



UNIVERSITY OF
CAMBRIDGE

Mathematical Tripos Part IB

Computational Projects
2011/2012

CATAM

Mathematical Tripos Part IB Computational Projects

July 2011

Edited by the Computational Projects Assessors Committee

Compiled by Dr D. Barden, DPMMS

Course Director: Dr S.J. Cowley, DAMTP

Computer-Aided Teaching of All Mathematics

Faculty of Mathematics

University of Cambridge

Contents

Introduction

Optional Introductory Non-Examinable Project

- 0.1 Root Finding in One Dimension

Core Projects

- 1.1 Ordinary Differential Equations
- 1.2 Golden Section Search for the Mode of a Function

Additional Projects

- 2.1 The Restricted Three Body Problem
- 2.2 Equipotentials in a Parallel Plate Capacitor
- 2.3 Continued Fractions
- 2.4 Computational Hypothesis Testing

For maximum credit, you should attempt both projects from section 1 (*Core Projects* above), and then two additional projects chosen from section 2 (*Additional Projects*). You may not attempt more than two additional projects. All projects carry equal credit.

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 first Computational Projects course is an element of Part IB of the Mathematical Tripos (there is a second Computational Projects course is an element of Part II). Although a Part IB course, lectures and introductory practicals are given in the Easter Term of the Part IA year. After the lectures and practicals you may work at your own speed on the examinable projects.

The course is an introduction to the techniques of solving problems in mathematics using computational methods. **It 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. At the time of writing (July 2011) this booklet 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 notebooks. The maximum credit obtainable is 80 marks and 4 quality marks (alphas or betas), which is roughly the same as for a 16-lecture course. Credit obtained is added to the credit gained in the written examination.

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. Particularly with respect to the Additional Projects (2.1 to 2.4), 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 to substantiate your arguments, or support your methodology.

1.3 Timetable

The timetable below is given as a guide to the expected workload.

Easter Term, Part IA: attend the introductory lectures and practicals. If you have no previous computing experience then you may need to spend extra time learning the basics; the summer vacation is a good opportunity to do this.

Over the summer and/or at the start of Michaelmas Term, Part IB: consider doing the **optional**, non-examinable, introductory project. Unlike the other projects you may collaborate as much as you like on this project, and (if your College is willing) have a supervision on it. A model answer will be provided a few weeks into the Michaelmas Term.

Michaelmas Term and Christmas vacation, Part IB: carry out the **two core projects** and write them up. Please make sure that you have read and understood §5, *Guidelines for Collaboration*, before starting these projects. A good *aim* is to finish these projects by the end of the Christmas vacation.

Lent Term and Easter vacation, Part IB: you have one week at the start of Lent Term to make *last-minute* changes to the core projects, which should then be submitted (see §6.2 below). Then undertake **two additional projects** (out of a choice of four) and write them up. Between the time of submission and the end of Lent Full Term, you may be called either for a routine *viva voce examination*, or for an *Investigative Interview* if unfair means are suspected (see §5.1).

Easter Term, Part IB: you have one week to make *last-minute* changes to the additional projects. Then submit your work (see §6.2 below).

After the examinations: **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 *Investigative Interview* if unfair means are suspected (see §5.1).

Do not leave writing up your projects until the last minute. Further, 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 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 review your write-ups in the final week before the relevant submission date.

1.4 Programming language[s]

This year the Faculty is supporting MATLAB. During your time in Cambridge the Faculty will provide you with one free copy of MATLAB for your computer. Alternatively you can use the version of MATLAB that is available on the University and College Personal Workstation Facilities (PWFs).

1.4.1 Your copy of MATLAB

There were opportunities to collect your copy of MATLAB (for Windows, MacOS or Linux) in the Easter Term of Part IA. If you still need a copy of MATLAB for your own personal computer you can collect a DVD¹ from the Reception desk at the CMS. You will need to take along your University identity card, or other form of photo identity. Please only collect the MATLAB software if you are taking Part IB of the Mathematical Tripos in 2011-12.

¹ Note that your computer will need a DVD reader.

1.4.2 Programming manual[s]

The Faculty of Mathematics have produced a short 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. Further:

- MATLAB has its own built-in help and documentation.
- *The MathWorks*, the suppliers of MATLAB, provide an introduction *Getting Started with MATLAB*. You can access this by clicking on the **Getting Started** link at the top of a MATLAB ‘*Command Window*’. Alternatively there is an on-line version available at²

http://www.mathworks.com/help/techdoc/learn_matlab/bqr_2pl.html

and a printable version is available from

http://www.mathworks.com/help/pdf_doc/matlab/getstart.pdf

- *The MathWorks* also provide links to a whole a raft of other tutorials

http://www.mathworks.co.uk/academia/student_center/tutorials/launchpad.html

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

<http://www.mathworks.com/help/techdoc/>

- There is also a plethora of books on MATLAB; for instance *MATLAB Guide* by D.J. Higham & N.J. Higham (SIAM, 2nd Ed. 2005, ISBN 0-89871-578-4). Further, Google returns 15,500,000 hits for the search ‘MATLAB introduction’, and 3,650,000 hits for the search ‘MATLAB introduction tutorial’.

1.4.3 To MATLAB, or not to MATLAB

Use of MATLAB is recommended, especially if you have not programmed before, but *you are free to write your programs in any computing language whatsoever*. C, C++, C#, Python, Java, Visual Basic, Mathematica and Maple have been used by several students in the past, and Excel has often been used for plotting graphs of computed results. A more complete list of possible alternative languages is provided in Appendix A. The choice is your own, provided your system can produce results and program listings on paper.³

However, you should bear in mind the following points.

² These links work at the time of writing. Unfortunately *The MathWorks* have an annoying habit of breaking their links.

³ There is no need to consult the CATAM Helpline.

- We do 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 some 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, 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.

2 The Project Reports

2.1 Project write-ups: examination credit

The assessors currently award the write-up of each project a numerical mark out of 20 and a quality mark: an α is normally awarded for marks in the range 15–20, and a β for the range 10–14. Of the 20 marks, 8 are awarded for writing programs that work and for producing correct graphs, tables of results and so on. A further 10 marks are given for answering mathematical questions in the project and for making appropriate mathematical observations about your results.⁴ The final 2 marks are awarded for the overall merit of the write-up. One of the ‘overall merit’ 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 ‘overall merit’ mark may be reduced and could even become negative, as low as -2 .

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. As for any examination in the Mathematical Tripos you are advised to *write legibly; otherwise you place yourself at a grave disadvantage*.

No marks at all are given for a program listing with no report or vice versa.

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

⁴ The 8:10 division in marks was a change introduced in 2010-11.

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 then 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 often 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 your notes, or the WWW, you should give both a brief justification of the result and a *full reference*.⁵ If you are actually asked to *prove* a result, you should do so briefly.
- Graphs will sometimes be required — for instance, to reveal some qualitative features of your results — and we expect such graphs to be computer-generated. Also, note that while it may be easier to print only one graph a page, it is often desirable (e.g. to aid comparison) to include two or more graphs on a page. Further, 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.

⁵ See also the paragraph on *Citations* in §5

The following are indicative of some points that might be addressed in the report, though 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?
- 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.

As an aid, for the **two core projects**, we have provided some brief additional comments giving further guidance as to the form and approximate length of answer expected for each question. These also contain a detailed mark-scheme, on which your marks for each question will be written and returned to you during the Lent term. For additional projects you are expected to use your judgement on the marks allocation.

2.2.1 Project write-ups: advice on length

The word *brief* peppers the last few paragraphs. However, each year many students just do not get it. To emphasise this point, in general **six sides of A4 of text**⁶ should be plenty for a clear concise report. You will need to add tables, graphs, printouts etc. to this total. Do not however 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. Remember that you are writing a report to be read by a *human being*, who will not want to wade through pages and pages of irrelevant or unimportant data. Twenty sides of graphs would be excessive for most projects, even if the graphs were printed one to a page.⁷ The assessors will be allowed to **deduct** up to 2 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.

⁶ Excluding inline graphs, tables, etc.

⁷ Recall that graphs should not as a rule be printed one to a page.

2.3 Project write-ups: technicalities

- As emphasised above, elaborate write-ups are not required. You may use a word processor if you wish,⁸ but *legible* hand-written reports, or some combination of the two, are acceptable (e.g. it may not be wise to spend excessive time typesetting lots of equations when they could just as well be written in by hand).
- So that your candidate number can be added to each project, on the first page of each project write-up you should **write the project number** clearly in the top left hand corner and should leave a gap 11 cm wide by 5 cm deep in the top right hand corner (for a sticky label). Your **name or user identifier should not appear anywhere** in the write-up (including any printouts), as the scripts are marked anonymously. Do not use green or red pen to write your reports. Write or print on only one side of the paper, leave a margin at least 2 cm wide at the left, and number each page, table and graph.
- At the end of each report you should include complete **printed listings** 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: do not mix program listings and program output 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. Many of you will use your own computer. However, you can also use the the CATAM Public Workstation Facility (PWF), commonly referred to as the *CATAM room*, which is located in room GL.04 in the basement of Pavilion G, CMS.

