Note also that introducing an intercept is not necessary, since it can be captured by augmenting the covariate and regression vector as follows:

$$b + X\beta = \begin{pmatrix} 1 & X \end{pmatrix} \begin{pmatrix} b \\ \beta \end{pmatrix}.$$

We will denote a training dataset of  $N$  response-covariate tuples by  $\mathcal{T} = \{(y_t, x_t) \mid t = 1, \dots, N\}$ , where each  $x_t$  is a  $1 \times p$  row vector representing an observation. Alternatively, we may employ matrix notation, letting  $\mathbf{y} = (y_t)_{t=1:N}$  be a row vector and  $\mathbf{x} = (x_{ti})$  an  $N \times p$  matrix where each row corresponds to an observation. The *least squares* (LS) estimate of  $\beta$  then is the minimiser of the *residual sum of squares* (RSS) over the training set:

$$\text{RSS}(\hat{\beta}; \mathcal{T}) = \frac{1}{N} \sum_{t=1}^N (y_t - x_t \hat{\beta})^2 = \frac{1}{N} (\mathbf{y} - \mathbf{x} \hat{\beta})^T (\mathbf{y} - \mathbf{x} \hat{\beta}), \quad \text{and} \quad \hat{\beta}^{\text{LS}}(\mathcal{T}) = \underset{\hat{\beta}}{\text{argmin}} \text{RSS}(\hat{\beta}; \mathcal{T}). \quad (2)$$

Assuming  $N > p + 1$ , which we do, the LS estimator can be written in closed-form as

$$\hat{\beta}^{\text{LS}}(\mathcal{T}) = (\mathbf{x}^T \mathbf{x})^{-1} \mathbf{x}^T \mathbf{y}. \quad (3)$$

The dependence on the training set will be omitted when understood. In this project we will often refer to a subset of covariates as a *model*  $\mathcal{M} \subseteq \{1, \dots, p\}$ . The LS estimate for  $\mathcal{M}$  is computed on a reduced dataset  $\mathcal{T}^{\mathcal{M}}$ , obtained by deleting all covariates not in the model:

$$\mathcal{T}^{\mathcal{M}} = (y_t, (x_{ti})_{i \in \mathcal{M}})_{t=1:N}.$$

For computational ease, we will instead represent the LS estimate for  $\mathcal{M}$  in the  $p$ -dimensional space of the original model. We denote this representation by  $\hat{\beta}^{\mathcal{M}}$ , where

$$\hat{\beta}_j^{\mathcal{M}}(\mathcal{T}) = \begin{cases} \hat{\beta}_{\pi(j)}^{\text{LS}}(\mathcal{T}^{\mathcal{M}}) & \text{if } j \in \mathcal{M}, \\ 0 & \text{otherwise.} \end{cases}$$

Here  $\pi : \mathcal{M} \rightarrow \{1, 2, \dots, |\mathcal{M}|\}$  maps indices of covariates in the model to their respective indices in the reduced dataset  $\mathcal{T}^{\mathcal{M}}$ , so that  $\mathbf{x} \hat{\beta}^{\mathcal{M}}(\mathcal{T})$  is a (simpler) notation for  $\mathbf{x}^{\mathcal{M}} \hat{\beta}^{\text{LS}}(\mathcal{T}^{\mathcal{M}})$ .



## 2 The Bias-Variance tradeoff

**Question 1** Assume model (1). Let  $\hat{\beta}$  be an estimator of  $\beta$ . Show that the expected squared prediction error at a fixed, arbitrary location  $u$  can be decomposed as follows:

$$\mathbb{E}_{\mathcal{T}, y|u}(y - u\hat{\beta})^2 = \sigma^2 + \left(u\beta - \mathbb{E}_{\mathcal{T}}[u\hat{\beta}]\right)^2 + \text{Var}_{\mathcal{T}}[u\hat{\beta}],$$

where  $y \perp \mathcal{T}$ . The summands on the right-hand side are often referred to as the *irreducible variance*, *squared estimation bias* and *estimation variance*, respectively. Describe what effect deleting the  $p$ th covariate can have on each of these quantities for the LS estimator (i.e., switch  $\hat{\beta} = \hat{\beta}^{\text{LS}}$  with  $\hat{\beta} = \hat{\beta}^{\{1, \dots, p-1\}}$ ). You may consider the simplest non-trivial case, where  $\mathbf{x}$  is fixed such that  $\mathbf{x}^T \mathbf{x} = \mathbf{I}_p$ . It might further be useful to look at  $\beta_p = 0$ , and then at  $\beta_p \neq 0$ .

**Question 2** Consider model (1) with  $p = 10$ ,  $\sigma^2 = 1$ , and  $X \sim N(0, I_p)$ , and set

$$\beta = (-0.5, 0.45, -0.4, 0.35, -0.3, 0.25, -0.2, 0.15, -0.1, 0.05)^T.$$

Simulate a training dataset with  $N_{\text{tr}} = 30$  and a test dataset with  $N_{\text{te}} = 1000$ . Now consider

$$\mathcal{M}_1 = \{1\}, \mathcal{M}_2 = \{1, 2\}, \dots, \mathcal{M}_p = \{1, \dots, p\}.$$

Write a procedure that computes the training and test error of  $\hat{\beta}^{\mathcal{M}_j}$  for  $j = 1, \dots, p$ . Repeat the experiment 100 times and report your results in a plot of training and test RSS averaged over experiments, against model size. What happens if  $N_{\text{tr}} = 200$ , and why?

The above demonstrates an effect that holds in much greater generality, namely that suitably reducing the complexity of a model (in this instance, the number of variables involved) can improve prediction accuracy. There may also be gains in *model discovery*, *interpretability*, and, of course, *reduced observation costs*. Consequently, variable selection methods are of interest.

## 3 Variable selection methods

We consider two approaches to variable selection, *subset selection* and *shrinkage-based methods*. Subset selection methods look among all possible subsets of variables for the one that minimises some suitable estimate of prediction error. The search problem becomes infeasible for large  $p$ , and non-exhaustive greedy search methods have to be employed. Shrinkage-based variable selection methods will instead penalise the RSS by a penalty term that forces the LS regression coefficients to shrink in a manner that favours *exact zeros* in  $\hat{\beta}$ .

### 3.1 Subset selection

**Question 3 Best subsets selection.** Write a procedure *bestsubset* which takes as input a training dataset  $\mathcal{T}$  and outputs a  $p \times p$  matrix  $B$ , whose  $j$ th column contains  $\hat{\beta}^{\mathcal{M}_j}$  for the best performing model (in the sense of RSS) of size  $j$ ,  $\mathcal{M}_j$ :

$$\mathcal{M}_j(\mathcal{T}) = \underset{\mathcal{M}: \|\mathcal{M}\|=j}{\text{argmin}} \text{RSS} \left( \hat{\beta}^{\mathcal{M}}(\mathcal{T}); \mathcal{T} \right).$$

What is the size of the model space  $\{\mathcal{M} \mid \mathcal{M} \subseteq \{1, \dots, p\}\}$ ? Your procedure will handle with difficulty values of  $p$  for which the size of the search space  $\{\mathcal{M} \mid \mathcal{M} \subseteq \{1, \dots, p\}, \|\mathcal{M}\| = j\}$  exceeds  $10^5$  for any  $j \in \{1, \dots, p\}$ . What is the smallest such  $p$  (show your work)?

**Question 4 Greedy subset selection.** Write a procedure *greedysubset*, using the same input-output format as before, that incrementally builds up the model sequence  $\mathcal{M}_j$  by adding at each iteration the covariate that improves model fit the most:

$$\mathcal{M}_0 = \emptyset, \quad \mathcal{M}_{d+1}(\mathcal{T}) = \mathcal{M}_d(\mathcal{T}) \cup \left\{ l \mid l = \underset{j}{\operatorname{argmin}} \operatorname{RSS} \left( \hat{\beta}^{\mathcal{M}_d(\mathcal{T}) \cup \{j\}}(\mathcal{T}); \mathcal{T} \right) \right\}.$$

Can the fact that the family of models  $\mathcal{M}_0, \dots, \mathcal{M}_p$  is nested be used to gain in computational efficiency? Explain how, without effecting the change. Assuming that  $\mathcal{M}_j = \{1, \dots, j\}$ , you might want to consider the upper left  $j \times j$  block of  $((x^{\mathcal{M}_{j+1}})^T x^{\mathcal{M}_{j+1}})^{-1}$ .

**Question 5 Forward F-test.** Amend *greedysubset* to stop whenever the newly added variable does not significantly improve fit (at the  $p = .05$  level), using the F-statistic

$$\frac{\operatorname{RSS}(\hat{\beta}^{\mathcal{M}_d}) - \operatorname{RSS}(\hat{\beta}^{\mathcal{M}_{d+1}})}{\operatorname{RSS}(\hat{\beta}^{\mathcal{M}_{d+1}})/(N - d - 1)},$$

which you may assume follows an  $F_{1, N-d-1}$  distribution (you may use the MATLAB function *cdf*). Would this method work for best subset selection?

**Question 6** We can represent a sparse (linear regression) estimator more generally as an algorithm that takes as input a training set  $\mathcal{T}$  and outputs a sequence of  $p$  candidate regression vectors for each model size (i.e., the  $j$ th candidate  $\hat{\beta}^{(j)}(\mathcal{T})$  has precisely  $p - j$  zeros). Best and greedy subset search are special cases of this definition for which each candidate is a least squares solution, a condition we will not insist on here. We would like to select among candidates on the basis of estimated prediction error  $\hat{\text{PE}}$ :

$$\hat{\beta}^{\text{CV}}(\mathcal{T}) = \hat{\beta}^{j^*}(\mathcal{T}), \quad \text{where } j^* = \underset{j}{\operatorname{argmin}} \left\{ \hat{\text{PE}}(j, \mathcal{T}) \right\}.$$

The prediction error can be estimated using 10-fold cross-validation as

$$\hat{\text{PE}}(j, \mathcal{T}) = \frac{1}{10} \sum_{k=1}^{10} \operatorname{RSS} \left( \hat{\beta}^{(j)}(\mathcal{T}^{-k}); \mathcal{T}^k \right),$$

where  $\mathcal{T}^k$  is the  $k$ th fold of the training set and  $\mathcal{T}^{-k}$  its complement:

$$\begin{aligned} \mathcal{T}^k &= \left\{ (y_{\pi(n)}, x_{\pi(n)}) \mid k - 1 < \frac{10n}{N} \leq k \right\}, \\ \mathcal{T}^{-k} &= \left\{ (y_{\pi(n)}, x_{\pi(n)}) \mid \frac{10n}{N} \leq k - 1 \text{ or } \frac{10n}{N} > k \right\}, \end{aligned} \quad (4)$$

where  $\pi$  is a random permutation of  $\{1, \dots, N\}$  (you may use the MATLAB function *randperm*). Implementing the above for an arbitrary sparse estimator would involve a function taking another function as an argument. In MATLAB, this can be achieved using *function handles*, as demonstrated by *handle\_demo.m* and *testerror.m* available from the CATAM website. Write a procedure *crossval* that implements the above (the MATLAB functions *ismember* and *find* might be useful in this). This procedure should take as input  $\mathcal{T}$  and a sparse estimator, and output  $\hat{\beta}^{\text{CV}}(\mathcal{T})$ .

### 3.2 The Lasso estimator

The Lasso estimator penalises the RSS by the  $L_1$  norm of the regression coefficients:

$$\hat{\beta}^{(L,\lambda)}(\mathcal{T}) = \underset{\hat{\beta}}{\operatorname{argmin}} \left\{ \operatorname{RSS}(\hat{\beta}; \mathcal{T}) + \lambda \sum_{j=1}^p |\hat{\beta}_j| \right\} \quad (5)$$

**Question 7** Express the Lasso as a quadratic program with linear constraints.

In the Lasso estimator, the degree of sparsity is controlled *indirectly* via the penalty weight  $\lambda$ , rather than directly as in earlier methods. For  $\lambda = 0$  the full model is employed, whereas increasingly many covariates are deleted from the model as  $\lambda \rightarrow \infty$ . Given an algorithm for solving (5), we can then use cross-validation to select among any finite set of values  $\lambda_1 < \lambda_2 < \dots < \lambda_q$  for  $\lambda$ . For simplicity, we will continue here to perform cross-validation to select model size rather than  $\lambda$ . To do so, we will rely on the LARS algorithm, which, subject to certain minor assumptions and modifications that do not concern us here, allows us to compute in an efficient manner one Lasso solution for each model size. The file *monotonic\_lars.m* available from the CATAM website contains an implementation of this modified LARS algorithm that can be used as input to *crossval*.

**Question 8** The file *prostate.dat* available from the CATAM website contains a prostate cancer dataset.\* The dataset has been preprocessed to standardise the covariates and make all variables zero mean, so that you can avoid using an intercept. Column 1 contains the response, *lpsa*, and columns 2 to 9 the covariates *lcavol*, *lweight*, *age*, *lbph*, *svi*, *lcp*, *gleason*, and *pgg45*. Augment the dataset by adding four zero-mean, unit-variance covariates sampled from a distribution of your liking *independently of variables in prostate.dat*. Separate the data into a training dataset of size 70 and a test dataset of size 27. Perform a variable selection analysis of the data using the tools developed above. Present and discuss your results.

---

\*reproduced from <http://www-stat.stanford.edu/~tibs/ElemStatLearn/>

# 11 Statistical Physics

## 11.2 Monte Carlo simulation of the Ising Model (10 units)

*Knowledge of the Statistical Physics course will be useful for this project.*

### 1 Introduction

The Ising model is a simple model for a magnet. It consists of  $N$  “spins”, which are variables  $\sigma_1, \sigma_2, \dots, \sigma_N$ . Each spin (for example  $\sigma_i$ ) takes values in  $\{-1, +1\}$ . A *configuration* is specified by giving the value of every spin; we denote such a configuration by  $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \dots, \sigma_N)$ .

To each configuration  $\boldsymbol{\sigma}$  we associate an energy  $E(\boldsymbol{\sigma})$  which for the one-dimensional (1d) model is

$$E(\boldsymbol{\sigma}) = -J \left[ \sigma_N \sigma_1 + \sum_{i=1}^{N-1} \sigma_i \sigma_{i+1} \right] - h \sum_{i=1}^N \sigma_i.$$

The first term in the square brackets is present because we have arranged the spins on a circle so the  $N$ th spin is adjacent to the first one (“periodic boundary conditions”). In the canonical ensemble at temperature  $T$ , configuration  $\boldsymbol{\sigma}$  occurs with a probability given by the Boltzmann distribution

$$p(\boldsymbol{\sigma}|T) = \frac{1}{Z} \exp\left(-\frac{E(\boldsymbol{\sigma})}{T}\right)$$

where  $Z$  is the partition function (normalisation constant).

A common aim in statistical mechanics is to compute averages of observable quantities with respect to  $p$ . For example we might compute the average energy  $U(T) = \sum_{\boldsymbol{\sigma}} E(\boldsymbol{\sigma})p(\boldsymbol{\sigma}|T)$ , where the sum runs over all possible configurations. For the 1d Ising model this sum can be done analytically. In two dimensions, the average energy can be computed analytically for the special case  $h = 0$ , but not otherwise. In three dimensions, the average energy cannot be computed analytically.

In this project, we consider a general numerical method for computing averages in the canonical ensemble. (In principle this method can be used in any dimension, here we consider  $d = 1$ .) The idea is to define a dynamical process by which the Ising model evolves, as a function of time. This process is random – within its steady state, configuration  $\boldsymbol{\sigma}$  appears with probability  $p(\boldsymbol{\sigma}|T)$ . Hence one can use the random process to estimate averages such as  $U(T)$ . This method is very useful in systems that cannot be solved analytically. It is related to Markov chain Monte Carlo (MCMC) methods in statistics.

#### 1.1 Theoretical analysis

The project focusses on the magnetisation of the system. The results in this section can be derived using methods from part II statistical physics. The magnetisation of a configuration is

$$\hat{M}(\boldsymbol{\sigma}) = \frac{1}{N} \sum_i \sigma_i$$

Its average (with respect to the Boltzmann distribution) is

$$M(T) = \sum_{\boldsymbol{\sigma}} \hat{M}(\boldsymbol{\sigma})p(\boldsymbol{\sigma}|T)$$

where the sum runs over all configurations. Note that  $M(T)$  depends on  $J, h, N$  as well as  $T$ . The (scaled) magnetic susceptibility is  $\chi(T) = T \frac{\partial M}{\partial h}$ . This quantity is related to the variance of  $\hat{M}$ , as

$$\chi(T) = N \sum_{\sigma} [\hat{M}(\sigma) - M(T)]^2 p(\sigma|T)$$

In the limit  $N \rightarrow \infty$ , it can be shown by transfer matrix methods that

$$M(T) = \sinh(h/T) \frac{\cosh(h/T) + \sqrt{\sinh^2(h/T) + e^{-4J/T}}}{\sinh^2(h/T) + e^{-4J/T} + \cosh(h/T) \sqrt{\sinh^2(h/T) + e^{-4J/T}}}. \quad (1)$$

For  $h = 0$  we have also

$$\chi(T) = e^{2J/T}. \quad (2)$$

## 2 The Simulation

In the following, you will use a computer to generate a sequence of configurations, which correspond to the Ising model evolving (stochastically) as a function of time. The time  $t$  is a positive integer and the  $t$ -th configuration is denoted by  $\sigma(t)$ .

To generate the configuration  $\sigma(t+1)$ , first set  $\sigma(t+1) = \sigma(t)$ ; then choose a random integer  $i$  between 1 and  $N$ ; and finally replace  $\sigma_i(t+1)$  by either  $-1$  or  $1$ , according to

$$\begin{aligned} \text{Prob}[\sigma_i(t+1) = +1] &= \frac{1}{1 + e^{-2\kappa_i/T}} \\ \text{Prob}[\sigma_i(t+1) = -1] &= \frac{1}{1 + e^{2\kappa_i/T}} \end{aligned} \quad (3)$$

with

$$\kappa_i = h + J(\sigma_{i+1} + \sigma_{i-1}).$$

This update yields the new configuration  $\sigma(t+1)$ , which may be identical to  $\sigma(t)$ , or it may differ in exactly one spin. (It is easy to verify that the probabilities of the two possible values of  $\sigma_i(t+1)$  sum to unity. In the formula for  $\kappa_i$  then  $\sigma_{N+1}$  should be interpreted as  $\sigma_1$  and  $\sigma_0$  should be interpreted as  $\sigma_N$ , to take care of the periodic boundaries.) The variable  $\kappa_i$  is sometimes called the “local field”: you can check that the difference in energy between configurations with  $\sigma_i = -1$  and  $\sigma_i = 1$  is  $2\kappa_i$ .

After many updates, the probability that the system ends in configuration  $\sigma$  converges to  $p(\sigma|T)$ , this is discussed in a later section. It is useful to define a rescaled time  $\tau = t/N$ . If  $\tau = 1$ , this means that  $N$  updates have been performed.

### 2.1 The program

You will write a program that stores a configuration of  $N$  Ising spins, and performs random updates using the rule described above. The configuration should be stored as a sequence of 1’s and  $-1$ ’s in an array of size  $N$ . You can verify that while the parameters  $J, h, T$  all appear in the algorithm, the behaviour of the system only depends on  $J/T$  and  $h/T$ . These parameters will have the same value for every update (they are independent of time). In order to do the updates, you will need to generate random numbers: the MATLAB manual has some information on this.

Here is an outline of the program that you should write. It depends on a few parameters whose values will be discussed at the end

- (1) Choose an initial condition with  $\sigma_i = 1$  for all  $i$ .
- (2) Define a parameter  $n_w$  and perform  $n_w \times N$  updates to “warm up” the system. [The initial configuration was our choice, but the idea is that if  $n_w$  is reasonably large then we the final configuration will be distributed (approximately) as  $p(\boldsymbol{\sigma}|T)$ .] After this warming up, the rescaled time is  $\tau = n_w$ . Set a counter  $k = 1$  and define a parameter  $K$ .
- (3) For the configuration obtained at the current time  $t$ , compute and store the magnetisation as

$$M_k = \frac{1}{N} \sum_i \sigma_i(t),$$

- (4) Continue updating for an additional rescaled time  $n_I$  (that is, perform  $n_I \times N$  updates). Increase the counter  $k$  by 1.
- (5) Repeat steps (3) and (4) until  $k = K$ . At the end of this procedure you will have stored a list of  $K$  values of the magnetisation, which are  $M_1, M_2, \dots, M_K$ . It is suggested that you store these in an array of size  $K$ .
- (6) Compute the average of your magnetisation values, and also a scaled measure of their variance, as

$$\begin{aligned} \overline{M} &= \frac{1}{K} \sum_{k=1}^K M_k, \\ \hat{\chi} &= \frac{N}{K} \sum_{k=1}^K (M_k - \overline{M})^2 \end{aligned}$$

The algorithm is designed in such a way that if  $K$  and  $n_w$  are large enough, we expect  $\overline{M}$  and  $\hat{\chi}$  to be close to the equilibrium magnetisation and susceptibility defined in (1,2).

## 2.2 Numerical results

As a starting point, some recommended values for parameters are

$$N = 50, \quad n_w = 10^4, \quad n_I = 5, \quad K = 2048 .$$

Instead of specifying the parameter  $T$ , it may be useful to work instead with  $\beta = 1/T$ . A typical simulation with these parameters should not take more than a few minutes. In some cases it may be useful to store your lists of magnetisation values in a file, to avoid recreating many similar lists.

**Question 1** Fix  $J = 1$  and  $h = 0.1$ . Choose some values of  $\beta$  between 0 and 2.5; for each value of  $\beta$ , run a simulation and compute  $\overline{M}$  and  $\hat{\chi}$ . Plot these values on a graph. Compare your result for  $\overline{M}$  with the theoretical prediction of (1).

**Question 2** The simulations use random numbers, so if you do several different computations with the same parameters, you should get different answers for  $\overline{M}$ . Check that you do get different answers (if not, you may need to read about choosing different “seeds” for your random number generator). For each point in the graph in question 1, do several simulations, compute the standard deviation of the values of  $\overline{M}$  and  $\hat{\chi}$  between the runs, and use these to add suitable error bars to your plot. (If the error bars are too small to see then this is not a problem, but you should state this fact.)

**Question 3** Repeat the analysis of questions 1 and 2, but now for  $J = 1$  and  $h = 0$ . (That is, make plots of  $\overline{M}$  and  $\hat{\chi}$  against  $\beta$ , with error bars.) In this case you can also compare your result for  $\hat{\chi}$  with the theoretical prediction of (2). Make sure to analyse the errors in  $\overline{M}$ : even if the theoretical prediction for this quantity is trivial, your numerical estimate  $\overline{M}$  is not a trivial quantity.

When interpreting these results, you should recall that this numerical method is accurate only if  $n_w$  and  $K$  are large enough. The dependence of the results on these parameters will be discussed later, in question 6.

### 2.3 Validity of the method, and error estimates

It was claimed above that as the number of updates goes to infinity, the final configuration generated by the method will be distributed as  $p(\sigma|T)$ . This can be proven, but we don't do it here. However, the main step in the proof would be to verify the *detailed balance* property, which concerns the probability that an update generates configuration  $\sigma$  at time  $t + 1$ , given that the configuration at time  $t$  was  $\sigma'$ . If this probability is  $P(\sigma' \rightarrow \sigma)$  then the required condition is that for all pairs  $\sigma, \sigma'$  then

$$p(\sigma'|T)P(\sigma' \rightarrow \sigma) = p(\sigma|T)P(\sigma \rightarrow \sigma')$$

Roughly speaking, this says that if the configurations are already distributed as  $p(\sigma|T)$ , then the probability to go from  $\sigma'$  to  $\sigma$  is equal to the probability of making the opposite transition (from  $\sigma$  to  $\sigma'$ ).

**Question 4** Show that the detailed balance property holds for the simulation algorithm that you are using.

We now make a short theoretical analysis of the sequence of magnetisation values  $M_1, M_2, \dots, M_K$  that is generated by your program. These are sequences of random variables, in the sense that repeating the same computation with the same parameters will give a different sequence. If  $n_w$  is large enough, each of these numbers should have the same distribution, but that they are correlated random variables (they are not independent, because values that are adjacent in the sequence are likely to be similar to each other). This is particularly true if  $n_I$  is small.

**Programming Task:** To investigate this, generate a long sequence of magnetisation values (at least  $K = 16384$ ), at the state point  $h = 0$ ,  $J = 1$ ,  $\beta = 0.7$ . You need to choose sensible values of  $n_w$  and  $n_I$  so that the  $\overline{M}$  obtained from this sequence is a good estimate of  $M(T)$ .

Split this sequence into blocks (subsequences) of length  $\ell$  with (for example)  $\ell = 2, 4, 8, \dots$ . There will be  $K/\ell$  such blocks and you will compute the average magnetisation of the  $r$ -th block as

$$m(\ell, r) = \frac{1}{\ell} \sum_{k=r\ell+1}^{(r+1)\ell} M_k$$

with  $r = 0, 1, 2, \dots, (K/\ell) - 1$ . Note,  $m(K, 0) = \overline{M}$ .

If the  $M_k$  were independent, standard statistical methods could be used to show that the variance of  $m(\ell, r)$  is proportional to  $1/\ell$ . If the  $M_k$  are not independent (as in your case), the variance of  $m(\ell, r)$  depends on whether typical data points within a block are strongly- or

weakly-correlated. Since you have a long time series, you can estimate the variance of  $m(\ell, r)$  as

$$S(\ell)^2 = \frac{\ell}{K} \sum_{r=0}^{(K/\ell)-1} [m(\ell, r) - \overline{M}]^2$$

and it is useful to define a rescaled variance

$$s(\ell) = \ell \cdot S(\ell)^2$$

If the correlations of the samples within a block are weak, one expects  $s(\ell)$  to depend weakly on  $\ell$ . This typically happens for longer blocks. On the other hand, if the correlations are strong, one expects  $s(\ell)$  to depend strongly on  $\ell$  (this typically happens for small blocks, in which all the values are strongly correlated with each other).

**Question 5** Using your long sequence of magnetisation results, compute  $s(\ell)$  for  $\ell = 2, 4, 8, 16, 32$  and present the results in a graph. Explain the behaviour that you observe. What happens at different state points, for example  $\beta = 1.7$ ? Why?

**Question 6** Consider the graph (with error bars) for  $\overline{M}$  from question 3. Investigate how the mean value  $\overline{M}$  and its error bar depend on  $n_w$  and  $K$ . Explain the behaviour that you observe. It may be useful to refer to your answer to question 5.

**Question 7** Discuss the physical behaviour that is occurring at high and low temperatures in the 1d Ising model. You will find it useful to explain how typical configurations depend on the system parameters. What happens if you increase (or reduce) the system size  $N$ ?

Explain how the behaviour of the system affects the computational time required to compute accurate estimates of  $M(T)$  using this simulation method.



## 12 Nonlinear Dynamics/Dynamical Systems

### 12.8 A Nonlinear Map and the Dynamics of Hydrogen Atoms in Electric Fields (7 units)

*Material in both the Part II course Dynamical Systems and the Part II course Classical Dynamics is relevant to this project.*

#### 1 Part 1

The behaviour of Rydberg atoms, i.e. atoms in highly excited electronic states, in the presence of external fields has been the subject of extensive investigations. Advances in theoretical and experimental techniques in the past few decades have made it possible to study the dynamics of hydrogen atoms with very high initial quantum number. In this limit, and because of the form of the potential, they are ideal candidates for exploring the borderline between classical and quantum mechanics.

This projects explores a simple dynamical model, used to approximate the particular case of hydrogen atoms in a monochromatic, linearly polarised electric field, which are initially prepared in a very extended state along the field direction.

This case can be modelled using classical dynamics, by the one-dimensional Hamiltonian

$$H(x, p, t) = \frac{p^2}{2} - \frac{1}{x} + \varepsilon x \sin \omega t \quad (1)$$

where  $p$  and  $x$  are momentum and position of the electron, and  $\varepsilon$  and  $\omega$  are strength and frequency of the external electric field. It is convenient to express this Hamiltonian in action-angle variables  $(I, \theta)$ :

$$H(I, \theta, t) = -\frac{1}{I^2} + \varepsilon \left[ \frac{3I^2}{2} - 2I^2 \sum_{s=1}^{\infty} \frac{J'_s(s)}{s} \cos(s\theta) \right] \sin \omega t, \quad (2)$$

where  $J'_s(z)$  denotes the derivative of an ordinary Bessel function and the term in square brackets is the Fourier expansion of  $x(I, \theta)$ .

The equations of motion are then

$$\begin{aligned} \dot{I} &= -\frac{\partial H}{\partial \theta} = \varepsilon \sin \omega t \frac{\partial x(I, \theta)}{\partial \theta} \\ \dot{\theta} &= \frac{\partial H}{\partial I} = \frac{1}{I^3} - \varepsilon \sin \omega t \frac{\partial x(I, \theta)}{\partial I}. \end{aligned} \quad (3)$$

By integrating these equations over one field period, we can obtain the solution in the form of a discrete map.

In the unperturbed case  $\varepsilon = 0$  we obtain the simple twist map

$$\begin{aligned} I_{n+1} &= I_n \\ \theta_{n+1} &= \theta_n + \frac{2\pi}{I_{n+1}^3}, \end{aligned} \quad (4)$$

while for  $\varepsilon \neq 0$  the solution has the form

$$\begin{aligned} I_{n+1} &= I_n + f(I_{n+1}, \theta_n, \varepsilon) \\ \theta_{n+1} &= \theta_n + \frac{2\pi}{I_{n+1}^3} + g(I_{n+1}, \theta_n, \varepsilon) . \end{aligned} \quad (5)$$

The functions  $f$  and  $g$  have no known analytical form. We could find an approximate solution by truncating the sum in (2), but for the purpose of this simple model we shall only require that: (a) the fixed points of the perturbed map are the same as, or very close to the fixed points of the unperturbed twist map; (b) the perturbed map is area-preserving; (c) the motion described by the perturbed map becomes ‘chaotic’ only above a critical value of the action  $I$ , which depends on the strength of the perturbation. Chaotic motion is that for which trajectories, i.e. iterated points of the map, do not lie on any invariant curve.

We shall therefore choose  $f = -\frac{\partial \mathcal{B}}{\partial \theta_n}$  and  $g = \frac{\partial \mathcal{B}}{\partial I_n}$ , with  $\mathcal{B} = \varepsilon I_{n+1}^{-2} \cos \theta_n$ . and study the map

$$\begin{aligned} I_{n+1} &= I_n + \varepsilon I_{n+1}^{-2} \sin \theta_n \\ \theta_{n+1} &= \theta_n + \frac{2\pi}{I_{n+1}^3} - 2 \frac{\log \varepsilon}{I_{n+1}^3} \varepsilon I_{n+1}^{-2} \cos \theta_n . \end{aligned} \quad (6)$$

**Question 1** Check that the fixed points of the perturbed map (6) are very close to those of the simple twist map (4).

By considering how the perturbed map (6) can be described as a transformation with a generating function of the form  $F_2(I_{n+1}, \theta_n)$ , check that it is area-preserving (why do we need this?).

Now write a program that plots trajectories of the perturbed map (6) in phase space. Calculate  $\theta_n$  modulus  $2\pi$ . Note that  $I_{n+1}$  is defined implicitly in terms of  $I_n$ , so you will need to use some root-finding method to calculate it at each step.

**Question 2** For two different values of the field parameter:  $\varepsilon = 0.0005$  and  $\varepsilon = 0.002$ , plot several phase space trajectories of the map on the same graph. In each case look for representative trajectories with  $I$  in the range  $[0.7, 2.0]$ , for example starting at  $I = 0.81$ . Can you restrict initial values of  $\theta$  to the range  $[0, \pi]$ ?

Comment on your choice of root-finding method, and the numerical error it introduces. Comment on your choice of initial conditions, and total number of iterations.

Estimate the complexity of your program.

Describe the structure of the phase space. Give an estimate, based on the observed behaviour, of the critical value of  $I$  above which chaotic regions of phase space exist.

Now iterate the map with several values of the field parameter in the range  $\varepsilon \in [0, 0.1]$

**Question 3** Plot phase space trajectories for some representative values of  $\varepsilon$ .

Describe how the phase space structure changes as a function of  $\varepsilon$ . Find how the critical value of  $I$  changes with  $\varepsilon$ . Comment on the number of iterations needed to show a good illustration of the behaviour.

## 2 Part 2

In this section we introduce a function that models a slow change of the field parameter  $\varepsilon$ , in order to explore the existence of adiabatic invariants in the non-integrable system under investigation. A similar problem is investigated in [3].

Adapt the program of Part 1 to iterate the perturbed map, but this time with a gradual ‘switch-on’ of the perturbation, using the discrete switch-on function defined by:

$$A(n, N_a) = \begin{cases} 0 & s \leq 0 \\ s^2(2 - s^2)^2 & 0 < s < 1 \\ 1 & 1 \leq s \end{cases}, \quad (7)$$

where  $N_a$  is the length (in number of steps) of the adiabatic switch, and  $s = n/N_a$ .