The CMS buildings are generally open from 8.30am–5.30pm, Monday–Friday. They are also open 8.30am–1pm on Saturdays during the Michaelmas and Lent terms (but not the Easter term). They are closed on Sundays. You should not remain in the *CATAM room* when the CMS buildings are locked.

In the past the *CATAM room* has become crowded at the start of the Lent and Easter Terms as the submission deadlines approach. You are strongly advised to complete all your computing work by the end of the Christmas and Easter vacations if at all possible, since the submission deadlines are early in Lent and Easter terms.

⁸ Large numbers of word processing packages are available. Many mathematicians use \LaTeX or \TeX (e.g. this document is written in \LaTeX). Both \LaTeX and \TeX are available on the PWFs. Alternatively they can be installed on your own personal computer for free, e.g. for recommendations and packages see

<http://www.tug.org/begin.html#install> and <http://www.tug.org/interest.html#free>

A Windows version of MiKTeX can also be downloaded from

<http://www.damtp.cam.ac.uk/internal/computing/faq-win/tex/miktex.html>

Alternative word processing packages include *Word* and *OpenOffice*; the former is commercial, while the latter can be installed for free for, *inter alia*, the Windows, MacOS and Linux operating systems (see <http://download.openoffice.org/>). Both *Word* and *OpenOffice* are available on the PWFs.

You should report problems with the PWF, e.g. broken hardware, printers that are not working, to catam-help@damtp.cam.ac.uk. Note that this is a *different* address to that of the *CATAM Helpline*.

You can also use other computing facilities around the University; for further information (including which Colleges are linked to the PWF network) see <http://www.cam.ac.uk/cs/pwf/>.⁹ At most PWF locations you can access the MATLAB software just as in the *CATAM room*, and any files you store on the PWF from one location should be accessible from any other PWF location.

3.1 Out-of-term work

The *CATAM room* is available most of the time that the CMS buildings are open, although the room is sometimes booked for other purposes during the vacations. Effort is made to ensure that it is available in the week after the end of the Michaelmas and Lent Full Terms, and in the week before the start of the Lent and Easter Full Terms. The availability of the room can be checked online at <http://www.damtp.cam.ac.uk/internal/catam/catambook.html>.

3.2 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 the system goes down or the disks crash. **Malfunctions of your own equipment or the PWF are not an excuse for late submissions:** leave yourself enough time before the deadline.

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*.

CATAM News and Email. Any important information about CATAM (e.g. corrections to projects or to other information in the *Handbook*, availability of demonstrators, temporary closures of the *CATAM room*) is publicised via *CATAM News*, which is linked to from the *CATAM web page*. You **must read CATAM News** from time to time 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.

⁹ Note that the Phoenix Teaching Room 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.

After 1 October 2011 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-IB* 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 Office (email: faculty-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. you need clarification about the wording of a project, or have queries about programming and/or MATLAB, or need a demonstrator to help debug your programs), you can send a query by email to the *CATAM Helpline*: catam@maths.cam.ac.uk. Almost all queries may be sent to the *CATAM 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 your Hermes email address both so that we may identify you and also so that your email is not identified as spam. In addition please specify, in the subject line of your email, ‘Part IB’ 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 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; for example emails sent directly to the Course Director may be subject to a far longer delay in answering (and could end up either being missed altogether or consigned to `/dev/null`).

In addition to the *Helpline*, at certain times of the year, e.g. in the period immediately before submission, demonstrators may be available in the *CATAM room*. As well as answering queries about general course administration, programming and/or MATLAB, you are allowed to ask demonstrators to help you debug your programs. The times when they will be available will be advertised in *CATAM News*.

The CATAM Part IB Q&A Web Pages. Before asking the *Helpline* about a particular project, please check the *CATAM Part IB Q&A web pages* (accessible from the main CATAM web page). These list questions which students have already asked this year together with the answers, 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; however, you may have a supervision on the *Introductory Project*, and/or another non-examinable project, and/or any work set by your Director of Studies. A supervisor can also provide feedback on the *Core Projects* **after** they have been submitted (e.g. **after** your marks have been returned).

5 Guidelines for Collaboration

You **must** work *independently* on the projects, both on the programming and on the write-ups, i.e. you must write and test all programs yourself, and all reports must be written independently. However, it is recognised that some candidates will 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. Acceptable collaboration may include a *general* discussion of the approach to a project and the numerical algorithms needed to solve it.

Unacceptable collaboration. If a general discussion gets to the point where notes are being exchanged (even on the back of an envelope or on a napkin) then it has reached the stage of unacceptable collaboration. Unacceptable collaboration also *includes*

- copying another 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.

These comments apply equally to copying from the work of previous Part IB students as they do to copying from the work of students in your own year. You should not allow any present or future Part IB 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, whatever the circumstances, you may be penalised yourself.

Citations. It is, of course, perfectly permissible to use reference books, journals, the WWW or other similar material: indeed, you are positively 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.

University and Faculty Statements on Plagiarism. You should familiarise yourself with the University *Statement on Plagiarism* that is reproduced as Appendix B. This statement is posted on the University's plagiarism website

<http://www.cam.ac.uk/plagiarism/>,

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

You should also familiarise yourself with the Faculty of Mathematics *Statement on Plagiarism* that is reproduced as Appendix C. This statement is posted on the Faculty's website at

<http://www.maths.cam.ac.uk/teaching/plagiarism.html>.

This statement includes advice on quoting, paraphrasing, referencing, general indebtedness, and the use of web sources.

Declaration. To certify that you have observed the above guidelines, you will be required to sign a declaration form when you submit your write-ups, and you are advised to read it carefully; it is reproduced (subject to revision) as Appendix D. You must list on the form **anybody** (students, supervisors and Directors of Studies alike) to whom you have talked about the projects at any more than a trivial level: any discussions which were of sufficient extent that they affected your approach to the projects must be listed. Failure to include on your declaration form any substantive discussion you may have had is a breach of these guidelines.

However, declared discussions 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 the *details* of your programs or write-ups with others after starting work on them, then there is unlikely to be a case that the limits have been breached.

The assessors will not have any knowledge of your declaration until after all your projects have been marked. The only time your declaration may affect your CATAM marks is if the assessors believe that discussions have gone beyond the limits of what is acceptable. You will then be summoned to an *Investigative Interview*, and this may lead to a change in the marks you receive for the Computational Projects.

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.** 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 have been copied.

Example. As an instance to clarify the limits of ‘acceptable collaboration’, if an assessor reading two anonymous write-ups (without knowledge of the declared lists of discussions) 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 Interview* would then be arranged.

Sanctions. The Computational Projects are considered to be a single piece of work within the Mathematical Tripos: therefore, if it is concluded that you have used unfair means for one or more individual write-ups, you may stand to lose your marks for the entire Computational Projects course. Unfortunately, there have been cases in recent years where *some individuals have been penalised by the loss of significant numbers of marks and alphas/betas.*

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

However, it is emphasised that the great majority of candidates have had no difficulty in keeping to these guidelines in the past; if you find them unclear in any way you should seek advice from your Director of Studies or the CATAM Director.

5.1 Oral examinations

Viva voce examinations. A number of candidates may be selected, either randomly or formulaically, for *viva voce examination* after submission of either the core or the optional 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.

Investigative interviews. The assessors also have the right to ask you to attend an *Investigative Interview*. If this happens, you have the right to be accompanied by your Tutor (or another representative at your request). One purpose of an *Investigative Interview* might be to confirm that you have fully declared all collaboration.¹⁰

Timing. *Viva voce examinations* and *Investigative Interviews* are a formal part of the Tripos examination, and if you are summoned then you must attend. In the case of the core projects these will usually take place during Lent Full Term (although exceptionally during Easter Full Term), and in the case of the optional projects these will usually take place during the last week of Easter Full Term. Easter Term *Viva voce examinations* are likely to take place on the Monday of the last week (i.e. Monday 11 June 2012), while *Investigative Interviews* may take place any time that week. If you need to attend during the last week of Easter Full Term you will be informed in writing just after the end of the written examinations. **You must be available** in the last week of Easter 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.