**Question 4** In the cases  $\varepsilon = 0.005$  and  $\varepsilon = 0.015$  repeat plots of the iterates of the maps for a range of initial conditions as in question 3, but now using the function  $A(n, N_a)$  to switch on the perturbation, with  $N_a = 100$ . Describe the effects of switching on the perturbation gradually.

Focus now on just two initial values of the action:  $I = 1.1$  and  $I = 1.236068$ ,

**Question 5** Plot results obtained with different lengths of the adiabatic switch, for example  $N_a = 10, 100, 500, 1,000, 2,000$  (or other values that you think best illustrate the dependence), and varying the number of total iterations of the map.

Describe how persistence of the invariant curves varies with the different parameters.

We can verify adiabatic invariance by using the non-adiabaticity parameter defined by Dana and Reinhardt (1987):

$$\Delta J(N_a) = \left[ \frac{1}{2\pi} \int_0^{2\pi} (I_{N_a}(\theta) - I(\theta))^2 d\theta \right]^{1/2}. \quad (8)$$

Write a programme to calculate numerically the non-adiabaticity parameter  $\Delta J(N_a)$  using a suitable numerical approximation for the integral.

**Question 6** Comment on your choice of approximation for the integration.

Show plots of  $\Delta J(N_a)$  v  $N_a$  for all the cases in Question 5.

Use the non-adiabaticity parameter to verify cases where adiabaticity holds, and where it breaks down.

Comment on the relevance of these results to quantisation of this classical system.

## References

- [1] Casati, G., Guarnieri I. and Shepelyansky D. L. “Hydrogen Atom in Monochromatic Field: Chaos and Dynamical Photonic Localization”, *IEEE Journal of Quantum Electronics*, 24, 1420, 1988.
- [2] Lichtenberg, A. J. and Lieberman M. A. *Regular and Stochastic Motion* Springer
- [3] Dana, I and Reinhardt W. P. “Adiabatic Invariance in the Standard Map”, *Physica D*, 28, 115, 1987.

## 12 Nonlinear Dynamics/Dynamical Systems

### 12.9 Differential Equations for Nonlinear Oscillators (10 units)

*Material in the Part II course Dynamical Systems is relevant to this project.*

#### Introduction

Many nonlinear differential equations arise in physical, biological and chemical contexts. Several of these describe nonlinear oscillators and exhibit interesting dynamics.

The general equation of a forced nonlinear oscillator can be written as

$$\ddot{x} + (\alpha + \beta x^m)\dot{x} - \gamma x + \delta x^n = f(t)$$

When  $\beta = 0$  and  $n = 3$  the system described by this equation is known as a Duffing oscillator. When  $\delta = 0$  and  $m = 2$  it is known as a van Der Pol oscillator.

#### Part 1

We shall study the forced Duffing equation with periodic forcing in the form

$$\ddot{x} + a\dot{x} - x + x^3 = b \cos t$$

where  $a$  and  $b$  are constants and dot signifies differentiation with respect to  $t$ .

**Question 1** Write a program (using the Runge-Kutta routine for example) to integrate this system from five initial conditions with  $-2 \leq x(0) \leq 2$  and  $-2 \leq \dot{x}(0) \leq 2$ , plotting all five solutions on a single picture. (Plot  $x(t)$  against  $\dot{x}(t)$ .)

**Question 2** Test your program by running it with  $b = 0$  at  $a = -0.2$ ,  $a = 0$  and  $a = 0.2$  and show and describe the results. Comment on any special features of the case  $a = b = 0$ .

Now set  $a = 0.15$  and  $b = 0.3$ . Use your program to find two stable solutions (one is a periodic orbit, the other looks like a “strange attractor”, so solutions on this attractor never appear to settle down to any simple closed loop) which both exist at these parameter values.

**Question 3** Choose two initial conditions, one which tends towards each attractor, and adapt your program to integrate with these two sets of initial conditions only. Display your results.

**Question 4** Repeat this numerical experiment with the same two initial conditions and the same parameter values, but this time, instead of plotting the whole solution, plot points (without joining them up) only when  $t = 2n\pi$  ( $n = 0, 1, 2, \dots$ ). Comment on the relationship between the two different ways (whole trajectories and points) of representing solutions.

**Question 5** Use this program to investigate in detail the behaviour of the system at different values of  $a$  between 0.1 and 0.5 (with  $b = 0.3$  and a range of initial conditions), in particular the evolution of the strange attractor (when it exists). Show any pictures that seem interesting (four or five extra pictures are sufficient).

## Part 2

Consider the equation

$$\ddot{x} + (x^2 - b)\dot{x} - ax + x^3 = 0.$$

For  $a$  and  $b$  small the parameter space can be divided into six regions as shown in Figure 1 overleaf. A choice of  $(a, b)$  in each of these six regions yields qualitatively different behaviour of solutions. Figure 1 remains a good approximate description of the regions and boundaries for moderate values of  $a, b$ , e.g.  $|a| \leq 2, |b| \leq 2$ .

Adapt the program of Part 1 to integrate these equations at given values of  $a$  and  $b$  with solutions from five different initial conditions displayed in a single picture.

**Question 6** Run this program (with suitable choices of the initial conditions so that the pictures are as clear as possible) for a single value of  $(a, b)$  in each of the six regions, and show representative pictures for each region. You may like to set  $b = \pm 1$  and vary  $a$  to find the different regions.

**Question 7** Describe the dynamics in each region, including the nature of any fixed points or other features, and the transition from one region to the next. What happens as you move through the boundaries between each region? There is no need to find  $c$ ; you are only required to find an example of behaviour in each region.

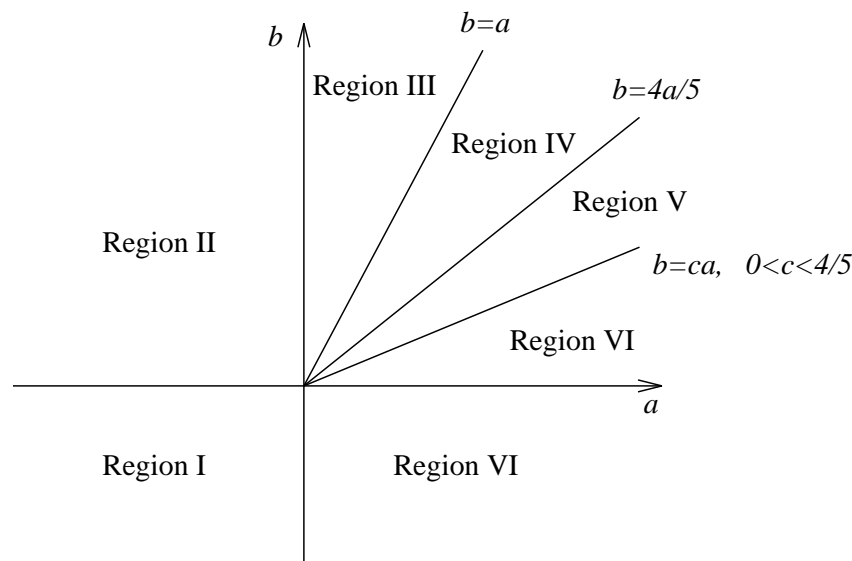


Figure 1

## Part 3

The forced van der Pol oscillator

$$\ddot{x} + a(x^2 - 1)\dot{x} + x = 1 + b$$

has a Hopf bifurcation when  $b = 0$ . (In a Hopf bifurcation, a periodic orbit appears near a stationary point as the parameter,  $b$ , increases or decreases.) The above equation may be written in Liénard coordinates as  $\dot{x} = y - a(x^3/3 - x)$ ,  $\dot{y} = -x + 1 + b$ . Adapt your program to integrate the forced van der Pol equation in this form.

**Question 8** Locate the periodic orbit when  $b = -0.001$  for  $a = 1$ ,  $a = 5$  and  $a = 10$ , giving a picture of the orbit by plotting  $x(t)$  against  $\dot{x}(t)$  at each time step.

**Question 9** Investigate the evolution of the periodic orbit for  $b \in [-0.1, 0)$  at each of these values of  $a$ , commenting on any unusual behaviour that you observe at particular values of  $b$ .

**Question 10** Consider also the appearance of the orbit in the  $x$ - $y$  plane. Can you explain its shape for large  $a$ ? Explain what numerical difficulties can arise in calculating such an orbit.

## References

- [1] Drazin, P.G. *Nonlinear Systems*. CUP
- [2] Glendinning, P. A. *Stability, Instability and Chaos*. CUP

# 14 General Relativity

## 14.1 Particle or Photon Orbits near a Black Hole (8 units)

*This project is most suitable for those attending the Part II course General Relativity.*

The aim of this project is to investigate the motion of a particle or photon in the Schwarzschild solution to the Einstein field equations.

### 1 Theory

Use the Lagrangian

$$L = F(r)\dot{t}^2 - \frac{\dot{r}^2}{F(r)} - (r\dot{\phi})^2,$$

where  $F(r) = 1 - 1/r$ , and dots denote differentiation with respect to a variable, say  $s$ , which will be the affine parameter along the geodesic. (The units are such that  $c = 1$  and  $2GM = 1$  which makes the Schwarzschild radius  $r = 1$ .)

**Question 1** Using Lagrange's equations and the Lagrangian above, obtain the equations of motion for a particle and a photon and establish equations for orbits in  $(u, \phi)$  space, where  $u = 1/r$ , using the second derivative of  $u$  with respect to  $\phi$ . You should set  $L = 0$  or  $1$  depending on whether you are using photons or particles, and use this in the differential equation if you need it.

### 2 Particle Orbits

To address the questions below you should integrate the equations using one of the Runge–Kutta routines or another of your choice.

**Question 2** A particle is at a distance corresponding to  $r = 6$ , with  $dr/d\phi = 0$  initially. For a range of angular momentum parameters from 0 to that for a circular orbit, determine

- the orbit in  $(r, \phi)$  space, giving graphical output of the orbits in this space;
- for what range of angular momentum parameters the particle is captured by the black hole (i.e., it reaches  $r = 1$ );
- the proper time taken for a particle to fall into a black hole if it does so.

**Question 3** What is the nature of the orbits if  $r = 2.5$  initially?

**Question 4** Compare your computed results with the theoretical results for some cases where analytic solutions are possible. In particular you should examine the stability of circular particle orbits at small radial distances from the black hole, and compare this with the numerical results.

**Question 5** How would the outputs change if the problem was formulated using the second derivative of  $r$  with respect to  $t$ , and the first derivative of  $\phi$  with respect to  $t$ ?

### 3 Particle Scattering

Consider a particle far from the hole with speed  $v$  and impact parameter  $b$  (the *impact parameter* is the minimum distance between the centre of the hole and the undeflected path that the particle would follow in the absence of the hole). For given  $v$  there is a critical value of  $b$ , say  $b_{crit}$ , such that particles with  $b < b_{crit}$  are captured, while particles with  $b > b_{crit}$  are not. The *cross-section*  $\sigma(v)$  is defined by  $\sigma(v) = \pi b_{crit}^2$ .

**Question 6** Devise a program to obtain  $\sigma(v)$  and to produce a graph of  $\sigma(v)$  for  $0 < v < 1$ . Comment on the limits  $v \ll 1$ ,  $v \approx 1$ .

### 4 Photon scattering

**Question 7** A photon far from the hole is directed towards it with impact parameter  $b$ . Compute the deflection angles when  $b$  is large, and compare this with results obtained analytically on the assumption that the bend angle is small.

Compute the angle of deflection of the photon for  $b = 3.2$ ,  $b = 2.8$  and  $b = 2.65$ . What happens when  $b = 2.5$ ?

### References

- [1] B. F. Schutz, *A first course in General Relativity*, CUP (1990).



## 14 General Relativity

### 14.7 Gravitational Radiation from Point Masses (8 units) in a Keplerian Orbit

*This project does not require prior knowledge of either of the Part II courses General Relativity or Cosmology, but does require Part IB Methods and Part IA Dynamics. (You may, however, find it useful to review some of your answers after taking appropriate Part II courses; but the computation may be attempted immediately.)*

For nearly Newtonian, slow-motion sources in General Relativity, the following are the formulae for the time-averaged losses due to gravitational radiation for the energy and angular momentum of a gravitating source:

$$\frac{dE}{dt} = -\frac{G}{5c^5} \langle \ddot{I}_{jk} \ddot{I}_{jk} \rangle \quad (1)$$

and

$$\frac{dJ_j}{dt} = -\frac{2G}{5c^5} \varepsilon_{jkl} \langle \ddot{I}_{ki} \ddot{I}_{il} \rangle. \quad (2)$$

Here  $i$ ,  $j$  and  $k$  are the three space indices, the summation convention is used and  $\varepsilon$  is the alternating tensor. (Note that the coordinate system used is Euclidean, so there is no distinction between upstairs and downstairs indices.) The dot represents a derivative with respect to time  $t$  (note the asymmetric time derivatives in (2)).  $I_{jk}$  is the trace-free part of the second moment of the mass distribution,

$$I_{jk} = Q_{jk} - \frac{1}{3} \delta_{jk} Q_{ii},$$

where  $Q_{jk}$  is the second moment of the mass distribution (the moment of inertia tensor):

$$Q_{jk} = \sum_a m^{(a)} x_j^{(a)} x_k^{(a)}$$

where the sum is over the (point) masses. (For continuous mass distributions the sum is replaced by an integral.)

The brackets  $\langle \bullet \rangle$  in (1) and (2) denote an average over a suitably large spacetime 4-volume (a more precise definition is given below).

Consider a Keplerian orbit of two point masses  $m_1$  and  $m_2$  in the  $x$ - $y$  plane, with semi-major axis  $a$  and eccentricity  $e$ . Assume the origin is the centre of mass. It is convenient to introduce the total mass  $M = m_1 + m_2$  and the reduced mass  $\mu = m_1 m_2 / M$ .

Let  $d$  be the distance between the two masses and let  $\psi$  be the angle of one of the masses relative to the  $x$ -axis (so that the other mass has angle  $\psi + \pi$ ) at a particular point of the orbit; so both  $d$  and  $\psi$  are functions of time. The first point mass is then at  $d_1(\cos \psi, \sin \psi, 0)$  and the second point mass is at  $d_2(-\cos \psi, -\sin \psi, 0)$ , where  $d_1 = d\mu/m_1$  and  $d_2 = d\mu/m_2$ .

**Question 1** Calculate the  $Q_{jk}$  and hence the  $I_{jk}$  as functions of  $d$ ,  $\mu$  and  $\psi$ . Simplify your answer and write it in matrix form.

For Keplerian motion the orbit equation is

$$d = \frac{a(1 - e^2)}{1 + e \cos \psi}$$

and the angular velocity is given by

$$\dot{\psi} = \frac{\sqrt{GMa(1-e^2)}}{d^2}.$$

In equations (1) and (2), assume  $\langle \bullet \rangle$  denotes an average in time over one orbit. For a function  $w$  of  $\psi$  this means that

$$\langle w \rangle = \frac{1}{T} \int_0^T w(\psi(t)) dt = \frac{1}{T} \int_0^{2\pi} \frac{w(\psi)}{\dot{\psi}} d\psi$$

where

$$T = \frac{2\pi a^{3/2}}{\sqrt{GM}}$$

is the period of the orbit.

Assume that  $a$  and  $e$  do not change appreciably in one orbit, so that their time derivatives can be ignored in the averages.

**Question 2** Note that  $J_1 = J_2 = 0$  (i.e., the  $x$  and  $y$  components of  $\mathbf{J}$  vanish). Show that

$$\left\langle \frac{dE}{dt} \right\rangle = -\frac{32G^4\mu^2M^3}{5c^5a^5} f(e) \quad (3)$$

where

$$f(e) = \frac{1 + \frac{73}{24}e^2 + \frac{37}{96}e^4}{(1-e^2)^{7/2}}$$

and that

$$\left\langle \frac{dJ_3}{dt} \right\rangle = -\frac{32G^{7/2}\mu^2M^{5/2}}{5c^5a^{7/2}} g(e) \quad (4)$$

where

$$g(e) = \frac{1 + \frac{7}{8}e^2}{(1-e^2)^2}.$$

For Keplerian orbits we have

$$E = -G\frac{\mu M}{2a}$$

and

$$J_3 = \sqrt{G\mu^2Ma(1-e^2)}$$

( $J_3$  is usually denoted  $L$ ).

**Question 3** For Keplerian orbits prove that

$$\left\langle \frac{da}{dt} \right\rangle = -\frac{64G^3\mu M^2}{5c^5a^3} f(e),$$

and that

$$\left\langle \frac{de}{dt} \right\rangle = -\frac{304G^3\mu M^2}{15c^5a^4} h(e)$$

where

$$h(e) = \frac{(1 + \frac{121}{304}e^2)e}{(1-e^2)^{5/2}}.$$

Under what conditions is it true that the time derivatives of  $a$  and  $e$  (as just calculated) can be ignored in the averages in (1) and (2) (as was assumed above in deriving (3) and (4))? (This is a consistency check.)

**Question 4** Find  $a$  as a function of time in the case when  $e \equiv 0$  (i.e., a circular orbit).

**Question 5** The WUMa eclipsing binary star system has  $m_1 = 0.77M_\odot$  and  $m_2 = 0.56M_\odot$  (the Sun has mass  $M_\odot \approx 2 \times 10^{30}$  kg), with period  $T = 0.33$  days. This determines the initial value of  $a$  but not of  $e$ , so we consider the situation as a function of  $e$ .

Using a computer, calculate how much power is being radiated (now) gravitationally for  $e = 0, 0.5$  and  $0.95$ . Compare the results with the electromagnetic output of the Sun (about  $4 \times 10^{26} \text{ J s}^{-1}$ ).

Write a program to determine how long it will take, as a function of initial eccentricity  $e_0$ , before  $a$  decays to zero for WUMa. Use this program to calculate the decay time for initial values  $e_0 = 0.00, 0.05, 0.10, \dots, 0.95$ . Compare the results with the present age of the universe ( $13.7 \times 10^9$  years).

**Question 6** Write a program to calculate the initial semi-major axis  $a_0$  that a binary must have in order for the inspiral time to be equal to the age of the universe. Use this to calculate  $a_0$  (in units of the solar radius  $R_\odot \approx 7 \times 10^8$  m) for a binary consisting of a solar mass star and giant gas planet of mass  $0.001M_\odot$  (about a Jupiter mass) with  $e_0 = 0.00, 0.05, 0.10, \dots, 0.95$ .

Hot Jupiters are a class of exoplanets with masses similar to Jupiter, but orbiting at  $< 0.5 \text{ AU}$  (where  $1 \text{ AU} \approx 1.5 \times 10^{11} \text{ m}$ ). Since Jupiter-like planets are expected to form at around  $5 \text{ AU}$ , using your above results comment on whether gravitational radiation is a possible mechanism for the Hot Jupiters to have subsequently migrated to their current location.

# 15 Number Theory

## 15.3 Positive Definite Binary Quadratic Forms (9 units)

*Background material is contained in the Part II course Number Theory. Part IB Groups, Rings and Modules and Part II Number Fields may be helpful, but are not necessary.*

### 1 Introduction

We start with a *binary quadratic form*  $f(x, y) = ax^2 + bxy + cy^2$  with  $a, b, c \in \mathbb{Z}$ , which we shall abbreviate as  $(a, b, c)$ . The *discriminant* of the form  $(a, b, c)$  is  $d = b^2 - 4ac$ . Note that  $d$  is always congruent to 0 or 1 modulo 4. We consider only *positive definite* forms, for which  $d$  is negative and  $a$  is positive.

Two forms  $f, g$  are *equivalent*, written  $f \sim g$ , if one can be transformed into the other by a *unimodular substitution*  $M$ , that is, if  $g(x, y) = fM(x, y) = f(sx + ty, ux + vy)$  where  $s, t, u, v \in \mathbb{Z}$  and  $sv - tu = 1$ , i.e.

$$M = \begin{pmatrix} s & t \\ u & v \end{pmatrix} \in \mathrm{SL}_2(\mathbb{Z}).$$

Equivalent forms have the same discriminant, but the converse is not true in general. A form  $(a, b, c)$  is *primitive* if no integer greater than one divides all three of  $a, b$  and  $c$ .

### 2 Computing the class number

A form  $(a, b, c)$  is *reduced* if either  $-a < b \leq a < c$  or  $0 \leq b \leq a = c$ . There are only finitely many reduced forms of given discriminant. It is known that distinct reduced forms are inequivalent, and that every form is equivalent to a reduced form.

**Question 1** Find bounds for the coefficients of a reduced form of given discriminant and use these to write a procedure to list all the reduced forms with given discriminant  $d$ . Find all the reduced forms of discriminant  $d$  for  $-32 \leq d < 0$ , and indicate which of these forms are primitive.

The number of equivalence classes of primitive forms of discriminant  $d$  is the *class number*  $h(d)$ . This is equal to the number of primitive reduced forms. Sometimes a slightly different definition is used, without the requirement that the forms are primitive. However you should use the definition given here.

**Question 2** Tabulate both the number of reduced forms of discriminant  $d$ , and the class number  $h(d)$ , for  $-120 \leq d < 0$ . Comment on the relationship between these numbers. Also comment on the relationship between  $h(d)$  and  $h(dk^2)$ , when  $k$  is an odd prime (you may ignore  $d = -3, -4$  here). You may find it helpful to make a table with a large enough range of  $d$  and  $k$  to look for patterns.

### 3 Reduction of positive definite forms

We can find the reduced form equivalent to a given form  $f$  by *reduction*. If  $f$  is not reduced then  $c < a$ , or  $|b| > a$ , or  $a = -b$ , or  $a = c$  and  $b < 0$ . We define operations  $S$ ,  $T$  and  $T^{-1}$  on forms by

$$\begin{aligned} S: (a, b, c) &\mapsto (c, -b, a), \\ T: (a, b, c) &\mapsto (a, b + 2a, a + b + c), \\ T^{-1}: (a, b, c) &\mapsto (a, b - 2a, a - b + c). \end{aligned}$$

These operations are represented by matrices  $S = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$  and  $T = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$  in  $\text{SL}_2(\mathbb{Z})$ , so that each operation yields an equivalent form. If a form is not reduced then one of these operations may be applied and the result will be “closer” to a reduced form (in the sense that  $|a| + |b|$  is made smaller).

**Question 3** Write a program to find the reduced form equivalent to a given form. Your program should read in the coefficients  $a$ ,  $b$ ,  $c$  and print the reduced form equivalent to  $(a, b, c)$  together with the sequence of reduction operations which are needed. Run your program on the forms  $(220, 594, 401)$  and  $(226, 367, 149)$ .

### 4 Composition of forms

The *composition* of primitive forms  $f_1 = (a_1, b_1, c_1)$  and  $f_2 = (a_2, b_2, c_2)$ , with the same discriminant  $d$ , is defined as follows. First we put  $\beta = (b_1 + b_2)/2$  and  $\gamma = (b_2 - b_1)/2$ . Then we use Euclid’s algorithm twice. The first time we compute  $m = \text{gcd}(a_1, \beta)$  and find integers  $x$  and  $y$  with  $a_1x + \beta y = m$ . The second time we compute  $n = \text{gcd}(m, a_2)$  and solve the congruence

$$(m/n)z \equiv \gamma x - c_1 y \pmod{(a_2/n)}$$

for  $z$ . The composition of  $f_1$  and  $f_2$  is then

$$f_3 = f_1 \circ f_2 = (a_1 a_2 / n^2, b_1 + 2a_1 z / n, *)$$

where the third coefficient is chosen so that  $f_3$  also has discriminant  $d$ .

**Question 4** Write a program to compute the composition of two primitive forms. Briefly explain how you solve for  $z$ . It is known that if  $f_1 \sim g_1$  and  $f_2 \sim g_2$  then  $f_1 \circ f_2 \sim g_1 \circ g_2$ . As a way of testing your program, give some examples checking that this property holds.

Let  $d$  be a discriminant, *i.e.* a negative integer that is congruent to 0 or 1 modulo 4. It is known that the set of equivalence classes of primitive binary quadratic forms of discriminant  $d$  is an abelian group under composition. This is called the *class group*. The identity class contains either  $(1, 0, -d/4)$  or  $(1, 1, (1-d)/4)$ . The inverse of the class containing  $(a, b, c)$  is the class of  $(a, -b, c)$ .

It is known that every (non-trivial) finite abelian group may uniquely be written in the form

$$C_{n_1} \times C_{n_2} \times \dots \times C_{n_t}$$

where  $n_1, \dots, n_t$  are integers greater than one with  $n_1 | n_2 | \dots | n_t$ . One way to distinguish these groups is by counting the number of elements of each given order.

**Question 5** Determine the class group for all discriminants  $d$  between 0 and  $-120$ , and in addition for  $d = -48247$ ,  $-71411$  and  $-28959$ . You are not required to write a program that works for arbitrary  $d$ , but you are expected to explain your reasoning.

## 5 An application to factoring

For the remainder of this project we will only consider primitive forms.

A form  $(a, b, c)$  is *ambiguous* if it is equivalent to  $(a, -b, c)$ .

**Question 6** Find all reduced ambiguous forms of discriminant  $d$  for  $d = -240$ ,  $-627$  and  $-1428$ . Comment on the relationship between the reduced ambiguous forms of discriminant  $d$  and the factorisation of  $d$ . What do you notice about the number of such forms?

By *factoring* we mean the task of finding a non-trivial factor of a given composite integer  $N$ . The following method uses binary quadratic forms.

We take a discriminant  $d = -kN$ , with  $k$  a small positive integer, and attempt to construct ambiguous forms of discriminant  $d$ . To do this we pick a form at random and raise it to a suitable power in the class group. If this fails to produce an ambiguous form that factors  $N$ , we repeat with another randomly chosen form. One difficulty with this method is that it seems to require knowledge of the class number  $h(d)$ .

**Question 7** Explain, in terms of complexity, why computing  $h(d)$  using the methods in Section 2 would not lead to a useful factoring algorithm.

Instead we fix a positive integer  $B$  and assume that  $h(d)$  is a product of prime powers less than  $B$ . We choose a form of discriminant  $d$  at random (for example by choosing a small value of  $a$  at random, and then solving for  $b$  and  $c$  if possible) and successively raise it to each odd prime power less than  $B$ . We then repeatedly square this form in the hope of finding an ambiguous form that factors  $N$ . If this method repeatedly fails we might increase the value of  $B$ , or change the value of  $k$ .

**Question 8** Describe an efficient procedure for computing powers in the class group, based on the programs you wrote for Questions 3 and 4.

Illustrate the above method by using it to factor  $N = 12597203$ ,  $33377419$  and  $49047121$ . You should find it sufficient to work with  $k \leq 10$  and  $B \leq 50$ . In each case you should specify both the value of  $k$  and the sequence of forms computed.

## References

- [1] Buell, D. A. *Binary quadratic forms*.
- [2] Cassels, J. W. S. *Rational quadratic forms*.
- [3] Jones, B. W. *The arithmetic theory of quadratic forms*.
- [4] Koblitz, N. *A course in number theory and cryptography*.

# 15 Number Theory

## 15.6 Computing Roots Modulo $p$ (7 units)

Background material for this project is contained in the Part II course Number Theory.

### 1 Introduction

Throughout,  $p$  will be an odd prime number. An integer  $a$  coprime to  $p$  is called a *quadratic residue* mod  $p$  if the congruence  $x^2 \equiv a \pmod{p}$  is soluble, otherwise  $a$  is termed a *non-residue* mod  $p$ . The *Legendre symbol*,  $(a/p)$ , is defined (for  $a$  any integer and  $p$  an odd prime as above) by

$$\left(\frac{a}{p}\right) = \begin{cases} 0 & \text{if } p|a; \\ 1 & \text{if } a \text{ is a quadratic residue mod } p; \\ -1 & \text{if } a \text{ is a non-residue mod } p. \end{cases}$$

In Section 2 we consider the problem of distinguishing quadratic residues from non-residues. The remainder of the project is concerned with computing square roots mod  $p$ , or more generally finding the roots of a polynomial mod  $p$ .

### 2 Computing Legendre symbols

The Legendre symbol  $(a/p)$  can be computed using Euler's criterion:

$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \pmod{p}.$$

**Question 1** Write a program to compute  $(a/p)$  for  $p$  an odd prime and  $a$  any integer, using Euler's criterion. (You should use the repeated squaring method for modular exponentiation – see [1] if this is not familiar.) Test your program with  $p = 10708729$  and

- (i) 100 random values for  $a$  between 1 and  $p$ ;
- (ii) all  $a$  between 1 and 100.

For each of (i) and (ii), keep a tally of the number of values of  $a$  for which  $(a/p) = 1$ .

The Jacobi symbol is a generalisation of the Legendre symbol. For  $n$  odd and positive it is defined by

$$\left(\frac{a}{n}\right) = \left(\frac{a}{p_1}\right) \cdots \left(\frac{a}{p_r}\right)$$

where  $n = p_1 \cdots p_r$  is a product of (not necessarily distinct) primes, and the symbols on the right are Legendre symbols. The Jacobi symbol satisfies the properties

$$\begin{aligned} \left(\frac{a}{n}\right) &= \left(\frac{a \bmod n}{n}\right), \\ \left(\frac{ab}{n}\right) &= \left(\frac{a}{n}\right) \left(\frac{b}{n}\right), \end{aligned}$$

and if  $m$  and  $n$  are odd positive and coprime,

$$\left(\frac{m}{n}\right) \left(\frac{n}{m}\right) = (-1)^{(m-1)(n-1)/4}.$$

**Question 2** Write a program to compute  $(a/p)$  for  $p$  an odd prime and  $a$  any integer, by manipulating Jacobi symbols. You should try to make your algorithm reasonably efficient. Make a note in your report of any properties of the Jacobi symbol you needed in addition to those listed above.

Estimate the complexity of your algorithm and compare with Question 1.

[You should estimate the number of basic operations required, where a basic operation could be addition or multiplication of two numbers. 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.]

### 3 Computing square roots mod $p$

Suppose that  $p$  is an odd prime and  $a$  is a quadratic residue mod  $p$ . How can we find  $x$  such that  $x^2 \equiv a \pmod{p}$ ? We could simply search through all congruence classes mod  $p$ , but if  $p$  is large then we need a better method.

**Question 3** Show that if  $p \equiv 3 \pmod{4}$  then the congruence  $x^2 \equiv a \pmod{p}$  has solution  $x \equiv a^{(p+1)/4} \pmod{p}$ . Further show that if  $p \equiv 5 \pmod{8}$  then the congruence  $x^2 \equiv a \pmod{p}$  has solution  $x \equiv 2^{k(p-1)/4} a^{(p+3)/8} \pmod{p}$  for some  $k \in \{0, 1\}$ .

Now suppose that  $p-1$  is a power of 2 and let  $g$  be a primitive root mod  $p$ , *i.e.* a generator for the multiplicative group of non-zero residues mod  $p$ . To solve the congruence  $x^2 \equiv a \pmod{p}$  we substitute  $x \equiv g^r \pmod{p}$  where  $r = \sum_{j \geq 0} r_j 2^j$  with  $r_j \in \{0, 1\}$ . Raising each side of the original congruence to suitable powers it is possible to solve for the binary digits  $r_0, r_1, r_2, \dots$  in turn. For example the first step is

$$r_0 = \begin{cases} 0 & \text{if } a^{(p-1)/4} \equiv 1 \pmod{p} \\ 1 & \text{if } a^{(p-1)/4} \equiv -1 \pmod{p}. \end{cases}$$

**Question 4** Use this method to solve the congruence  $x^2 \equiv 58256 \pmod{65537}$ .

A general algorithm for computing square roots mod  $p$  is obtained by combining the methods of Questions 3 and 4. We begin by writing  $p-1 = 2^\alpha s$  with  $s$  odd. Then we find a non-residue  $n$ , and compute  $b \equiv n^s \pmod{p}$ . Since  $s$  is odd it suffices to solve the congruence  $y^2 \equiv a^s \pmod{p}$ . We do this by substituting  $y \equiv b^r \pmod{p}$  and solving for the binary digits of  $r$ .

**Question 5** Write a program for computing square roots mod  $p$ , based on the method described above. Test your program for  $p = 10708729$  and all quadratic residues mod  $p$  between 1 and 20, and for a few other values of  $p$  and  $a$ . Estimate the complexity of your algorithm.

### 4 Computing roots of polynomials mod $p$

In this section we work with polynomials whose coefficients are integers mod  $p$ .



**Question 6** Write procedures to compute the quotient and remainder when we divide one polynomial by another. Use them to write a procedure to find the greatest common divisor of two polynomials. Illustrate by computing

$$\begin{aligned} \gcd(x^3 + 8x^2 + 12x + 4, x^3 + 6x^2 + 2x + 10) & \text{ with } p = 109, \\ \gcd(x^3 + 2x^2 + 6x + 8, x^3 + 11x^2 + x + 2) & \text{ with } p = 131, \\ \gcd(x^3 + 3x^2 + 7x + 1, x^3 + 3x^2 + 4x + 12) & \text{ with } p = 157. \end{aligned}$$

To compute the roots of a polynomial  $f(x)$  mod  $p$  we first compute  $\gcd(f(x), x^p - x)$ . This reduces us to the case where  $f(x)$  is a product of distinct linear factors. We then pick a small integer  $v$  at random and attempt to factor  $f(x)$  by computing  $\gcd(f(x), g(x))$  where  $g(x) = (x + v)^{(p-1)/2} - 1$ . This will be successful unless the numbers  $\alpha + v$  for  $\alpha$  running over the roots of  $f(x)$  are either all quadratic residues, or all non-residues. If unsuccessful we try another value of  $v$ .