6.1 Declaration form

When you submit your projects you will be required to sign a declaration form detailing which projects you have attempted and listing all discussions you have had concerning CATAM (see §5, *Guidelines for Collaboration*, and Appendix D). Further details, including the definitive submission form, will be made available when the arrangements for electronic submission of programs (see below) are announced.

6.2 Submission of written work

In order to gain examination credit, you must:

- submit electronic copies of your programs (see §6.3);
- complete and sign your declaration form;
- submit, with your declaration form, your written reports and program printouts for every project for which you wish to gain credit;
- sign a submission list.

¹⁰ See <http://www.admin.cam.ac.uk/offices/exams/examiners/plagiarism.pdf> for more information.

The location for handing in your work will be announced via *CATAM News* and email closer to the time.

For the core projects the submission date is

Tuesday 24th January 2012, 10am–4pm,

while for the additional projects the submission date is

Tuesday 1st May 2012, 10am–4pm.

No submissions will be accepted before these times, and **4pm on 24th January and on 1st May are the final deadlines**. After these times, projects may be submitted only under exceptional circumstances via your College Tutor, with a letter of explanation. In any case, the CATAM Director will be permitted to **reduce** the marks awarded for any projects which are submitted late (including electronic submission of programs).

6.3 Submission of programs

You are also required to submit copies of your program source files electronically both for inspection by the assessors and to enable automatic checks on the independence of your work to take place. This applies whether you have done your work on your own computer, on the PWF, or elsewhere, and regardless of which programming language you have chosen. Full details of the procedure will be announced about one week before the submission deadlines via *CATAM News* and email, so please do *not* make enquiries about it until then.

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

If you cannot remember your password on the PWF you will need to ask the Computing Service to reset it.¹¹ This can take some time, so check that you know your PWF password well before submission day.

Naming convention. To make submission and marking easier, please put all your files related to different projects in separate directories/folders. Further, please name each directory/folder using a convention whereby the first few characters of the directory/folder name give the project number, with the dot replaced by either a minus sign or underscore (-). For example, all the programs written for project 2.3 should be placed in a directory/folder with a name beginning with 2-3 or 2_3.

6.4 (Non)-return of written work

We regret that students' submitted work cannot be returned to them after the examination; it must be retained in case of a query or an appeal at a later stage. You are recommended to make a photocopy of your work before submitting it. You must, however, submit the original.

A copy of your submission is likely to be particularly useful for the core projects for which you will be given a breakdown of the marks you've obtained. Since the manuals will be taken off-line after the close of submission, you might also like to make a hard copy of the projects you have attempted.

¹¹ E.g. see <http://www.cam.ac.uk/cs/docs/faq/n3.html>.

A Appendix: Other Computer Languages and Packages

There are many computer packages and languages suitable for mathematics, and none is “best”; before MATLAB, the supported languages for CATAM were, respectively, FOCAL, BASIC, Pascal and C. Should you need, someday, to tackle a serious piece of computation then you will need to consider which language or package is most suitable. Factors in this decision include ease of programming, speed of execution and, in some cases, cost of purchase.

Some languages are *compiled* languages where source code written in the language has to be translated to machine code before it can be executed on a computer; other languages are *interpretative* languages such that no translation is necessary before execution (although in practice this distinction can get blurred, for example because most interpreting systems also perform some translation work, just like compilers). Interpretative languages allow you to type simple, or quite complicated, commands directly in to a window, after which the commands are interpreted and the results are printed out directly. Languages like this tend to be easier to learn, and simple programs can be quickly tested. The downside of interpretative languages is that the code typically executes slower than a compiled language. Many people therefore aim for the best of both worlds, by using an interpreted language initially to try simple programming ideas, and then transferring the developed ideas to a compiled language for more intensive work. The various packages and languages can differ a lot in detail, but the fundamental principles are fairly similar, and the experience of doing CATAM should make it much easier for you to pick up another language. It should be added that there are many tools designed for specific mathematical purposes (not mentioned here), so it makes sense to ask other people what they use.

Below are some general packages and languages that you might come across. The list is not complete, nor is it a list of recommendations. Further, while you are welcome to do the CATAM projects in any programming language,¹² **the Faculty only provides support for MATLAB; if you choose a language other than MATLAB you cannot expect support from the Faculty.**

A.1 Mathematical languages and packages

There are a number of programming languages and packages that have been specifically designed for mathematics. Some are specialised (e.g., the software package Magma has been designed to solve computationally hard problems in algebra, number theory, geometry and combinatorics), while others have more general applicability. All packages have their pros and cons, and devoted adherents and detractors. Some are *proprietary*, in which case they will often cost you or others money (but will usually come with professional support), while others are *free* (but will not come with support).¹³ An advantage of *open source* software (normally free) is that you can see what is going on under the hood. Hence an attractive feature of, say, R (see below) is that once you have prototyped in R and are ready to make a production version in, say C, the C code under the hood is readily available for linking or cutting and pasting into your C code.

Below we list a number of packages that have been used for CATAM in the past, or might be

¹² In an average year, something less than 10% of projects submitted do not use the supported programming language.

¹³ It should be noted that while the user interface of these packages can be significantly different, this is often not the case under the hood. For instance, in almost all cases the vector/matrix operations which one uses to program in these languages are essentially implemented by the same (free) C and FORTRAN libraries written 40+ years ago; similarly for the implementations of other common tasks such as optimisation, the numerical solution of ODEs, PDEs, etc.

suitable for CATAM.¹⁴ **However, we emphasise that the Faculty only provides support for MATLAB.**

MATLAB is a *proprietary* numerical computing environment and programming language that allows easy implementation of numerical algorithms, as well as visualisation. It has a graphical debugger, and a Symbolic Math Toolbox allowing access to computer algebra capabilities. There is a comprehensive help facility, and extensive documentation. MATLAB is available on both the Mathematics and the Central/College PWFs.

Octave is a *free* numerical computing environment which is mostly compatible with MATLAB (the Octave FAQ notes that there are still a number of differences between Octave and MATLAB, but in general differences between the two are considered as bugs). Octave does not have MATLAB's graphical interface. Programs might run at different speeds under MATLAB and under Octave, even on the same machine, due to the way the commands are executed; Octave is in general slower. Octave is available for free download for the Linux, MacOS and Windows operating systems from <http://www.gnu.org/software/octave/>.

Scilab is a *free* numerical computing environment. Like all the above packages it is a high level programming language. It is similar in functionality to MATLAB, and the syntax is similar, but not identical to MATLAB (Scilab includes a package for MATLAB-to-Scilab conversions). There is a graphical user interface. Scilab is available for free download for the Linux and Windows operating systems, and some versions of MacOS, from <http://www.scilab.org/>.

Maple is a general purpose *proprietary* mathematics software package that supports both symbolic computations and arbitrary precision numerical calculations, as well as visualisation (i.e., plotting of functions and data). It has a graphical debugger. There is a comprehensive help facility, and extensive documentation. It is the recommended language for some Part II projects. It is available on both the Mathematics and the Central/College Windows PWFs.

Mathematica is another general purpose *proprietary* mathematics software package that supports both symbolic computations and arbitrary precision numerical calculations, as well as visualisation. It has a graphical debugger. It is available on both the Mathematics and the Central/College PWFs. Under an agreement with the University (that expires on 23/08/2013),¹⁵ mathematics students can download versions of Mathematica for the Linux, MacOS and Windows operating systems from

<http://www.damtp.cam.ac.uk/computing/software/mathematica/>

R is a *free* programming language and software environment for statistical and numerical computing, and graphics. R uses a command line interface though several graphical user interfaces are available. For numerical calculations it has similar functionality (but *not* the same syntax) as MATLAB and Octave. R is available for free download for the Linux, MacOS and Windows operating systems from <http://www.r-project.org/>.

¹⁴ Further comparisons are available from Wikipedia, e.g.,

- http://en.wikipedia.org/wiki/Comparison_of_computer_algebra_systems
- http://en.wikipedia.org/wiki/Comparison_of_numerical_analysis_software
- http://en.wikipedia.org/wiki/Comparison_of_statistical_packages

¹⁵ E.g. see the introduction at <http://www.damtp.cam.ac.uk/computing/software/mathematica/>.

A.2 More General Purpose Languages

There are many popular languages in this class (such as C, C++, FORTRAN, Java, Python and so on). To use many of these languages you will need a compiler to convert your program (stored in a text file) into an executable binary file which can then be run.

C is an extremely widely used general purpose programming language. It allows complete control of data at the level of bits and bytes and is very efficient; much of the software you use every day was written in C. More complicated data structures (e.g., polynomials) can be handled by writing suitable functions. However while C offers great flexibility in handling data structures, the syntax is not particularly intuitive and bugs may be hard to detect.¹⁶ There are a number of C compilers available. The best known *freely* available compiler is gcc available from <http://gcc.gnu.org/>.