**Question 7** Write a program for computing square roots mod  $p$ , based on the method described above. Explain how your program avoids working with polynomials of excessively large degree. Investigate how many values of  $v$  we expect to use (on average). Compare this method with that of Question 5 and comment on the theoretical behaviour for large  $p$ .

**Question 8** Modify your program to compute the roots of any polynomial mod  $p$ , and run it on the polynomials

$$\begin{aligned} f_1(x) &= x^4 + 9x^3 + 13x^2 + 2x - 9, \\ f_2(x) &= x^4 + x^3 + x^2 + x + 25, \\ f_3(x) &= x^4 + x^3 - 10x^2 - 749379x - 120288 \end{aligned}$$

with  $p = 10708729$ .

## 5 Programming

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.

## References

- [1] Koblitz, N. *A Course in Number Theory and Cryptography*, Graduate Texts in Mathematics 114, Springer, 1987.

# 16 Algebra

## 16.8 The Character Table of a Finite Group (9 units)

*This project is related to material in the Part II course Representation Theory.*

### 1 Introduction

Let  $G$  be a finite group. A representation  $(\rho, V)$  of  $G$  consists of a finite dimensional complex vector space  $V$  and a group homomorphism  $\rho : G \rightarrow \text{GL}(V)$ . If  $V$  is  $m$ -dimensional then we may identify  $\text{GL}(V)$  with  $\text{GL}(m, \mathbb{C})$ ; the group of  $m \times m$  invertible matrices over  $\mathbb{C}$ . The character of  $\rho$  is the function  $\chi : G \rightarrow \mathbb{C}$  given by  $\chi(g) = \text{tr } \rho(g)$ . It is known that representations are uniquely determined (up to equivalence) by their characters.

Let  $G$  have conjugacy classes  $C_1, \dots, C_r$ . The character table of  $G$  is the  $r \times r$  complex matrix with entries  $\chi_i(g_j)$  where  $\chi_1, \dots, \chi_r$  are the irreducible characters of  $G$  and  $g_1, \dots, g_r$  are representatives for the conjugacy classes. The character table conveys a great deal of information about the group  $G$ . For example, it can be used to decompose any given character as a sum of irreducibles, or to find the normal subgroups of  $G$ .

### 2 Permutation groups

A permutation  $\pi$  of  $X = \{1, \dots, n\}$  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  $X$  is the symmetric group  $S_n$ . A permutation group is a subgroup of  $S_n$  for some  $n$ . We specify a permutation group by giving a (usually very small) set of generating permutations  $\pi_1, \dots, \pi_t$ .

**Question 1** Write a program to compute the conjugacy classes in a permutation group. The program should output the group order, the conjugacy class sizes and a representative for each conjugacy class. Test your program for  $G = S_3$ .

The permutation groups considered in this project will have order less than 10000. So it should be feasible to store all the elements of  $G$ . However you should still try to make your program reasonably efficient. What is the complexity of your method in terms of  $|G|$ ,  $t$  and  $n$  (where for example storing a permutation is  $O(n)$  operations)?

### 3 Burnside's algorithm

A formal sum  $\sum_{g \in G} \lambda_g g$  (where  $\lambda_g \in \mathbb{C}$ ) belongs to the centre  $Z(\mathbb{C}[G])$  of the group ring  $\mathbb{C}[G]$  if and only if the function  $g \mapsto \lambda_g$  is constant on conjugacy classes. Thus  $Z(\mathbb{C}[G])$  is a complex vector space with basis  $b_1, \dots, b_r$  where  $b_i = \sum_{g \in C_i} g$ .

**Question 2** Write a program to determine the integers  $\nu_{ijk}$  such that

$$b_i b_j = \sum_{k=1}^r \nu_{ijk} b_k$$

for all  $1 \leq i, j \leq r$ . Prove that if  $N_i$  is the matrix with  $(j, k)$ -entry  $\nu_{ijk}$  then the matrices  $N_1, \dots, N_r$  pairwise commute. Illustrate for the example in Question 1.

**Question 3** Let  $(\rho, V)$  be an irreducible representation of  $G$  with character  $\chi$ . An argument using Schur's lemma shows that  $\rho(b_i) = \sum_{g \in C_i} \rho(g)$  is a scalar matrix. Determine this scalar by taking the trace, and hence show that the vector

$$\begin{pmatrix} |C_1| \chi(g_1) \\ \vdots \\ |C_r| \chi(g_r) \end{pmatrix}$$

(where  $g_i \in C_i$ ) is an eigenvector for each of the matrices  $N_1, \dots, N_r$  in Question 2. What are the corresponding eigenvalues? Prove that some linear combination of  $N_1, \dots, N_r$  has  $r$  distinct eigenvalues.

The inner product of characters  $\chi_1$  and  $\chi_2$  is  $\langle \chi_1, \chi_2 \rangle = \frac{1}{|G|} \sum_{g \in G} \chi_1(g) \overline{\chi_2(g)}$ . Computing the simultaneous eigenvectors of  $N_1, \dots, N_r$  determines each row of the character table up to a scalar multiple. The scaling of each row is uniquely determined by the requirement that for each irreducible character  $\chi$  we have  $\langle \chi, \chi \rangle = 1$  and  $\chi(1) > 0$ .

In computing simultaneous eigenvectors you should be careful to avoid problems due to rounding errors. See also the note on programming at the end of the project.

**Question 4** Write a program to compute the character table of a permutation group. Run your program on some alternating and symmetric groups, and on the following groups.

$$\begin{aligned} P_1 &= \langle (1, 2, 3, 4)(5, 6, 7, 8), (1, 5, 3, 7)(2, 8, 4, 6) \rangle, \\ P_2 &= \langle (1, 2, 3, 4)(5, 6, 7, 8), (1, 8)(2, 7)(3, 6)(4, 5) \rangle, \\ G_1 &= \langle (1, 2, 3)(4, 5, 6)(7, 8, 9)(10, 11, 12), (2, 9)(4, 11)(5, 8)(7, 13) \rangle \\ G_2 &= \langle (1, 2, 3, 4)(5, 6, 7, 8)(9, 10, 11, 12)(13, 14), (1, 3, 14, 5, 11, 7)(2, 4, 6, 10, 8, 13) \rangle, \\ G_3 &= \langle (1, 2, 3, 4, 5, 6, 7), (1, 4)(2, 3)(5, 8)(6, 7) \rangle, \\ G_4 &= \langle (1, 2, 3, 4)(5, 6, 7, 8), (3, 9, 8, 10)(4, 6, 11, 7) \rangle. \end{aligned}$$

In giving your answers you should list the characters in increasing order of dimension and the conjugacy classes in increasing order of size. You should also head each column with the size of the corresponding conjugacy class. It may help to improve the readability of your answers if you record the non-integer entries separately.

Is  $P_1$  isomorphic to  $P_2$ ?

**Question 5** There is a risk of rounding errors in your answers to Question 4. For each character that appears to take only integer values, how could you modify your program to be sure that the corresponding entries are correct?

## 4 Some applications of the character table

Let  $G \subset S_n$  be a permutation group. There is a natural action of  $G$  on  $X^{(d)}$ , the set of subsets of  $X = \{1, \dots, n\}$  of size  $d$ . Let  $\phi_d$  be the character of the corresponding permutation representation.

**Question 6** Write a program that uses the character table to decompose a given character as a sum of irreducibles. For each of the groups in Question 4 run your program on the characters  $\phi_2$  and  $\phi_3$ .

**Question 7** Which, if any, of the groups  $P_1, P_2, G_1, G_2, G_3, G_4$  are simple? For each of these groups, what is the smallest possible dimension of a faithful linear representation, and what is the smallest possible dimension of a faithful permutation representation? What is the smallest possible index of a proper subgroup?

## Programming note

Your programs for Questions 1 and 2 will need to determine whether a given permutation  $\pi$  appears in a list of permutations  $\pi_1, \dots, \pi_k$ . You may find that comparing  $\pi$  to each  $\pi_i$  in turn is unreasonably slow. One alternative is to use a hash function, i.e. make some ad hoc choice of function  $h$  from permutations to integers that is not too far from being injective. Then use the MATLAB function `find` to find all occurrences of  $h(\pi)$  from the list  $h(\pi_1), \dots, h(\pi_k)$ . A better method would involve sorting the permutations (e.g. by hash value) but this should not be necessary.

Depending on the method you use for Question 4, it may help to note that the MATLAB function `rref` for Gaussian elimination accepts a tolerance as its second argument.

## References

[1] L.C. Grove, *Groups and Characters*, Wiley-Interscience, 1997.

# 16 Algebra

## 16.9 Resultants and Resolvents

(8 units)

Background material is contained in the Part IB course Groups, Rings and Modules. The later questions relate to material in the Part II course Galois Theory.

### 1 Introduction

The resultant of polynomials  $f, g \in \mathbb{C}[x]$  is a polynomial in the coefficients of  $f$  and  $g$  that vanishes if and only if they have a common root. This project looks at some ways of computing the resultant and gives some applications. The final section is concerned with the Galois group of a polynomial, and for this we also need resolvents.

### 2 Resultants

The resultant of polynomials  $f(x) = a \prod_{i=1}^m (x - \alpha_i)$  and  $g(x) = b \prod_{i=1}^n (x - \beta_i)$  is

$$\text{Res}(f, g) = a^n b^m \prod_{i=1}^m \prod_{j=1}^n (\alpha_i - \beta_j) = a^n \prod_{i=1}^m g(\alpha_i).$$

**Question 1** Write a procedure for computing the resultant of two polynomials in  $\mathbb{C}[x]$ . (You may use any inbuilt procedure to compute the roots.) Test it on some polynomials with small integer coefficients and comment on the results.

The Sylvester matrix of  $f(x) = a_m x^m + a_{m-1} x^{m-1} + \dots + a_0$  and  $g(x) = b_n x^n + b_{n-1} x^{n-1} + \dots + b_0$  is the  $(m+n) \times (m+n)$  matrix

$$\begin{pmatrix} a_m & a_{m-1} & \dots & a_1 & a_0 & 0 & \dots & 0 \\ 0 & a_m & a_{m-1} & \dots & a_1 & a_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & a_m & a_{m-1} & \dots & a_1 & a_0 \\ b_n & b_{n-1} & \dots & b_1 & b_0 & 0 & \dots & 0 \\ 0 & b_n & b_{n-1} & \dots & b_1 & b_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b_n & b_{n-1} & \dots & b_1 & b_0 \end{pmatrix}$$

where the coefficients of  $f$  are repeated on  $n$  rows, and the coefficients of  $g$  are repeated on  $m$  rows.

**Question 2** Show that  $f$  and  $g$  have a common root if and only if the Sylvester matrix is singular. (Hint: Consider dependence relations between the polynomials  $f, xf, \dots, x^{n-1}f$  and  $g, xg, \dots, x^{m-1}g$ .) Compute the determinant of the Sylvester matrix for the examples in Question 1 and comment on the results.

We write  $\partial f$  for the degree of  $f$ . The resultant has the following properties.

$$\begin{aligned} \operatorname{Res}(f, g) &= (-1)^{\partial f \partial g} \operatorname{Res}(g, f) && \text{for } f, g \in \mathbb{C}[x], \\ \operatorname{Res}(\lambda f, \mu g) &= \lambda^{\partial g} \mu^{\partial f} \operatorname{Res}(f, g), && \text{for } f, g \in \mathbb{C}[x] \text{ and } \lambda, \mu \in \mathbb{C}, \\ \operatorname{Res}(f, gh) &= \operatorname{Res}(f, g) \operatorname{Res}(f, h) && \text{for } f, g, h \in \mathbb{C}[x], \\ \operatorname{Res}(f, g) &= \operatorname{Res}(f, g + hf) && \text{for } f, g, h \in \mathbb{C}[x] \text{ with } f \text{ monic.} \end{aligned}$$

**Question 3** Write a procedure that given polynomials  $f, g \in \mathbb{Z}[x]$  (with  $g$  non-zero) computes  $0 \neq c \in \mathbb{Z}$  and  $q, r \in \mathbb{Z}[x]$  with  $cf = qg + r$  and  $\partial r < \partial g$ . Then write a recursive procedure for computing the resultant of two polynomials in  $\mathbb{Z}[x]$  using only integer arithmetic. Briefly describe how your program works.

**Question 4** Show that for all but finitely many primes  $p$ , the following pairs of polynomials are coprime mod  $p$ , i.e. the polynomials obtained by reducing each coefficient mod  $p$  are coprime in  $\mathbb{F}_p[x]$ .

- $f_1(x) = x^3 - 3x^2 + 2x + 1$  and  $g_1(x) = 2x^2 - x + 1$ ,
- $f_2(x) = x^3 + 4x^2 + 5x + 13$  and  $g_2(x) = 3x^3 + 2x^2 + 4x - 9$ ,
- $f_3(x) = x^5 + x^2 - 9x + 25$  and  $g_3(x) = 2x^3 + 7x^2 + 31x + 69$ ,
- $f_4(x) = x^6 + 7x^2 + x - 3$  and  $g_4(x) = x^5 + 3x^2 + 31x + 10$ .

In each case determine the finite set of exceptional primes.

The *discriminant* of  $f(x) = \prod_{i=1}^m (x - \alpha_i)$  is  $\Delta(f) = \prod_{i < j} (\alpha_i - \alpha_j)^2$ .

**Question 5** Find a formula for the discriminant in terms of a resultant.

### 3 Solving polynomial equations

The resultant can be defined more generally for polynomials with coefficients in any ring  $R$ . For instance  $R$  could itself be a polynomial ring.

**Question 6** Write a program to compute the resultant of two polynomials with coefficients in  $\mathbb{Z}[y]$ . You should do this *either* by adapting your earlier programs *or* by using the fact that a polynomial of degree  $r$  is uniquely determined by any  $r + 1$  values. For the latter you will first need a bound on the degree of the answer as a polynomial in  $y$ .

Use your program to solve the following sets of polynomial equations.

$$\left\{ \begin{array}{l} 2x^2 - 2xy + 6x - 3y^2 + y + 4 = 0 \\ 3x^2 - 3x - 2y^2 - 6y - 4 = 0 \end{array} \right\}$$

$$\left\{ \begin{array}{l} 2x^2 + 3xy - x + 2y^2 - 2y - 4 = 0 \\ 5x^2 + 4xy + 4y^2 - 16 = 0 \end{array} \right\}$$

## 4 The Galois group of a polynomial

Let  $f \in \mathbb{Z}[x]$  be a monic polynomial of degree  $n$ , which we assume has no repeated roots. The Galois group of  $f$  is  $\text{Gal}(f) = \text{Gal}(K/\mathbb{Q})$ , where  $K$  is the splitting field of  $f$ . It acts by permuting the roots  $\alpha_1, \dots, \alpha_n$  of  $f$ , and hence is a subgroup of  $S_n$ .

Now let  $S_n$  act on the multivariate polynomial ring  $P = \mathbb{Z}[X_1, \dots, X_n]$  by permuting the indeterminates  $X_1, \dots, X_n$ . Let  $h_1, \dots, h_m$  be the orbit of some multivariate polynomial  $h \in P$  (with say  $h_1 = h$ ) under this action and write  $\text{Stab}(h) \leq S_n$  for the stabilizer. The *resolvent* of  $f$  with respect to  $h$  is the polynomial

$$R_h(f) = \prod_{i=1}^m (x - h_i(\alpha_1, \dots, \alpha_n)).$$

For example if  $f(x) = x^4 + px^2 + qx + r$  and  $h = -(X_1 + X_2)(X_3 + X_4)$  then

$$R_h(f) = x^3 + 2px^2 + (p^2 - 4r)x - q^2.$$

If  $R_h(f)$  has distinct roots then it can be shown that  $\text{Gal}(f)$  is conjugate in  $S_n$  to a subgroup of  $\text{Stab}(h)$  if and only if  $R_h(f)$  has an integer root.

**Question 7** What does this construction tell us in the cases  $h(X_1, \dots, X_n) = X_1$  and  $h(X_1, \dots, X_n) = \prod_{i < j} (X_i - X_j)$ ?

We can use floating point approximations to exactly determine a number already known to be an integer, or to prove that a number is *not* an integer. (Strictly speaking we need to bound the rounding errors. Such an analysis is not expected for this project.) However it is in general impossible to *prove* that a number is an integer using floating point approximations.