C++ is another general purpose programming language that is a development of C aimed at higher-level structures (e.g., it introduces some object-oriented features to C). It is in widespread use, for example in the banking sector. As with C there are a number of C++ compilers available; the best known *freely* available compiler is g++ available from <http://gcc.gnu.org/>

C# is a simple, modern, general-purpose, object-oriented programming language. Some view this a Microsoft's answer to Java (see below); others do not.

FORTRAN is one of the oldest languages around. It is a general-purpose programming language that is especially suited to numerical computation and scientific computing. It was designed for computation with real numbers, and its evolved form remains popular in universities and in industry because it is still excellent for this purpose. The best known *freely* available compiler is gfortran available from <http://gcc.gnu.org/>

Java is a programming language that derives much of its syntax from C and C++. Unlike C it is an interpreted language where the interpreter is the, so called, Java Runtime Environment (JRE). Java code will run on any architecture on which a JRE is installed without needing to be recompiled (as a result Java is popular for web applications). Java is available for *free* from <http://www.oracle.com/technetwork/java/>.¹⁷ The computer Laboratory teaches Java to it's students (see <http://www.cl.cam.ac.uk/teaching/current/ProgJava>).

Python is a *free* general-purpose high-level programming language. Its design philosophy emphasises programmer productivity and code readability. Python has a large standard library providing tools suited to many disparate tasks. Because of the wide variety of tools provided by the standard library combined with the ability to use a lower-level language such as C and C++, Python is sometimes viewed as a powerful *glue* between languages and tools. Python is available for *free* download for the Linux, MacOS and Windows operating systems from <http://www.python.org/>.

¹⁶ As such it may not be the ideal language for a beginner to learn.

¹⁷ There is also the *freely* available gcj, the GNU Compiler for Java (see <http://gcc.gnu.org/java/>).

B Appendix: University Statement on Plagiarism

The General Board, with the agreement of the Board of Examinations and the Board of Graduate Studies, has issued this guidance for the information of candidates, Examiners, and Supervisors.¹⁸ It may be supplemented by course-specific guidance from Faculties and Departments.

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. It is both poor scholarship and a breach of academic integrity.

Examples of plagiarism include copying (using another person's language and/or ideas as if they are a candidate's own), by:

- quoting verbatim another person's work without due acknowledgement of the source;
- paraphrasing another person's work by changing some of the words, or the order of the words, without due acknowledgement of the source;
- using ideas taken from someone else without reference to the originator;
- cutting and pasting from the Internet to make a pastiche of online sources;
- submitting someone else's work as part of a candidate's own without identifying clearly who did the work. For example, buying or commissioning work via professional agencies such as 'essay banks' or 'paper mills', or not attributing research contributed by others to a joint project.

Plagiarism might also arise from colluding with another person, including another candidate, other than as permitted for joint project work (i.e. where collaboration is concealed or has been forbidden). A candidate should include a general acknowledgement where he or she has received substantial help, for example with the language and style of a piece of written work.

Plagiarism can occur in respect to all types of sources and media:

- text, illustrations, musical quotations, mathematical derivations, computer code, etc;
- material downloaded from websites or drawn from manuscripts or other media;
- published and unpublished material, including lecture handouts and other students' work.

Acceptable means of acknowledging the work of others (by referencing, in footnotes, or otherwise) vary according to the subject matter and mode of assessment. Faculties or Departments should issue written guidance on the relevant scholarly conventions for submitted work, and also make it clear to candidates what level of acknowledgement might be expected in written examinations. Candidates are required to familiarize themselves with this guidance, to follow it in all work submitted for assessment, and may be required to sign a declaration to that effect. If a candidate has any outstanding queries, clarification should be sought from her or his Director of Studies, Course Director or Supervisor as appropriate.

Failure to conform to the expected standards of scholarship (e.g. by not referencing sources) in examinations may affect the mark given to the candidate's work. In addition, suspected

¹⁸ For the latest version of this statement see

<http://www.admin.cam.ac.uk/univ/plagiarism/students/statement.html>

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's Courts. The Courts have wide powers to discipline those found guilty of using unfair means in an examination, including depriving such persons of membership of the University, and deprivation of a degree.

Discipline Regulation 6

No candidate shall make use of unfair means in any University examination. Unfair means shall include plagiarism¹⁹ and, unless such possession is specifically authorised, 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.

¹⁹ Plagiarism is defined as submitting as one's own work that which derives in part or in its entirety from the work of others without due acknowledgement.

C Appendix: Faculty of Mathematics Statement on Plagiarism

See <http://www.admin.cam.ac.uk/univ/plagiarism/students/statement.html> for the latest version of this statement.

The Board of Examinations publishes information on plagiarism at

<http://www.admin.cam.ac.uk/offices/exams/plagiarism/>

In particular, the definition and scope of plagiarism, together with ways to avoid it, are given in the University statement on plagiarism for undergraduate and graduate students at

<http://www.admin.cam.ac.uk/offices/gradstud/current/submitting/plagiarism.html>

There is a reference to this University statement in the Part III Essay booklet and in each M.Phil. course booklet. **Please read this statement carefully.** It is your responsibility to read and abide by the University statement.

The guidelines below are provided by the Faculty to help students interpret what the University statement means for Mathematics. However neither the University nor the Faculty set of guidelines supersede the University's Regulations as set out in the Statutes and Ordinances. If you are unsure as to the interpretation of either set of guidelines, or the Statutes and Ordinances, you should ask your course director.

What is plagiarism?

Plagiarism can be defined as **the unacknowledged use of the work of others as if this were your own original work**. In the context of any University examination, this amounts to **passing off the work of others as your own to gain unfair advantage**.

Such use of unfair means will not be tolerated by the University or the Faculty. If detected, the penalty may be severe and may lead to failure to obtain your degree or certificate. This is in the interests of the vast majority of students who work hard for their degree through their own efforts, and it is essential in safeguarding the integrity of the degrees and certificates awarded by the University.

Checking for plagiarism

Faculty Examiners will routinely look out for any indication of plagiarised work. They reserve the right to make use of specialised detection software if appropriate.

For information on the procedures that will be followed for handling suspected cases of plagiarism

- in undergraduate courses (Part III, etc.), please see the Board of Examinations statement on *Plagiarism and Collusion* available at

<http://www.admin.cam.ac.uk/offices/exams/examiners/plagiarism.pdf>

- in graduate courses, please see the Board of Graduate Studies guide *Plagiarism: Guide for Supervisors, Examiners, Assessors and Degree Committees* available at

http://www.admin.cam.ac.uk/univ/plagiarism/examiners/examiners_guidance.pdf

The scope of plagiarism

Plagiarism may be due to

- **copying** (this is using another person's language and/or ideas as if they are your own);
- **collusion** (this is collaboration either where it is forbidden, or where the extent of the collaboration exceeds that which has been expressly allowed).

How to avoid plagiarism

Your course work, essays and projects (Part III, M.Phil. etc.), are marked on the assumption that it is your own work: i.e. on the assumption that the words, diagrams, computer programs, ideas and arguments are your own. Plagiarism can occur if, without suitable acknowledgement and referencing, you take any of the above (i.e. words, diagrams, computer programs, ideas and arguments) from books or journals, obtain them from unpublished sources such as lecture notes and handouts, or download them from the web.

Plagiarism also occurs if you submit work that has been undertaken in whole or part by someone else on your behalf (such as employing a 'ghost writing service'). Furthermore, you should not deliberately reproduce someone else's work in a written examination. These would all be regarded as plagiarism by the Faculty and by the University.

In addition you should not submit any work that is substantially the same as work you have submitted, or are concurrently submitting, for any degree, diploma or similar qualification at any university or similar institution.

However, it is often the case that parts of your essays, projects and course-work will be based on what you have read and learnt from other sources, and it is important that in your essay or project or course-work you show exactly where, and how, your work is indebted to these other sources. The golden rule is that **the Examiners must be in no doubt as to which parts of your work are your own original work and which are the rightful property of someone else.**

A good guideline to avoid plagiarism is not to repeat or reproduce other people's words, diagrams or computer programs. If you need to describe other people's ideas or arguments try to paraphrase them in your own words (and remember to include a reference). Only when it is absolutely necessary should you include direct quotes, and then these should be kept to a minimum. You should also remember that in an essay or project or course-work, it is not sufficient merely to repeat or paraphrase someone else's view; you are expected at least to evaluate, critique and/or synthesise their position.

In slightly more detail, the following guidelines may be helpful in avoiding plagiarism.