**Question 8** Write a program to compute the resolvent in the following cases. Your program should take as input a monic polynomial  $f$  with integer coefficients.

$$\begin{aligned} n = 4 & \quad h = X_1X_2 + X_3X_4, \\ n = 4 & \quad h = X_1X_2^2 + X_2X_3^2 + X_3X_4^2 + X_4X_1^2, \\ n = 5 & \quad h = \sum_{i=1}^5 X_i^2(X_{i+1}X_{i+4} + X_{i+2}X_{i+3}). \end{aligned}$$

In the third case the subscripts should be read as integers mod 5.

**Question 9** Use the programs you have written for this project to investigate the Galois groups of the following irreducible polynomials. Why can you assume that they do not have repeated roots?

$$\begin{aligned} x^4 - 7x^2 - 6x + 1, & & x^5 - x^3 - 7x^2 - x - 3, \\ x^4 - x^3 + 9x + 10, & & x^5 - x^4 + 8x^2 - 7x + 3, \\ x^4 + 2x^3 + 23x^2 + 22x + 6, & & x^5 - 2x^4 + 6x^3 - 3x^2 - x + 6. \end{aligned}$$

## 5 Programming

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 (for instance adding, multiplying, evaluating). There is no need to include the class file in your program listings, assuming you do not modify it.

You may use inbuilt functions for integer arithmetic such as `gcd` and `factor`.

In Question 3 you are asked to write your own function, and so should not use the MATLAB function `deconv`.



# 17 Combinatorics

## 17.5 Matchings

(6 units)

*This project is based on material found in the Part II course Graph Theory.*

*In this project you will need to be able to generate random  $n \times n$  bipartite graphs, that is, bipartite graphs whose each class has  $n$  vertices. The edges appear independently and at random with probability  $p$ . Sometimes it will be necessary to generate a random bipartite graph process; starting with an edgeless  $n \times n$  bipartite graph, the edges are added one by one until the complete bipartite graph is obtained, the edge to be added at any stage being chosen at random from those currently absent.*

A *matching* in a graph is a set of *independent* edges (edges with no common vertex). A *1-factor* in a graph is a 1-regular spanning subgraph, or, in other words, a matching meeting every vertex. A 1-factor in a  $n \times n$  bipartite graph is a matching of size  $n$ .

Let  $G$  be a bipartite graph with vertex classes  $X$  and  $Y$ , where  $|X| = |Y| = n$ . If  $A$  is a subset of  $X$  its *neighbourhood*  $\Gamma(A)$  is the set of vertices in  $Y$  joined to at least one member of  $A$ . A set  $A \subseteq X$  is called a *blocking set* if  $|\Gamma(A)| < |A|$ . Hall's theorem tells us that  $G$  has a 1-factor if and only if  $G$  has no blocking set. The main task in this project is to develop an algorithm which, given input  $G$ , will output either a 1-factor or a blocking set.

**Question 1** One algorithm to check whether  $G$  has a 1-factor is to check each subset  $A$  to see if it is a blocking set. Why is this a poor method?

Our algorithm to find a 1-factor will consist of finding successively larger matchings, starting with the empty matching. Let  $M$  be a matching meeting the sets of vertices  $A \subseteq X$  and  $B \subseteq Y$ . Let  $u \in X \setminus A$ . An *alternating path* is a path from  $u$  to a vertex  $v \in Y$ , such that every second edge is in  $M$ . We say that  $v$  is *reachable* from  $u$  by an alternating path; let  $V$  be the subset of  $Y$  consisting of vertices reachable from  $u$ .

**Question 2** How can a matching larger than  $M$  be found if  $V \not\subseteq B$ ?

**Question 3** How can a blocking set be found if  $V \subseteq B$ ?

The set  $V$  can be computed by initially setting  $V = \emptyset$ . Add to  $V$  the neighbours of  $u$ , then find the vertices of  $X$  matched to  $V$ , then add to  $V$  the neighbours of those vertices, and so on. For each vertex added, the preceding vertex should be recorded which led to the addition.

**Question 4** Write down clearly an algorithm to find a 1-factor based on these ideas. You should give enough detail to make it clear exactly how the computation is done, but do not give implementation details.

**Question 5** Implement your algorithm. Your program should do its computation using the adjacency matrix of the input graph, and the output should show (at least) whether a 1-factor exists or, if not, the size of a blocking set. For each of the seven values of  $p$  varying from 0.05 to 0.35 by steps of 0.05 and from  $0.1 \ln n/n$  to  $1.9 \ln n/n$  by steps of  $0.3 \ln n/n$ , run the program on twenty random  $n \times n$  bipartite graphs with  $n = 60$ . Tabulate your results, and comment on them briefly; give some attention in your comments to the sizes of the blocking sets that your algorithm finds.

**Question 6** Run your program on ten random bipartite graph processes, with  $n = 40$ . Tabulate your results. What simple properties of a graph are necessary for a 1-factor to exist? What do you notice about your results? Why do you think the above values of  $p$  were chosen?

An alternative way to store the graph in the computer is by an *adjacency list*. This is where, for each vertex in  $X$  and for each vertex in  $Y$ , a list of its neighbours is stored. Thus if  $x_4$  is joined only to  $y_1$ ,  $y_9$  and  $y_{14}$ , the list for  $x_4$  is just  $(y_1, y_9, y_{14})$ .

**Question 7** What is the (order of magnitude) complexity of your algorithm when the adjacency matrix is used? What would be the effect of using the adjacency list instead? (Do not implement a version using the adjacency list.)

**Question 8** Is the alternating path method for finding a 1-factor effective for all graphs, and not just bipartite ones? Justify your answer.

If a bipartite graph fails to have a 1-factor, it is nevertheless of interest to find a matching of the maximum possible size.

**Question 9** In what ways can your algorithm be modified to find a maximum matching in a bipartite graph?

## References

- [1] Bollobas, B., *Modern Graph Theory*, Springer, 1998.

# 17 Combinatorics

## 17.7 Graph planarity (10 units)

*This project is based on material found in the Part II course Graph Theory.*

A *planar* graph is one which can be drawn in the plane without edge crossings. Kuratowski's theorem gives a theoretical characterisation of planar graphs. However, it does not provide a practical algorithm for testing whether a graph is planar, and finding such an algorithm is not easy. This project describes one such algorithm. The focus is on simplicity rather than efficiency, and the process is broken into several separate steps. Nonetheless, the total amount of coding required for this project is still comparatively large.

Planar graphs are sparse, so graphs in the project are specified by means of an *edge list*: this is just a list of edges, each edge being a pair of vertices. We may assume that the vertices are labelled by integers. Note that the edge list does not identify isolated vertices: these don't affect planarity and you might as well assume that the number  $n$  of vertices is the largest integer label. It is often convenient to have an *adjacency list*, which is a collection of  $n$  lists, the  $j^{\text{th}}$  list giving the neighbours of vertex  $j$ . You will find it useful to write a routine to derive an adjacency list from an edge list.

Even though a planarity testing algorithm will tell whether a graph is planar, it might not supply a way to actually draw the graph in the plane. Finding such a drawing is a separate problem, and we begin with that.

### 1 Graph drawing

A beautiful theorem of Tutte gives a way to draw a 3-connected planar graph  $G$  in the plane. Let  $C$  be a cycle with exactly one bridge (as defined in the next section). If  $C$  has  $k$  vertices, place these at the corners of a convex  $k$ -gon. Place the remaining vertices so that each is at the centroid of its neighbours: that is, if  $u$  has  $d$  neighbours placed at positions  $\mathbf{x}_1, \dots, \mathbf{x}_d$  in the plane, then  $u$  has position  $\mathbf{x} = (1/d) \sum_{i=1}^d \mathbf{x}_i$ . Finding these positions requires solving a system of linear equations. Tutte proved that, under the stated conditions, there is a unique solution, and that, by adding straight line segments between adjacent vertices, we obtain a planar drawing in which each face is a convex polygon.

**Question 1** Write a program that, given a graph and a cycle  $C$ , draws the graph with  $C$  at the vertices of a regular polygon. (You need not check that  $G$  is 3-connected or that  $C$  has one bridge.)

Give your output for each of the five Platonic solids, whose edge lists can be found at [http : //www.maths.cam.ac.uk/undergrad/catam/data/Platonic\\_x.txt](http://www.maths.cam.ac.uk/undergrad/catam/data/Platonic_x.txt), where  $x$  is one of 4, 6, 8, 12 or 20 (representing the Tetrahedron, Cube, Octahedron, Dodecahedron and Icosahedron respectively). In each case you can take the vertices of the outer face  $C$  to be those labelled  $1, 2, \dots, k$  for the appropriate  $k$ .

We define the graph  $K_2$  with vertex set  $\{1, 2\}$  and single edge 1–2, and the graph  $P_5$  with vertex set  $\{3, 4, 5, 6, 7\}$  and edge set  $\{3-4, 4-5, 5-6, 6-7\}$ . From these, we define the graph  $K_2 + P_5$  with vertex set  $\{1, 2, 3, 4, 5, 6, 7\}$ , and edge set consisting of all edges of  $K_2$ , all edges of  $P_5$ , and all possible edges between a vertex of  $K_2$  and a vertex of  $P_5$  (thus  $K_2 + P_5$  is a 'complete bipartite union' of  $K_2$  and  $P_5$ ). Draw the graph  $K_2 + P_5$ , using 1, 2, 3 as the outer face.

## 2 Bridges and components

Nowadays, a bridge is usually defined to be the same as an isthmus, that is, an edge whose removal increases the number of components. Tutte used the word quite differently but we keep his terminology, though with a different meaning.

Given a graph  $G$  and a cycle  $C$  in it, a *bridge* of  $C$  is a non-empty set of edges defined as follows: it is the edges of a component of  $G[V(G) \setminus V(C)]$ , together with any edges joining that component to  $C$ . A chord of  $C$  is also defined to be a bridge of  $C$ , having a single edge. Therefore the bridges partition  $E(G) \setminus E(C)$ .

Notice that isolated vertices of  $G$  do not feature in any bridge. However, an isolated vertex of  $G[V(G) \setminus V(C)]$  which is joined to  $C$  by some edges will give rise to a bridge consisting of just those edges. Hence a bridge with one edge is either a chord of  $C$ , or it is an edge joining a vertex of  $C$  to a vertex of degree one outside  $C$ , or it is an edge joining two vertices of degree one both outside  $C$ .

The *vertices of attachment* of a bridge are the vertices of  $C$  which are end vertices of edges in the bridge. So a bridge might have no vertices of attachment (if it is the edges of a component of  $G[V(G) \setminus V(C)]$  not joined to  $C$ ), or it might have one or more.

Note that bridges can meet each other, but only at vertices of attachment.

**Question 2** Write a program to find the components of a graph. (For example, pick a vertex, find its neighbours, then their neighbours, and so on.)

Write a further program to find the bridges of a given cycle  $C$  in a graph, together with their vertices of attachment.

## 3 Interleaving

Two bridges  $B'$  and  $B''$  of the cycle  $C$  are said to *interleave* if there are four distinct vertices  $a, b, c, d \in V(C)$ , appearing in that order on the cycle (but not necessarily adjacent), such that  $a$  and  $c$  are vertices of attachment of  $B'$ , and  $b$  and  $d$  are vertices of attachment of  $B''$ . Additionally,  $B'$  and  $B''$  are also said to interleave if they both have exactly three vertices of attachment, these three vertices being the same for  $B'$  as for  $B''$ . Notice that this entire definition is symmetric in  $B', B''$ .

If  $C$  has  $\ell$  bridges  $B_1, \dots, B_\ell$ , then the *interleave graph*  $H$  has  $\ell$  vertices  $h_1, \dots, h_\ell$ , with  $h_i h_j \in E(H)$  if and only if  $B_i$  and  $B_j$  interleave.

**Question 3** Suppose  $G$  is a graph with a cycle  $C$  that has  $\ell$  bridges  $B_1, \dots, B_\ell$ . Explain why  $G$  is planar if and only if the following holds: each of the subgraphs with edges  $E(C) \cup B_i$ ,  $1 \leq i \leq \ell$ , is planar, and the interleave graph  $H$  is bipartite.

**Question 4** Write a program to construct the interleave graph from a cycle  $C$  and its bridges. Write a program to test whether a graph is bipartite.

## 4 The core of a graph

Suppose  $G$  has a vertex  $v$  of degree one. We can remove the edge at  $v$  without affecting planarity. Likewise if  $G$  has a vertex  $v$  of degree two, with edges  $uv$  and  $vw$ , we can remove these two

edges, adding the edge  $uw$  if it is not already present. This does not affect planarity either. Repeating such operations as much as possible, we arrive at the *core*  $G^*$  of  $G$ , in which each vertex has degree zero or at least three. (Possibly  $G^*$  has no edges.) The labels of the vertices in  $G^*$  can depend on the order in which the operations are done but, other than that,  $G^*$  is determined by  $G$ . Hence we need not worry about the order of the operations.

**Question 5** Write a program to find the core  $G^*$  of a graph  $G$ .

**Question 6** Describe a procedure for finding a cycle in a graph of minimum degree at least two.

Describe a procedure for finding a cycle with a chord in a graph of minimum degree at least three.

Write a program to find a cycle with a chord in a non-empty core  $G^*$ .

## 5 A planarity algorithm

Here is a recursive algorithm for testing whether a graph  $G$  is planar.

Find the core  $G^*$  of  $G$ .

If  $G^*$  is empty,  $G$  is planar.

Else find a cycle  $C$  in  $G^*$  with a chord  $e$ .

Find the bridges of  $C$  in  $G^*$  and the interleave graph  $H$ .

If  $H$  is not bipartite then  $G$  is not planar.

Else  $G$  is planar if and only if  $G^* - e$  is planar.

**Question 7** Explain why this algorithm works correctly.

**Question 8** Write a program to determine whether a graph is planar.

Test your program on various examples. You might use the graphs in Question 1, and the same graphs with one or two edges removed or added.

**Question 9** As a further test, write a program to build a random maximal planar graph with  $n$  vertices, by starting with the empty graph, and testing each of the  $\binom{n}{2}$  possible edges in a random order: if the addition of an edge maintains planarity, keep it in the graph, but if it violates planarity, throw it away. For extra interest, your program should tell you how many edges were added before the first violation.

How many edges should your graph have at the end?

Generate 30 random maximal planar graphs with 45 vertices. In each case, show how many edges were added before the first violation.

Draw one of the graphs, using your program from Question 1.

**Question 10** Estimate the complexity of the planarity algorithm from Question 8.

## References

[1] Bollobas, B., *Modern Graph Theory*, Springer, 1998.

# 19 Communication theory

## 19.2 Information content of natural language (4 units)

Background material for this project is given in the Part II course Coding and Cryptography.

Let  $I_m$  be a set of  $m$  messages which may be transmitted with non-zero probability  $p_i$ ,  $i = 1, \dots, m$ . If successive messages are independent the source is called *Bernoulli* — we do not assume this in general. Define the *source entropy* to be

$$h = - \sum_{i=1}^m p_i \log_2 p_i.$$

The *Huffman* binary code for  $I_m$  is produced by the algorithm:

- (i) Order the messages in  $I_m$  so that  $p_1 \geq p_2 \geq \dots \geq p_m$ .
- (ii) Assign **0** to be the last character of the codeword for message  $m - 1$ , and **1** for message  $m$ .
- (iii) If  $m > 2$ , combine messages  $m - 1$  and  $m$  to form a *reduced* alphabet  $I_{m-1} = \{1, 2, \dots, m - 2, (m - 1, m)\}$  with respective probabilities  $p_1, p_2, \dots, p_{m-2}$  and  $p_{m-1} + p_m$  and start again at step (i).

Whether or not a message source is Bernoulli, we can often improve the expected codeword length on a per-message basis by *segmentation*, that is, grouping messages in blocks of  $n$  and regarding them as coming from the message set  $I_m^n$ .

The files <http://www.maths.cam.ac.uk/undergrad/catam/data/II-19-2-datax.txt>, where  $x$  is one of A, B, C or D, contain samples of English texts encoded by  $A = 1, \dots, Z = 26$  with space = 0. Each file contains 401 records with 25 numbers per record, except the last, which contains a single negative number.

Choose one of the data files to work with.

**Question 1** Estimate the source entropy of English text, construct the corresponding Huffman code and find the expected codeword length. Do the same for the Shannon–Fano code and compare the two. Discuss how segmentation would improve the expected length if the source were assumed Bernoulli.

**Question 2** Discuss the extent to which English text is not Bernoulli. Construct the Huffman code for pairs of letters. What effect does segmentation have in this case? Compare the effect of segmentation on English text with its effect on a Bernoulli source with the same distribution of letter frequencies as English.