Quoting. A quotation directly from a book or journal article is acceptable in certain circumstances, provided that it is referenced properly:

- short quotations should be in inverted commas, and a reference given to the source;
- longer pieces of quoted text should be in inverted commas and indented, and a reference given to the source.

Whatever system is followed, you should additionally list all the sources in the bibliography or reference section at the end of the piece of work, giving the full details of the sources, in a format that would enable another person to look them up easily. There are many different styles for bibliographies. Use one that is widely used in the relevant area (look at papers and books to see what referencing style is used).

Paraphrasing. Paraphrasing means putting someone else's work into your own words. Paraphrasing is acceptable, provided that it is acknowledged. A rule of thumb for acceptable

paraphrasing is that an acknowledgement should be made at least once in every paragraph. There are many ways in which such acknowledgements can be made (e.g. “Smith (2001) goes on to argue that ...” or “Smith (2001) provides further proof that ...”). As with quotation, the full details of the source should be given in the bibliography or reference list.

General indebtedness. When presenting the ideas, arguments and work of others, you must give an indication of the source of the material. You should err on the side of caution, especially if drawing ideas from one source. If the ordering of evidence and argument, or the organisation of material reflects a particular source, then this should be clearly stated (and the source referenced).

Use of web sources. You should use web sources as if you were using a book or journal article. The above rules for quoting (including ‘cutting and pasting’), paraphrasing and general indebtedness apply. Web sources must be referenced and included in the bibliography.

Collaboration. Unless it is expressly allowed, collaboration is collusion and counts as plagiarism. Moreover, as well as not copying the work of others you should not allow another person to copy your work.

D Appendix: Example Declaration

PART IB MATHEMATICAL TRIPOS 2011-12

CORE COMPUTATIONAL PROJECTS

STATEMENT OF PROJECTS SUBMITTED FOR EXAMINATION CREDIT

Name:

College User Identifier

Please observe these points when submitting your CATAM projects:

1. Your name, College or user identifier **must not** appear anywhere in the submitted work.
2. The project number should be written clearly in the **top left hand corner** of the first sheet of the write-up. Leave a space 11 cm wide by 4 cm deep in the **top right hand corner** of the first sheet.
3. Complete the declaration overleaf before arriving at the submissions desk.
4. During the submission process your work will be placed into plastic wallets. The individual wallets will be sent to different examiners, so **each project should have its own wallet**.
5. Put your work into the plastic wallets so that the top is at the opening.
6. Without damaging your work or over-filling try not to use more than one wallet per project. (If the pages will not go into the wallet flat, you may need to use more than one wallet.) Ensure that the write-up is at the front and the program listing at the back; if you have used two wallets for a project, they should be securely attached to each other and the second one should contain the program listing.
7. Remember that everyone else will also hit the submissions desk 30 minutes before the deadline. You can avoid a stressful situation by submitting early.

IMPORTANT

You **MUST** check and sign the declaration overleaf 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. It could result in all your marks for the Computational Projects being removed and also render you liable to further sanctions from the Examiners or the University Courts.

DECLARATION BY CANDIDATE

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

Project Number	Brief Title

I certify that I have read and understood the *Guidelines for Collaboration* in the project booklet and that I have conformed with the guidelines given there. I understand that the penalties may be severe if I am found to have broken the *Guidelines for Collaboration* or committed plagiarism. I agree to the Faculty of Mathematics using specialised software to automatically check whether my submitted work has been copied 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 anyone else's work;
- the computer programs and listings and results were not copied from anyone else's work (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 demonstrators, with whom I have had discussions about the CATAM projects, together with the nature of those discussions.

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

SignedDate

0.1 Root Finding in One Dimension

This is an optional, introductory, non-examinable project. Unlike the other projects **there are no marks awarded for it**. Also, unlike the other projects, you may collaborate as much as you like, and (if your College is willing) have a supervision on the project. A model answer will be provided a few weeks into the Michaelmas Term.

The Methods

The aim of this project is to study three numerical methods for finding roots of an algebraic or transcendental equation $F(x) = 0$ by:

- (a) binary search (or bisection or interval halving);
- (b) fixed-point (or Picard) iteration after rewriting the equation in the form or, more precisely, one of the *non-unique* forms, $x = f(x)$, and then using the iteration scheme

$$x_N = f(x_{N-1}), \quad (1)$$

with a suitable initial guess x_0 ;

- (c) Newton-Raphson iteration (which is a special case of fixed-point iteration), using the scheme

$$x_N = x_{N-1} - \frac{F(x_{N-1})}{F'(x_{N-1})}. \quad (2)$$

The theoretical background to these methods is covered in most textbooks on *Numerical Analysis* (a few of which are listed at the end of this project).*

Convergence

If the exact root being approximated is x_* , we define the *truncation error* in the N^{th} iterate as $\epsilon_N = x_N - x_*$. The method is said to have p^{th} -order convergence if

$$|\epsilon_{N-1}| < \eta \quad \Rightarrow \quad |\epsilon_N| \leq C|\epsilon_{N-1}|^p \quad (3)$$

for some positive constants η and C (first-order convergence, $p = 1$, requires $C < 1$). It can be shown that

- (a) binary search is first-order convergent,
- (b) fixed-point iteration, when convergent, is *in general* first-order convergent for a simple root, i.e. one with $F'(x_*) \neq 0$,
- (c) Newton-Raphson iteration, when convergent, is second-order convergent for a simple root, but only first-order convergent for a multiple root.

* There are also articles in *Wikipedia*, but that on fixed-point iteration is in need of some improvement as of July 2011.

Examples

The cases to be studied as examples are

$$F(x) \equiv 2x - 3 \sin x + 5 = 0, \quad (4)$$

and

$$F(x) \equiv x^3 - 8.5x^2 + 20x - 8 = 0. \quad (5a)$$

Note that equation (5a) can be factorised and rewritten as

$$F(x) \equiv \left(x - \frac{1}{2}\right)(x - 4)^2 = 0. \quad (5b)$$

Question 1 Show graphically that equation (4) has exactly one root (which is in fact $-2.88323687\dots$).

Binary Search

Programming Task: write a program to solve equation (4) by binary search.[†] Provide for termination of the iteration as soon as the truncation error is guaranteed to be less than 0.5×10^{-5} , and print out the number of iterations, N , as well as the estimate of the root. Run the program for a number of suitable starting values to check that it is working; include some of these results in your report.

Question 2 Suppose that the rounding error in evaluating $F(x)$ in equation (4) is at most δ for $|x| < \pi$. By considering a Taylor expansion of $F(x)$ near x_* , or otherwise, estimate the accuracy that may be expected for the calculated value of the root.

Hint: note that $|F'(x)| > 4$ for $-5\pi/4 < x < -3\pi/4$.

Fixed-Point Iteration

There are many possible choices of f , e.g.

$$f(x) = x - h(F(x)), \quad (6)$$

for some function[‡] $h(F)$ such that $h(0) = 0$.

Programming Task: write a program to implement the iteration scheme in equation (1) for general f . Provide for termination of the process as soon as $|x_N - x_{N-1}| < \epsilon$ or when $N = N_{max}$, whichever occurs first. Print out the values of N and x_N for each N , so that you can watch the progress of the iteration.

Question 3 Use the program to solve (4) by fixed-point iteration by taking

$$h(F) = \frac{F}{2+k} \quad (7a)$$

in (6), so that

$$f(x) = \frac{3 \sin x + kx - 5}{2+k}, \quad (7b)$$

for some constant k .

[†] You may like to consider using a recursive function.

[‡] Or *functional*.

- (i) First run the program with $k = 0$, $\epsilon = 10^{-5}$, $x_0 = -2$, $N_{max} = 10$. Plot $y = f(x)$ and $y = x$ on the same graph, and use these plots to show why convergence should not occur. Explain the divergence by identifying a theoretical criterion that has been violated.[§]
- (ii) Determine values of k for which convergence is guaranteed if x_N remains in the range $(-\pi, -\pi/2)$.
- (iii) Choose, giving reasons, a value of k for which *monotonic* convergence should occur near the root, and also a value for which *oscillatory* convergence should occur near the root. Verify that these two values of k give the expected behaviour, by running the program with $N_{max} = 20$.
- (iv) Also run the case $k = 16$. This should converge only slowly, so set $N_{max} = 50$. Discuss whether the truncation error is expected to be less than 10^{-5} in this case?
- (v) Discuss whether your results are consistent with first-order convergence.

Question 4 Now use your program to find the *double root* of equation (5a) by fixed-point iteration by taking

$$h(F) = \frac{1}{20} F, \quad (8a)$$

in (6), so that

$$f(x) = \frac{1}{20}(-x^3 + 8.5x^2 + 8). \quad (8b)$$

By considering $f'(x_*)$ explain why convergence will be slow at a multiple root for any choice of differentiable function h in (6).