The text is derived from the Oxford Text Archive and is protected by copyright. Permission has been granted to use it for educational purposes only. Further copying of the data file is forbidden.

## References

- [1] C.M. Goldie and R.G.E. Pinch, *Communication theory*, CUP, 1991.

# 20 Probability

## 20.1 The Percolation Model

(7 units)

*This project does not presuppose attendance at any particular Part IB or Part II course.*

### 1 Introduction

The percolation process is a standard model for a random medium. Such a process possesses a singularity about which it is hard to prove much rigorous mathematics. The purpose of this project is to explore such a singularity by numerical methods.

Take as (directed) graph  $G$  the first quadrant of the square lattice, with northerly and easterly orientations. The vertices are the points  $x = (x_1, x_2)$  with  $x_1, x_2 \in \{0, 1, 2, \dots\}$ . We set  $|x - y| = |x_1 - y_1| + |x_2 - y_2|$  for two such  $x, y$ , and we join  $x$  to  $y$  by an edge  $\langle x, y \rangle$  if  $|x - y| = 1$ ; this edge is directed upwards or rightwards as appropriate.

Let  $p$  satisfy  $0 \leq p \leq 1$ . Each edge of  $G$  is designated *open* with probability  $p$ , different edges having independent designations. Edges not designated open are called *closed*. Water is supplied at the origin  $(0,0)$ , and is allowed to flow along open edges in the directions given. The problem is to study the geometry of the random set  $C$  containing all wetted points. In particular, for what values of  $p$  is there strictly positive probability that  $C$  is infinite?

Writing ' $P_p$ ' for the probability function when  $p$  is the parameter given above, we define  $\theta(p) = P_p(|C| = \infty)$ . It may be shown that  $\theta$  is a non-decreasing function, and the *critical probability* is defined by

$$p_c = \sup\{p : \theta(p) = 0\}.$$

It may be shown that  $0 < p_c < 1$  (and better bounds are known), but the true value of  $p_c$  is unknown.

You are required to investigate this percolation numerically and to comment on various aspects of the behaviour as described below. You are only expected to comment on results that you can obtain using a reasonable amount of computer time, and sensible discussions of the limitations of your methods will receive more credit than reports of excessive computations. You should, however, give some thought to how you design your programs in order to achieve larger values of  $n$  and  $m$  (defined below) than otherwise might be the case. You should comment on any such special features of your programs in *each section* of your write-up.

### 2 Estimating $\theta(p)$

One method for estimating  $\theta(p)$  is as follows. Let  $Q_n$  be the set of all points  $x = (x_1, x_2)$  with  $x_1 + x_2 = n$ . Define the sequence  $C_0, C_1, C_2, \dots$  of sets in the following inductive manner. First,  $C_0 = \{(0,0)\}$ . Having found  $C_0, C_1, \dots, C_K$ , we next define  $C_{K+1}$ . For  $y = (y_1, y_2) \in Q_{K+1}$ , we place  $y$  in  $C_{K+1}$  if and only if

$$\begin{aligned} &\text{either: } y' = (y_1 - 1, y_2) \in C_K, \text{ and } \langle y', y \rangle \text{ is open,} \\ &\text{and/or: } y'' = (y_1, y_2 - 1) \in C_K, \text{ and } \langle y'', y \rangle \text{ is open.} \end{aligned} \tag{1}$$

By generating pseudo-random numbers, we may obtain a realization of the model, and an associated sequence  $C_0, C_1, \dots$ ; for each such realization, define  $I_n$  to be 0 or 1 depending on

whether  $C_n$  is empty or non-empty (respectively). If  $I_n(1), I_n(2), \dots, I_n(m)$  are the values of  $I_n$  obtained in  $m$  independent realizations of the model, then

$$\hat{\theta}_{m,n}(p) = \frac{1}{m} \sum_{j=1}^m I_n(j)$$

may be used to estimate  $\theta_n(p) = P_p(C_n \neq \emptyset)$ . If  $n$  is sufficiently large, then  $\hat{\theta}_{m,n}(p)$  may be used to estimate  $\theta(p) = \lim_{n \rightarrow \infty} \theta_n(p)$ .

**Question 1** Give an explanation of why  $\theta$  is non-decreasing in  $p$ , that is  $\theta(p_1) \leq \theta(p_2)$  if  $p_1 \leq p_2$ . Show also that  $\theta_n(p)$  is decreasing in  $n$ . Give an estimate for the likely size of the error  $\hat{\theta}_{m,n}(p) - \theta_n(p)$ .

**Question 2** Use the scheme described above (but see the notes below) to plot  $\hat{\theta}_{m,n}(p)$  for  $p \in [0.5, 0.75]$  for suitable  $n$  and  $m$ . How would you expect a graph of the true value  $\theta(p)$  to look like in relation to your graph?

### 3 Estimating $p_c$

For fixed  $n$  and  $m$  an estimate of  $p_c$  may be obtained by finding  $\sup\{p : \hat{\theta}_{m,n}(p) = 0\}$ . Denote this estimate by  $\hat{p}_c = \hat{p}_c(m, n)$ .

**Question 3** Investigate the dependence of the estimate  $\hat{p}_c$  on the values of  $n$  and  $m$ . For fixed  $m$ , describe how the estimate varies with  $n$ , and explain why this should be so. How does the estimate vary with  $m$  for fixed  $n$ ?

### 4 Subcritical behaviour

As above, let  $C_n$  be the set of points  $x = (x_1, x_2)$  which satisfy  $x \in C$ ,  $x_1 + x_2 = n$ . When  $p < p_c$ , it may be shown that there is a constant  $\gamma > 0$  (depending on  $p$ ) for which  $P_p(C_n \neq \emptyset) \leq \exp(-\gamma n)$  and

$$\frac{1}{n} \log P_p(C_n \neq \emptyset) \rightarrow -\gamma \quad \text{as} \quad n \rightarrow \infty.$$

**Question 4** Estimate  $\gamma$  for  $p = 0.3, 0.4, 0.5$  and  $0.6$ , choosing appropriate values of  $n$  and  $m$  for each case. Describe briefly how you chose  $n$  and  $m$  in each case and why you did so. What behaviour do you expect as  $p \uparrow p_c$ ?

### 5 Notes

- (i) You may find it helpful to know that we believe  $p_c \approx 0.644$  (though you may obtain a different estimate).
- (ii) At first sight, the scheme described above appears to require  $m$  realizations of the model for each value of  $p$ . The following construction enables the same realizations to be used for all values of  $p$  simultaneously. For each edge  $e$ , we choose a pseudo-random number  $R_e$  which is uniformly distributed on  $[0, 1]$ ; different edges receive independent numbers.



To each vertex  $x$  we assign a real number  $Z(x)$  defined as follows. First,  $Z(\mathbf{0}) = 0$ , where  $\mathbf{0} = (0, 0)$  is the origin. Having calculated  $Z(x)$  for  $x \in Q_0 \cup Q_1 \cup \dots \cup Q_K$ , we define  $Z(y)$  for  $y \in Q_{K+1}$  by

$$Z(y) = \min\{A', A''\}$$

where

$$A' = \max\{Z(y'), R_{\langle y', y \rangle}\}, \quad A'' = \max\{Z(y''), R_{\langle y'', y \rangle}\},$$

where  $y'$  and  $y''$  are given in (1). For given  $p$ , we may obtain a percolation realization as follows. Call an edge  $e$  *open* if  $R_e \leq p$  (an event having probability  $p$ ). It may now be seen that the set  $\{y : Z(y) \leq p\}$  has the same distribution as the set  $C$  of wetted points given above. Much computational time may be saved by this device.

- (iii) In practice, you may find that much of the computation time is spent in generating the pseudo-random numbers. Time may be saved if they are not calculated to excessive precision.
- (iv) There are many interesting features of the super-critical behaviour ( $p > p_c$ ); in particular the ‘shape’ of the infinite cluster when it exists. You are **not** asked to comment on these features for this project.
- (v) Further details about percolation in general may be found in: *Percolation*, G R Grimmett, Springer, Berlin 1999. Further details about the particular percolation used in this project (two-dimensional oriented bond percolation) can be found in, for example, R. Durrett, *The Annals of Probability* **12**:999-1040 (1984). It is **not** necessary (nor even particularly desirable) to consult either of these references before attempting this project.

## 20 Probability

### 20.2 Importance Sampling and Fast Simulation (5 units)

This project assumes some basic knowledge of discrete-time Markov Chains, as covered in the Part IB course.

#### 1 Importance sampling

*Importance sampling* is a technique for simulating random variables. Suppose that  $X$  is a random variable with density  $\pi$ , and suppose we wish to use simulation to estimate the probability that  $X$  takes a value in some set  $A$ . If  $X$  is difficult to simulate, or if the event that  $X$  is in  $A$  is very rare, then this might be hard to do.

Suppose that we have some other random variable  $Y$  with density  $\pi'$ , such that  $\pi'(x) > 0$  whenever  $\pi(x) > 0$ . Let  $L$  be the likelihood ratio,

$$L(y) = \frac{\pi(y)}{\pi'(y)}.$$

Let  $\gamma = \mathbb{P}(X \in A)$ , and consider the estimator

$$\hat{\gamma} = L(Y)1[Y \in A]$$

where  $1[\cdot]$  is the indicator function. We call  $\pi'$  the *twisted density*.

**Question 1** Show that  $\hat{\gamma}$  is an unbiased estimator of  $\gamma$ . How could you run a simulation to estimate  $\gamma$  using this fact?

This method for estimating  $\gamma$  is called *importance sampling*. Now let  $X$  be an exponential random variable with mean 3, and consider the event  $B = \{X > 30\}$ . Suppose we wish to estimate  $\mathbb{P}(B)$  by importance sampling, using as our twisted distribution an exponential with mean  $\lambda^{-1}$ .

**Question 2** What is  $\mathbb{P}(B)$ ? Write a program to estimate this probability using importance sampling. Simulate for different values of  $\lambda$  and include in your report some typical outputs for each  $\lambda$ . (*Programming hint. If  $U$  is a uniform random variable on  $[0, 1]$ , then  $-\log U$  is an exponential random variable with mean 1.*)

You should find that some values of  $\lambda$  lead to better estimators than others.

**Question 3** Modify your program to estimate how long a simulation you need in order to obtain a good estimate, for a range of values of  $\lambda$ . Explain your method. What value of  $\lambda$  seems best? Give an intuitive reason why it is so.

**Question 4** Prove that for  $\lambda \geq \frac{2}{3}$  the simulation is useless. Calculate the optimal value of  $\lambda$ .

This shows that importance sampling isn't automatically good—it is important to choose the distribution carefully.

Of course, the probability that an exponential random variable exceeds some amount doesn't need simulating, but it is still useful to be able to do importance sampling of exponential random variables for the following reason:

The exponential distribution is often used to model the time until an event occurs. Now imagine trying to simulate, say, breakdowns in a large system. It is useful to be able to increase the rate of breakdowns while leaving the rate of other events unchanged. Importance sampling can be used to do this.

## 2 Fast simulation

Importance sampling can be applied to Markov chains, when it is often called *fast simulation*. Suppose we have a discrete-time Markov chain with jump probabilities  $P_{ij}$ . The path which the Markov chain takes can be regarded as a random variable in the space of sample paths, and we might be interested in the probability that the path is in some particular set of paths.

Consider another Markov chain with *twisted jump probabilities*  $P'_{ij}$  such that  $P'_{ij} > 0$  whenever  $P_{ij} > 0$ .

**Question 5** If we observe a path  $x_0 \rightarrow x_1 \rightarrow \dots \rightarrow x_n$ , what is the likelihood ratio  $L$  of that path?

Consider a simple random walk  $(X_n)_{n \geq 0}$  on the non-negative integers with up probability  $p < 1/2$  and down probability  $1 - p$ , except up probability 1 when the walk is at 0. We might be interested in the event that, on a single excursion away from 0, the random walk hits some high level  $C$  before it returns to 0.

**Question 6** Calculate the probability of this event, for  $p = 1/4$  and  $C = 30$ . That is, find  $\mathbb{P}(T_C < T_0 | X_0 = 0)$ , where  $T_x = \inf\{n \geq 1 : X_n = x\}$ , for these values of the parameters.

This probability is too small to simulate directly. Now consider a fast simulation using another random walk with up probability  $p' > 1/2$  and down probability  $1 - p'$ .

**Question 7** What is the probability that the twisted random walk reaches  $C$  on a single excursion away from 0? How does this change as  $C$  becomes large, compared to the random walk with up probability  $p$ ? What consequences does this have for simulation?

**Question 8** Carry out a simulation to estimate the probability referred to in Question 6. Try a range of values of  $p' > 1/2$ . Comment on your results, keeping in mind that the estimate should be unbiased.

**Question 9** In the case  $p' = 1 - p$ , calculate the exact distribution of your estimator. How close would you expect your estimator to be to the true value, after 10,000 trials?

This model might represent the behaviour of a buffer in a computer network, where we would be interested in the probability that the buffer became full and started losing messages. Again, a single buffer does not need simulating, but in a more complicated network, it may not be possible to calculate the overflow probability analytically. The sort of overflow probability which one sees in real computer networks is often of the order of  $10^{-8}$ , so there is a genuine need for fast simulation in these situations.

## 23 Astrophysics

### 23.1 Generating a Consistent Self-Gravitating System (8 units)

*The project is self-contained, though some knowledge of galactic structure may be advantageous.*

#### Introduction: distribution functions of gravitating systems

A collection of  $N$  particles moving under their mutual gravitational attraction only, can be well described by a continuous *distribution function*,  $f(\mathbf{x}, \mathbf{v}, t) d^3\mathbf{x} d^3\mathbf{v}$ , in the limits where  $N$  is large and the particles have some positions  $\mathbf{x}_i$  and velocities  $\mathbf{v}_i$ ,  $i \in \{1, \dots, N\}$ , at time  $t$ .

In the limit where two-body interactions can be neglected ( $N$  large enough), the flow in phase space is said to be *collisionless* and the distribution function satisfies the Boltzmann equation:

$$\frac{df}{dt} = 0. \quad (1)$$

The dynamics of the system are governed by the Poisson equation,

$$\nabla^2 \Phi(\mathbf{x}) = 4\pi G \rho(\mathbf{x}), \quad (2)$$

where the spatial density is  $\rho(\mathbf{x}, t) = \int f d^3\mathbf{v}$ ,  $\Phi$  is the potential and  $G$  is the gravitational constant.

#### Instructions

The aim of this project is to generate a discrete particle realisation of a spherical self-gravitating system, given a potential–density pair describing the system, and compare the results for a finite number of particles with those from continuous distributions. Here we will consider a specific example of a spherical, isotropic distribution, the so-called  $\gamma$  distribution functions ([2]). The  $\gamma$  functions have density profiles that are proportional to  $r^{-\gamma}$  as  $r \rightarrow 0$ . Here we consider  $\gamma = 0$ .

For a spherical, isotropic system,  $f$  can be expressed as a function of the specific energy,  $E$ , only. For a particle  $i$ ,  $E_i = \frac{1}{2}v_i^2 + \Phi(r_i)$ , where  $r_i = |\mathbf{x}_i|$  and  $v_i = |\mathbf{v}_i|$ .

**Question 1** The  $\gamma = 0$  model is specified by the density profile

$$\rho(r) = \frac{3Ma}{4\pi(r+a)^4}. \quad (3)$$

The associated potential is determined by solving (2). Show that in this case

$$\Phi(r) = -\frac{GM}{2a} \left[ \frac{(r+a)^2 - r^2}{(r+a)^2} \right] \quad (4)$$

where  $M$  is the total mass.

The enclosed mass within radius  $r$  is given by

$$M(r) = \int_0^r 4\pi\rho(r')r'^2 dr' = M \left( \frac{r}{r+a} \right)^3.$$

Without loss of generality, we take  $a = 1$ ,  $G = 1$ , and  $M(r \rightarrow \infty) = 1$ . Additionally define the dimensionless binding energy  $\epsilon = -E \times (a/GM)$  and potential  $\Psi = -\Phi \times (a/GM)$ .

Given  $\rho(\Phi)$ , it is straightforward to derive  $f(\epsilon)$  with an Abel transform ([1], especially page 651). For this model

$$f(\epsilon) = \frac{3}{4\sqrt{2}\pi^3} \int_0^\epsilon \frac{(1-y)^2(2y+4y^2)}{y^4\sqrt{\epsilon-\Psi}} d\Psi, \quad (5)$$

where  $y(\Psi) = \sqrt{1-2\Psi}$ .