In your calculations some care may be needed over the choice of x_0 . Also,

- (a) since convergence will be slow, take $N_{max} = 1000$;
- (b) suppress the printing of each iterate, but print out the *final* values of N and x_N .

Is this an example of first-order convergence? Does the termination criterion ensure a truncation error of less than 10^{-5} ?

Note: it can be shown that the truncation error ϵ_N is asymptotic to $40/(7N)$ as $N \rightarrow \infty$.

Newton-Raphson Iteration

A refinement of (6) is to let h depend on the derivatives of F , i.e.

$$f(x) = x - h(F, F', F'', \dots). \quad (9a)$$

In Newton-Raphson iteration

$$h = \frac{F}{F'}. \quad (9b)$$

Programming Task: modify your program to recalculate the root of equation (4), and the double root of equation (5a), using Newton-Raphson iteration.

Question 5 For equation (4), experiment with various x_0 until you have demonstrated a case that converges, and also a case that has not converged in 10 iterations. In the unconverted case, show graphically what happened in the first few iterations.

For both equation (4) and equation (5a) do your (converged) results bear out the theoretical orders of convergence? Comment on the effects of rounding error.

Hint: you may want to use a smaller value for ϵ .

[§] The references at the end may prove helpful.

References

- [1] Epperson, J.F., *An Introduction to Numerical Methods and Analysis*, John Wiley & Sons (2007). ISBN-13: 9780470049631.
- [2] Kharab, A. and Guenther, R.B., *An Introduction to Numerical Methods: A MATLAB Approach*, Second Edition, CRC Press (2005). ISBN-13: 9781584885573
- [3] Press, W.H., Teukolsky, S.A. and Vetterling, W.T. and Flannery, B.P., *Numerical Recipes: The Art of Scientific Computing*, Third Edition, Cambridge University Press (2007). ISBN-10: 0521880688.
- [4] Süli, E. and Mayers, D., *An Introduction to Numerical Analysis*, Cambridge University Press (2003). ISBN-10: 0521810264 (hardback), ISBN-10: 0521007941 (paperback).

1.1 Ordinary Differential Equations

This project builds on theory covered in Part IA Differential Equations and Part IB Methods.

1 Background Theory

The aim in the first part of this project is to study the performance of three different numerical methods for step-by-step integration of a first-order ordinary differential equation (ODE)

$$\frac{dy}{dx} = f(x, y)$$

with given initial conditions. A simple case has been chosen which has an analytic solution for comparison. We denote the exact solution by $y(x)$. In the second part of this project, one of the methods is extended to solve a second-order problem.

The numerical methods to be investigated are as follows.

- (a) The **Euler** method is the simplest method. It employs the scheme

$$Y_{n+1} = Y_n + hf(x_n, Y_n) \tag{1}$$

where Y_n denotes the numerical solution at $x_n \equiv nh$, that is, at the n th step with step length h . The Euler method has first-order accuracy, which means that the local truncation error e_{n+1} is $O(h^2)$ as $h \rightarrow 0$. The local truncation error is found by setting $Y_n = y(x_n)$ (the exact solution at x_n), computing Y_{n+1} using equation (1), then calculating $e_{n+1} = Y_{n+1} - y(x_{n+1})$. On the other hand, the global error in the numerical solution using $n + 1$ steps starting from the initial condition $Y_0 = y(x_0)$ is denoted by E_{n+1} . The Euler method is called a *single-step* method, since Y_{n+1} is obtained from the previous step Y_n .

- (b) The **Leapfrog** (LF) method employs the scheme

$$Y_{n+1} = Y_{n-1} + 2hf(x_n, Y_n) \tag{2}$$

This is a multi-step method (using both Y_{n-1} and Y_n to obtain Y_{n+1}) and has second-order accuracy, i.e. e_{n+1} is $O(h^3)$ as $h \rightarrow 0$. Note that the first step must be taken by a single-step method (e.g. the Euler method).

- (c) The fourth-order **Runge–Kutta** (RK4) method employs the scheme:

$$Y_{n+1} = Y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{3}$$

where

$$\begin{aligned} k_1 &= hf(x_n, Y_n) \\ k_2 &= hf(x_n + \frac{1}{2}h, Y_n + \frac{1}{2}k_1) \\ k_3 &= hf(x_n + \frac{1}{2}h, Y_n + \frac{1}{2}k_2) \\ k_4 &= hf(x_n + h, Y_n + k_3) \end{aligned}$$

The RK4 method has fourth-order accuracy, i.e. e_{n+1} is $O(h^5)$ as $h \rightarrow 0$. When the RK4 method is used on coupled ODEs, each of f , Y_n , k_1 , k_2 , k_3 and k_4 become vectors of the same dimension as the number of coupled ODEs.

The theoretical background for the accuracy and stability of these methods is set out in *A Simple Introduction to Numerical Analysis* by Harding and Quinney, *Applied Numerical Analysis* by Gerald and Wheatley and *Numerical Recipes* by Press *et al.*

2 Specific case to be studied

The case to be studied in detail is

$$f(x, y) = -2y - \frac{1}{4}x^2 + \frac{1}{8} \quad (4)$$

with initial condition $y(0) = 1$. This has the exact solution

$$y(x) = e^{-2x} - \frac{1}{8}x^2 + \frac{1}{8}x. \quad (5)$$

3 Comparison of the numerical methods for solving ODEs

Programming Task: Write program(s) to implement each of the methods (a), (b) and (c) above. You should be able to select any one of these methods and use it to solve the ODE specified by equation (4) subject to the given initial condition.

3.1 Stability of the LF method

Select the LF method.

Question 1 Starting with $Y_0 = 1$, compute Y_n for x up to 10 with $h = 0.2$ (use the Euler method for the first step). Tabulate the values of x_n , the numerical solution Y_n , the analytic solution $y(x_n)$ from (5) and the global error $E_n = Y_n - y(x_n)$. You should find that the numerical result is unstable for this range of x , with the instability manifest by wild oscillations whose magnitude ultimately grows exponentially, proportional to $e^{\gamma x}$ where the ‘growth rate’ γ is a constant.

Repeat with $h = 0.05$. Comment on the effect of reducing h on the size of the instability, and on its growth rate.

Question 2 Compute the analytic solution to the LF *difference* equation

$$Y_{n+1} = Y_{n-1} + 2h \left[-2Y_n - \frac{1}{4} (nh)^2 + \frac{1}{8} \right]$$

with

$$Y_0 = 1, \quad Y_1 = 1 - \frac{15}{8}h \quad (\text{from the Euler method}),$$

and use it to explain the results of the previous question. Would a further reduction of the step length be expected to eliminate the instability, or reduce its growth rate? Would it improve matters to use a more accurate method such as RK4 for the first step?

3.2 Accuracy of the Euler and RK4 methods

This section considers the orders of accuracy of the Euler and RK4 methods. The case to be studied is again the $f(x, y)$ given in equation (4), with the initial condition $y(0) = 1$.

Question 3 Integrate the ODE numerically with $h = 0.4$, from $x = 0$ to $x = 4$, using both the Euler and RK4 methods. Plot a graph of each numerical solution Y_n , and plot the exact solution $y(x_n)$ on the same graph. Compare the exact and numerical solutions.

Question 4 Tabulate, and make a log–log graph of, the magnitude of the error $|E_n|$ at $x_n = 0.4$ versus different values of h from 0.4 to 0.001, for both the Euler and RK4 methods. Comment on the relationship of your results to the theoretical orders of accuracy of the two methods.

4 Numerical solutions of second-order ODEs

This section is concerned with ‘normal-mode’ oscillations of a non-uniform string with ends fixed at $x = 0$ and $x = 1$ [in suitable units]. The string’s transverse displacement $\eta(x, t)$ as a function of longitudinal distance x and time t satisfies

$$m(x) \frac{\partial^2 \eta}{\partial t^2} = \frac{\partial}{\partial x} \left[T(x) \frac{\partial \eta}{\partial x} \right]$$

where $m(x)$ is the mass per unit length of the string and $T(x)$ its tension [which may vary along the string if an external longitudinal force is applied], together with the boundary conditions $\eta(0, t) = \eta(1, t) = 0$ for all t . This problem admits ‘normal-mode’ solutions of the form

$$\eta(x, t) = y(x) \cos(\omega t + \phi)$$

where ω , the [angular] frequency, and ϕ are constants (with $\omega \geq 0$ WLOG), and $y(x)$ satisfies

$$\frac{d^2 y}{dx^2} + \frac{T'(x)}{T(x)} \frac{dy}{dx} + \frac{m(x)}{T(x)} \omega^2 y = 0, \quad y(0) = y(1) = 0. \quad (6)$$

There is a non-trivial solution for y (‘eigenfunction’) only for a discrete set of values of ω (the ‘eigenvalues’).

Suppose first that $m(x) = m_0 e^x$ and $T(x) = T_0 e^x$, where m_0 and T_0 are constants, so that the problem becomes

$$\frac{d^2 y}{dx^2} + \frac{dy}{dx} + p^2 y = 0 \quad \text{where } p = \sqrt{\frac{m_0}{T_0}} \omega, \quad (7)$$

subject to

$$y(0) = y(1) = 0. \quad (8)$$

Question 5 Find the analytic solution of equation (7) subject to the initial conditions $y = 0$, $dy/dx = 1$ at $x = 0$, for a general value of p . Deduce that the smallest (positive) eigenvalue of (7)–(8) is $p = \sqrt{\pi^2 + 1/4}$, and write down the two next smallest eigenvalues.

Equation (7) can be solved numerically by breaking it down into two first-order ODEs. Using the notation

$$z(x) = \frac{dy}{dx}$$

we can write

$$\begin{aligned} \frac{dy}{dx} &= f_1(x, y, z) \equiv z \\ \frac{dz}{dx} &= f_2(x, y, z) \equiv -(z + p^2 y) \end{aligned}$$

These can then be solved using the methods discussed for first-order ODEs. For this part of the project you should use the RK4 method.

Programming Task: Write program(s) to solve equation (7) with initial values $y = 0$, $dy/dx = 1$ at $x = 0$, and to plot a graph of Y_n against x_n .

Question 6 Run your program for the case $p = 3.0$ and a few values of h , comparing the numerical solution Y_n at $x_n = 1$ to the analytic solution $y(1)$. Repeat for $p = 3.5$. Comment on the accuracy of the numerical solutions.

Programming Task: Write a program to search for eigenvalues using interval-halving, stopping when a value of p has been found for which the computed value of Y_n at $x_n = 1$ has $|Y_n| < \epsilon$, where ϵ is a specified small value.

Question 7 Use the program to obtain numerical approximations to the smallest three eigenvalues correct to within 10^{-5} . What values did you use for ϵ and h and why?

Now suppose that $m(x) = m_0 e^x$ but $T(x) = T_0$, where m_0 and T_0 are constants, so that the eigenvalue problem (6) becomes

$$\frac{d^2 y}{dx^2} + p^2 e^x y = 0, \quad y(0) = y(1) = 0, \quad (9)$$

where as before $p = \sqrt{m_0/T_0} \omega$. There is now no simple analytic solution.

Question 8 Modify your program(s) to find the smallest three eigenvalues p of (9) correct to within 10^{-5} , and plot graph(s) of the corresponding eigenfunctions. Explain carefully why you are satisfied that the eigenvalues found are indeed the smallest, and that they have the required accuracy. Comment on the graphs.

Reference

Boyce, W. E., and DiPrima, R. C., 2001, *Elementary Differential Equations and Boundary Value Problems*, 7th edition. Publ. John Wiley & Sons Inc.

Project 1.1: Ordinary Differential Equations

Marking Scheme and additional comments for the Project Report

The purpose of these additional comments is to provide guidance on the structure and length of your CATAM report. Use the same concepts to write the rest of the reports. To help you assess where marks have been lost, this marking scheme will be completed and returned to you during Lent Term. You are advised to keep a copy of your write-up in order to correlate your answers to the marks awarded.

Question no.	marks available ¹	marks awarded ²
Programming task <i>Program:</i> for instructions regarding printouts and what needs to be in the write-up, refer to the introduction to the manual.		
Question 1 <i>Tables:</i> for presentation and layout, refer to the introduction. <i>Comments:</i> you will need a way of estimating the growth rate of the instability. [approx. 3 lines] ³	C2+M0	
Question 2 <i>Analytic solution:</i> do not include trivial steps in your worked answer. <i>Comments:</i> to understand the relationship between the solutions of the difference equation and the differential equation, consider the limit $h \rightarrow 0$ but $n \rightarrow \infty$, so that $x_n \equiv nh$ is non-zero. Explain the relationship and how it predicts the instability observed in Question 1. Give brief answers with reasons to the specific questions posed. [approx. 15 lines] ³	C0+M1 C0+M3	
Question 3 <i>Graphs:</i> you may use either one graph or two.	C1+M0	
Question 4 <i>Comments:</i> what are the corresponding orders of the methods and how do e_n and E_n vary with h ? Consider the nature and properties of the lines in the graph. Relate the latter to the corresponding orders of accuracy. [approx. 6 lines] ³	C1+M1	
Question 5 <i>Analytic solution:</i> Do not include trivial steps in your worked answer; include some very brief comments if appropriate; write down the values of the smallest three eigenvalues. [approx. 10 lines] ³	C0+M1	
Question 6 <i>Analytic solution and numerical solution compared:</i> The reason for computing an analytic solution is to check that the program gives the right answer ('validation'). For each of the two cases, tabulate the numerical solution Y_n at $x = 1$ and the global error $Y_n - y(1)$ against h for 3 or 4 representative values, and comment [approx. 2 lines] ³ .	C1+M0.5	
Question 7 <i>Numerical approximations to the smallest three eigenvalues:</i> Write down the values of p found numerically; explain how you have chosen ϵ and h to ensure the required accuracy. [approx. 10 lines] ³	C1+M1.5	
Question 8 <i>Comments:</i> explain the shape of the eigenfunctions both mathematically and in terms of the physical model. [approx. 20 lines] ³	C2+M2	
Quality marks awarded for (among other things) mathematical clarity and good, clear output (graphs and tables — see the introduction to the project manual).	Q2	
Total marks	20	

¹ C#+M# : computational and mathematical marks

² For use by the assessor

³ This figure is only meant to be indicative of the length of your answer, rather than the exact number of lines you are expected to write

1.2 Golden Section Search for the Mode of a Function

This project is not strongly related to any particular course.

1 Definition

A function f is *unimodal* on $[a, b]$ with a maximum at $c \in [a, b]$ if f is strictly increasing on $[a, c]$ and strictly decreasing on $[c, b]$. A function f is unimodal on $[a, b]$ with a minimum at c if and only if $-f$ is unimodal with a maximum at c .

2 The golden search method

Let f be unimodal with a maximum and let x, y be points in $[a, b]$ with $a < x < y < b$. We wish to locate the mode (i.e. the unique maximum) of f while minimising the number of evaluations of f that are required; this will be useful in cases where it is very expensive or time-consuming to evaluate f . If we evaluate f at x and at y we can make use of the following implications:

$$(i) \quad f(x) \geq f(y) \quad \Rightarrow \quad \text{mode lies in } [a, y]$$

$$(ii) \quad f(x) \leq f(y) \quad \Rightarrow \quad \text{mode lies in } [x, b]$$

to narrow the region of search for the mode to a shorter interval within which we already have one evaluation of f . A second evaluation within the subinterval enables us to repeat this process. We terminate the algorithm when the size of the interval is smaller than twice the required precision.

A convenient way to do this, with an efficiency not much less than that of the theoretical optimal method, is to divide $[a, b]$ at points x, y such that

$$(y - a)/(b - a) = (\sqrt{5} - 1)/2 \tag{1}$$

and

$$b - x = y - a; \tag{2}$$

that is, the interval is divided from each end in “golden section”.

Question 1 Prove that the subinterval in which the mode is deduced to lie is found to be already divided in golden section from one end by the point in its interior at which we already have a function evaluation.

3 Programming

Programming Task: Write a program to implement the golden section search (to locate modes that are either maxima or minima). Your program should prompt the user to enter the interval’s boundaries and the required precision. Ensure also that your program does not evaluate f more often than is necessary. In your write-up explain the following points.

- (i) How does your program deal with the possibility that $f(x) = f(y)$ on one or more iterative steps?

- (ii) Is it preferable to use equation (1) or equation (2) to locate the point for the second function evaluation in each new subinterval, and why?
- (iii) How would your program function if the mode's position were at an end-point of the original interval?

Question 2 As a check that you understand the method, first program it to find the position of the mode in $[0, 1]$ of the function

$$f(x) = 1 + x + x^2 - 4x^4$$

to some appropriate accuracy. Your output should include the mode, the number of iterations performed and an indication of how accurate your result is.

3.1 Computational cost

Consider the alternative (and more intuitively obvious) algorithm in which, instead of using (1) and (2), the subdivisions are defined by $x - a = \frac{1}{3}(b - a)$, $y - a = \frac{2}{3}(b - a)$.