**Question 2** Verify that

$$f(\epsilon) = \frac{3}{2\pi^3} \left[ \frac{(3-4\epsilon)\sqrt{2\epsilon}}{1-2\epsilon} - 3 \sinh^{-1} \sqrt{\frac{2\epsilon}{1-2\epsilon}} \right]. \quad (6)$$

The incremental mass of particles,  $dM$ , with binding energies in the energy interval  $\epsilon$  to  $\epsilon + d\epsilon$  is given by the differential energy distribution

$$\frac{dM}{d\epsilon} = f(\epsilon)g(\epsilon), \quad (7)$$

where the density of states  $g(\epsilon)$  ([2]) is given by

$$g(\epsilon) = 8\pi^2 \left[ \sqrt{1-2\epsilon} \frac{3-14\epsilon-8\epsilon^2}{12\epsilon^2} - \pi + \frac{1-6\epsilon+16\epsilon^2}{(2\epsilon)^{5/2}} \cos^{-1}(-\sqrt{1-2\epsilon}) \right]. \quad (8)$$

We wish to generate a realisation of our ( $\gamma = 0$ ) distribution, using a Monte-Carlo acceptance-rejection algorithm, as discussed below.

**Programming Task:** Generate an  $N = 5000$  particle realisation of the distribution function above. You should truncate the distribution at some finite radius  $r_T$  (recommended values are 100 or 300) and renormalise your particle mass after generating the realisation so that  $M = 1$ .

To do this you may find the following information useful: the maximum value of the quantity  $r^2v^2 \times f(\epsilon)$  is  $2.884 \times 10^{-3}$ ; the maximum binding energy  $\epsilon_{\max}$  is  $\frac{1}{2}$ ; and the maximum speed of a particle  $v_{\max}$  is  $\sqrt{2\epsilon_{\max}}$ .

One way to approach this problem is to draw a pair of uniform random numbers with  $r_r \in [0, r_T)$  and  $v_r \in [0, v_{\max})$ . Then compare the quantity  $(r_r/a)^2(v_r^2/(M/a))f(\epsilon(r_r, v_r))$  and its maximum value with another uniform random variable  $\xi_i$  and accept or reject your draw from phase space accordingly.

## Comparisons with analytic results

**Question 3** Compare your numerical energy distribution  $dM(\epsilon)/d\epsilon$  with the expected analytic differential energy distribution.

**Question 4** Given your set of  $N$  particles,  $r_i, v_i, i = 1, \dots, N$ , generate a uniform three-dimensional realisation of your distribution in Cartesian co-ordinates. That is, form the set  $\{x_i, y_i, z_i, v_{xi}, v_{yi}, v_{zi}\}$  by Monte-Carlo generation of a uniform distribution of Cartesian components of  $r_i, v_i$ .

What is the actual mass of your  $N$  particles and how does it compare with the mass you expected given the choice of  $r_T$ ? You may want to try different values of  $r_T$  to see how  $M(r_T)$  varies with  $r_T$ .

**Question 5** Write a short routine that sorts the particles into radius bins and generates a numerical density profile of your distribution. How does your actual density profile compare with the expected analytic density profile? How does the density profile fit change as you vary your bin size? (**Hint:** use a log–log plot.)

**Question 6** Show that for this distribution the dispersion  $\sigma^2(r) = \langle v^2 \rangle$ , where angle brackets denote the average value over the particles, is given by

$$\sigma^2(r) = \frac{GM(a + 6r)}{10(r + a)^2}. \quad (9)$$

Compare your numerical dispersion with the analytic estimate. Calculate the angular momentum  $\mathbf{L}(r, \Delta r) = \sum_{r < r_i < r + \Delta r} m_i \mathbf{x}_i \times \mathbf{v}_i$ , in radial bins. Does your distribution have any net angular momentum? Should it?

**Question 7** The anisotropy is defined as  $\beta(r) = 1 - \langle v_t^2 \rangle / 2 \langle v_r^2 \rangle$ , where  $\langle v_t^2 \rangle = \langle v^2 \rangle - \langle v_r^2 \rangle$  and  $\langle v_r^2 \rangle = \langle (\mathbf{v} \cdot \mathbf{x} / r)^2 \rangle$ . Plot the anisotropy as a function of radius. Is your distribution anisotropic? Should it be?

**Question 8** As a function of radius, what is the potential of your realisation and how does it compare with the analytic potential estimated? How does the  $\langle v^2 \rangle$  compare with the local escape speed as a function of radius? How does varying the particle number affect your results?

## References

- [1] Binney, J. and Tremaine, S. *Galactic Dynamics*, Princeton Series in Astrophysics, 1987.
- [2] Dehnen, W. *A family of potential–density pairs for spherical galaxies and bulges*, MNRAS, **265**, 250–256 (1993). Full article available online from NASA ADS, [http://ukads.nottingham.ac.uk/abstract\\_service.html](http://ukads.nottingham.ac.uk/abstract_service.html)

## 23 Astrophysics

### 23.4 Stellar Structure

(8 units)

*The project involves the structure of stars. Knowledge of the Structure and Evolution of Stars course in Part II Astrophysics is useful but all relevant equations are defined and explained in the project itself.*

#### 1 Introduction

The physics of stars can be encapsulated in a set of differential equations which can be solved with appropriate boundary conditions. The most efficient method of solving the equations is by a relaxation technique. In order to converge, it relies on an initial guess to the solution that is not far from the actual solution. In this project we construct such an initial model of a star by a shooting technique that directly integrates the equations from the boundaries, where conditions are varied, until the two solutions meet in the middle.

#### 2 The Equations of Stellar Structure

The structure of a spherically symmetric star of uniform and unchanging composition, in thermal equilibrium, can be described by four non-linear differential equations in five variables together with an equation of state and boundary conditions.

1) Hydrostatic Equilibrium,

$$\frac{dp}{dr} = -\rho \frac{Gm}{r^2}, \quad (1)$$

where  $p$  is the pressure and  $\rho$  the density at radius  $r$ , measured from the centre,  $m$  is the mass interior to  $r$  and  $G$  is Newton's gravitational constant ( $6.6726 \times 10^{-11} \text{ m}^3\text{kg}^{-1}\text{s}^{-2}$ ).

2) Mass continuity,

$$\frac{dm}{dr} = 4\pi r^2 \rho. \quad (2)$$

3) Energy generation,

$$\frac{dL_r}{dr} = 4\pi r^2 \rho \epsilon, \quad (3)$$

where  $L_r$  is the luminosity, the outward flow of energy through a sphere at radius  $r$ , and

$$\epsilon = \epsilon(\rho, T, \text{composition}) \quad (4)$$

is the energy generation rate per unit mass.

4) Energy can be transported by radiation (or equivalently conduction) or by bulk convective motions. In the radiative case

$$\frac{dT}{dr} = -\frac{3\kappa\rho L_r}{16\pi a c r^2 T^3}, \quad (5)$$

where

$$\kappa = \kappa(\rho, T, \text{composition}) \quad (6)$$

is the opacity,  $a$  is the radiation constant ( $7.5646 \times 10^{-16} \text{ J m}^{-3}\text{K}^{-4}$ ) and  $c$  the speed of light in a vacuum ( $2.9979 \times 10^8 \text{ m s}^{-1}$ ).

The equation of state relates pressure to density, temperature and composition throughout the star,

$$p = p(\rho, T, \text{composition}). \quad (7)$$

Appropriate boundary conditions at the centre are

$$m = 0, \quad L_r = 0 \quad \text{at} \quad r = 0. \quad (8)$$

At the surface,  $r = R_*$ , the radius of the photosphere, an Eddington approximation to a plane parallel grey\* atmosphere leads to

$$L_* = 4\pi R_*^2 \sigma T^4, \quad (9)$$

where  $\sigma = ac/4$  is the Stefan–Boltzmann constant,  $L_*$  is the bolometric luminosity of the star and

$$p\kappa = \frac{2}{3} \frac{GM_*}{R_*^2}, \quad (10)$$

where  $M_*$  is the stellar mass. There are four independent variables  $p(r)$ ,  $m(r)$ ,  $L_r$  and  $T(r)$ , when  $\rho(r)$  is determined by equation (7), for which a unique solution can be found.

## 2.1 Choice of variables

In practice it is better to use  $m$  rather than  $r$  as independent variable and, for uniform composition, the solution is unique for a given stellar mass. We then apply the surface boundary conditions at  $m = M_*$ .

**Question 1** Find the derivatives with respect to  $m$  of the new dependent variables,  $r^3$ ,  $L_r$ ,  $T^4$  and  $\ln p$ .

## 2.2 Equations at the centre

As they stand the equations are not suitable for numerical integration at the centre. It is therefore necessary to develop them to obtain conditions at some small but finite value of  $r$ .

**Question 2** Develop the differential equations obtained in Question 1 at the centre to obtain equations such as

$$p = p_c - \frac{2}{3} \pi G \rho_c^2 r^2 \quad (11)$$

for small  $r$ , where  $p_c$  and  $\rho_c$  are the central values. Do not forget to take account of the central boundary conditions.

## 3 The Physics of the Equation of State, Energy Generation and Opacity

For the purposes of this project we shall assume that stars are composed entirely of hydrogen (mass fraction  $X$ , assumed to be 0.7 throughout this project) and helium (mass fraction  $Y = 1 - X$ ) and that the contributions to pressure, other than that of the perfect gas, are negligible so that

$$p = \frac{\rho R^* T}{\mu}, \quad (12)$$

---

\*A grey atmosphere is one in which the opacity is independent of wavelength.



where  $R^*$  is the gas constant ( $8.3145 \times 10^3 \text{ J kg}^{-1} \text{ K}^{-1}$ ) and  $\mu$  is the mean molecular weight. It is sufficient to calculate  $\mu$  on the assumption that the material is completely ionized so that

$$\frac{1}{\mu} = 2X + \frac{3}{4}Y. \quad (13)$$

The same applies to  $\gamma$  (see question 4), the adiabatic exponent, which may be taken to be constant at  $\frac{5}{3}$  for a monatomic ideal gas. We shall further assume that the opacity,  $\kappa$ , may be approximated by the contribution from electron scattering

$$\kappa_{\text{es}} = 0.02(1 + X)\text{m}^2 \text{kg}^{-1}. \quad (14)$$

Finally we assume that nuclear burning proceeds via a combination of the proton-proton chain and CNO cycle, producing energy at a rate

$$\epsilon = \left(0.25 X^2 e^{-33.8T_6^{-1/3}} + 8.8 \times 10^{18} X e^{-152.28T_6^{-1/3}}\right) T_6^{-2/3} \frac{\rho}{\text{kg m}^{-3}} \text{W kg}^{-1} \quad (15)$$

where  $T_6 = T/10^6 \text{ K}$ .

**Question 3** The Sun is observed to have mass  $M_\odot = 1.9891 \times 10^{30} \text{ kg}$ , radius  $R_\odot = 6.9598 \times 10^8 \text{ m}$  and luminosity  $L_\odot = 3.8515 \times 10^{26} \text{ W}$ . Estimate the temperature at which nuclear reactions can halt the gravitational collapse of a solar mass star and argue that this is a good initial guess for  $T_c$ . Also, make a linear approximation to the pressure gradient through the Sun, choosing a sensible boundary condition for the pressure at the the stellar surface, to obtain an estimate of the central pressure  $p_c$ .

## 4 A Shooting Solution

We are now set up to find a numerical solution to the equations. A logical way to proceed is to guess central values of the variables,  $T_c$  and  $p_c$  and integrate the equations to the surface where the solution will not necessarily fit the boundary conditions. Unfortunately, it turns out that such direct integrations either from the centre to the surface or from the surface to the centre diverge unacceptably at the surface or the centre for small changes in the undetermined boundary variables (the equations are non-linear). They are, however, well behaved in between. We can therefore integrate both outwards from the centre and inwards from the surface (guessing the radius  $r = R_*$  and luminosity  $L_r = L_*$  at  $m = M_*$ ) and meet at some point in the interior  $m = M_s$ . We can then vary  $R_*$ ,  $L_*$ ,  $p_c$  and  $T_c$  until the two solutions are continuous at  $m = M_s$ . *Hint: The boundary conditions need to be specified at the centre ( $M_r = 0$ ) and surface ( $M_r = M_*$ ), then quantities found from solving the model ODEs.*

The questions below seek solutions for a star of mass  $3 M_\odot$ .

**Question 4** Use the shooting method to solve for a simplified stellar structure from equations (1) and (2). Just for this question, make the approximation that  $T \propto \rho^{\gamma-1}$ , and take the radius of the star to be  $R_* = 1.5 R_\odot$ . Integrate both inwards and outwards to some suitable intermediate point (e.g.,  $M_s = 0.5M$ ). Describe how the boundary conditions at the surface and centre are implemented in your integrations. Using the differences of the dependent variables at  $M_s$  as a measure of the quality of the fit, discuss how the fit varies with  $p_c$  and  $T_c$  and use this knowledge to find values for  $p_c$  and  $T_c$  that match the dependent variables at  $M_s$  to within 1% of their values there. Provide a plot with your best solution. Also indicate on this plot how the solution changes with  $p_c$  and  $T_c$ .

**Question 5** Now include equations (3), (5) and the associated boundary conditions to implement the shooting method to find a more detailed stellar structure model. Describe how the new boundary conditions differ from those used in question 4. From question 4 you have a reasonable first estimate for  $p_c$ ,  $T_c$  and  $R_*$ , but an estimate of  $L_*$  is required because a  $3 M_\odot$  star is significantly brighter than the Sun. An estimate may be made by shooting inwards from the surface to close to the centre using the complete set of equations. Discuss, and illustrate concisely, how the values of each of the dependent variables close to the centre vary as  $L_*$  is varied. Use this knowledge to give an estimate of  $L_*$ . *Hint: When do the estimates of  $L_*$  become unphysical?*

**Question 6** Now shoot both in and out to an intermediate point, and as before, use the differences of the dependent variables at that point to determine the quality of the solution. Use your initial estimates for  $R_*$ ,  $L_*$ ,  $p_c$  and  $T_c$ , and your observations of how the fit changes as these parameters are varied, to describe how to obtain by hand a relative difference in the dependent variables at  $M_s$  that is less than 10%. Provide your best-fitting parameters, the relative difference in each of the dependent variables at  $M_s$  and a plot of each dependent variable against  $m$ .

Adjusting the solution by hand is not a very effective way of refining the parameters to get an accurate solution. Let  $\Delta x_i(R_*, L_*, p_c, T_c)$ ,  $i \in 1, 2, 3, 4$  be the differences between the inward and outward values of each variable  $x_i$  at the intermediate point. A better estimate for these parameters can be found by applying the correction  $(\Delta R_*, \Delta L_*, \Delta p_c, \Delta T_c)$  from the solution to the matrix equation:

$$\begin{pmatrix} \frac{\partial \Delta x_1}{\partial R_*} & \frac{\partial \Delta x_1}{\partial L_*} & \frac{\partial \Delta x_1}{\partial p_c} & \frac{\partial \Delta x_1}{\partial T_c} \\ \frac{\partial \Delta x_2}{\partial R_*} & \frac{\partial \Delta x_2}{\partial L_*} & \frac{\partial \Delta x_2}{\partial p_c} & \frac{\partial \Delta x_2}{\partial T_c} \\ \frac{\partial \Delta x_3}{\partial R_*} & \frac{\partial \Delta x_3}{\partial L_*} & \frac{\partial \Delta x_3}{\partial p_c} & \frac{\partial \Delta x_3}{\partial T_c} \\ \frac{\partial \Delta x_4}{\partial R_*} & \frac{\partial \Delta x_4}{\partial L_*} & \frac{\partial \Delta x_4}{\partial p_c} & \frac{\partial \Delta x_4}{\partial T_c} \end{pmatrix} \begin{pmatrix} \Delta R_* \\ \Delta L_* \\ \Delta p_c \\ \Delta T_c \end{pmatrix} = - \begin{pmatrix} \Delta x_1 \\ \Delta x_2 \\ \Delta x_3 \\ \Delta x_4 \end{pmatrix}_{(R_*, L_*, p_c, T_c)}, \quad (16)$$

where  $\frac{\partial \Delta x_i}{\partial R_*}$  can be determined by computing  $\Delta x_i(R_* + dR_*, L_*, p_c, T_c)$ , and likewise for the other partial derivatives. This procedure can be iterated until a desired accuracy is achieved.

**Question 7** Implement the matrix method described above and iterate until the parameters are accurate to within four significant figures. Provide plots of each variable with respect to  $m$ , a table with the values of each variable at the meeting point and your values of the parameters. *Hint: You may need to rescale the variables and parameters so that the matrix is well-conditioned and the inversion is stable.*

**Question 8** The model you find does not agree with observations of such stars. Suggest what needs to be added to the model to make it more realistic.