Question 3 What is likely to be the most time-consuming part of either algorithm in a real-life problem. How would the number of numerical operations required for this alternative algorithm compare with that required for the golden section search algorithm. Give quantitative estimates if possible.

[Note that no additional computational work should be done to answer this question.]

4 Theoretical considerations

Question 4 What properties of the function $f(x)$ determine the numerical accuracy that is attainable?

Question 5 If the mode was located to some accuracy, what would be the corresponding accuracy in the height of the mode? How does your answer depend on the properties of $f(x)$?

5 Application: Optimization of gramophone design

The playing region of a *gramophone record* is an annulus of inner radius r and outer radius R . The record spins about its centre S . Sound is picked up from grooves in the record by a *stylus*, which is mounted at the end P of a *tone-arm* QP of length l which is fixed at Q (a point beyond the outer perimeter of the record). The grooves run (almost) in concentric circles around S . The sound pick-up is optimal if the stylus points in the same direction as the groove at P . See Figure 1, a plan view.

The designer of the tone-arm has the following parameters under his control:

- (i) the distance d of the tone-arm pivot Q from the centre S of the record;
- (ii) the toe-in angle θ between the tone-arm and the direction of the stylus.

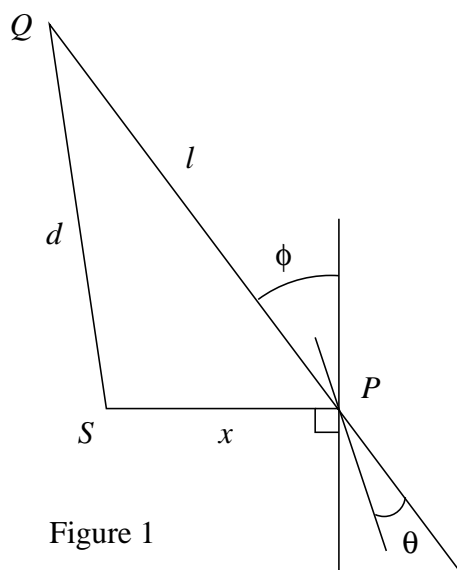


Figure 1

The designer wishes the stylus to track so that its direction diverges as little as possible from alignment with the groove; that is, so that as the stylus moves from the outer edge of the record to the inner one, the maximum absolute value of $(\phi - \theta)$ is minimised, where ϕ is the angle between QP and the tangent to the groove at P .

Question 6 Taking $r = 6.5$, $R = 16$, $l = 24$ (all in cm), find the optimum values of d and θ using golden section search. Your program will need to carry out a double iteration. The inner iteration should find the variation of ϕ considered as a function of x (the distance SP); in other words, find $\Delta\phi = \max\phi(x) - \min\phi(x)$. The outer iteration then adjusts d to minimise $\Delta\phi$, and finally the optimum choice of θ can easily be made.

Your write-up should address the following points:

- (i) What initial intervals did you use for the inner and outer iterations, and why?
- (ii) Is it clear that the functions you were minimising/maximising are unimodal on the relevant intervals?
- (iii) Approximately how many inner function evaluations are required to find d to one decimal place? How is the accuracy of your result affected by the accuracy with which you perform the inner and outer iterations?

Note that no extra marks are available for solving this part of the project analytically, though you may find that this is possible. No proofs are required at any stage: qualitative arguments supported by appropriate graphs will suffice.

Project 1.2: Golden Section Search for the Mode of a Function

Marking Scheme and additional comments for the Project Report

The purpose of these additional comments is to provide guidance on the structure and length of your CATAM report. Use the same concepts to write the rest of the reports. To help you assess where marks have been lost, this marking scheme will be completed and returned to you during Lent Term. You are advised to keep a copy of your write-up in order to correlate your answers to the marks awarded.

Question no.	marks available ¹	marks awarded ²
Programming task <i>Program:</i> for instructions regarding printouts and what needs to be in the write-up, refer to the introduction to the manual.		
Question 1 <i>Comments:</i> An analytical solution is required to show that the subinterval is divided in golden section. Qualitative arguments are acceptable for (i), (ii) and (iii). For (iii), it is insufficient to just show that your program works. [approx. 8 lines] ³	C1.5+M2.5	
Question 2 <i>Comments:</i> Write the values of the mode, number of iterations and accuracy obtained from computation using your program.	C2+M0	
Question 3 <i>Comments:</i> Quantitative details are required for the comparison of the complexity of the golden section search algorithm and the alternative algorithm. No computations should be done. [approx. 15 lines] ³	C0+M1	
Question 4 <i>Comments:</i> Give quantitative arguments where possible. [approx. 4 lines] ³	C1.5+M0.5	
Question 5 <i>Comments:</i> No computations are required. [approx. 5 lines] ³	C0+M1	
Question 6 <i>Comments:</i> This must be solved computationally. Graphs can be used to show that the functions are unimodal. [approx. 20 lines] ³	C3+M5	
Quality marks awarded for (among other things) mathematical clarity and good, clear output (graphs and tables — see the introduction to the project manual).	Q2	
Total marks	20	

¹ C#+M# : computational and mathematical marks

² For use by the assessor

³ This figure is only meant to be indicative of the length of your answer, rather than the exact number of lines you are expected to write

2.1 The Restricted Three-Body Problem

This project may be attempted with knowledge of the Part IA Dynamics and Relativity lecture course.

1 Introduction

The problem of determining the motion of a number of gravitating bodies is a classical one. For two bodies there is a stable analytic solution, describing rotation of the bodies about their joint centre of mass. The problem for three bodies is not soluble analytically. Various simplifications have historically been considered, one of which is the ‘restricted three-body problem’ in which the third body is taken to be much smaller in mass than the other two and therefore does not affect their motion. The problem is then to solve for the motion of the third body under the influence of the gravitational field of the first two bodies.

It is convenient to transform to a rotating frame of reference in which the first two bodies appear stationary and the origin corresponds to their joint centre of mass. Scalings may be chosen so that the angular velocity of this frame is 1 and the distance between the two bodies is 1. The only parameter then appearing is the quantity μ defined such that the two masses are in the ratio $\mu : 1 - \mu$ and are situated respectively at the points $(\mu - 1, 0)$ and $(\mu, 0)$. It is convenient to refer to these points as P_1 and P_2 . The equation of motion for the third body, whose position at time t is $(x(t), y(t))$ may then be written as:

$$\ddot{x} - 2\dot{y} = -\frac{\partial\Omega}{\partial x}, \quad (1)$$

$$\ddot{y} + 2\dot{x} = -\frac{\partial\Omega}{\partial y}, \quad (2)$$

where

$$\Omega = -\frac{1}{2}\mu r_1^2 - \frac{1}{2}(1 - \mu)r_2^2 - \frac{\mu}{r_1} - \frac{1 - \mu}{r_2}, \quad (3)$$

with $r_1^2 = (x + 1 - \mu)^2 + y^2$ and $r_2^2 = (x - \mu)^2 + y^2$.

Despite the substantial restriction to the full three-body problem which this represents, it is not possible to solve (1) and (2) analytically.

Question 1 Show from (1) and (2) that the quantity

$$J = \frac{1}{2}\dot{x}^2 + \frac{1}{2}\dot{y}^2 + \Omega(x, y)$$

is constant following the motion. Deduce that trajectories must be confined to the region

$$\Omega(x, y) \leq \Omega(x_0, y_0) + \frac{1}{2}u_0^2 + \frac{1}{2}v_0^2, \quad (4)$$

where x_0, y_0, u_0 and v_0 are the initial values of x, y, \dot{x} and \dot{y} , respectively.

Programming Task: Write a program to solve (1) and (2) numerically, given suitable initial conditions on x, y, \dot{x} and \dot{y} . You may wish to use a suitable MATLAB ODE solver, such as `ode45`, which automatically adjusts the time-step. You may need to adjust the error tolerance of your solver. If you use a different programming language and a fixed-step integration routine, you may need to use very small time-steps.

Write a second program to plot contours of $\Omega(x, y)$ in the (x, y) plane.

Whenever you write a computer program to find a numerical solution, it is necessary to check that the program is yielding accurate results. Standard checks include (i) testing the program against known analytic solutions (if there are any), and (ii) varying the time-step or error tolerance. For this problem, the fact that J is constant provides another possible check on the accuracy of the numerical solution. In your report, comment on the checks you have performed.

2 Space travel

Assume that the third body is a spacecraft, with the first two bodies being co-orbiting planets of equal size (i.e., $\mu = 0.5$).

Question 2 Consider motion in the neighbourhood of P_2 say, so that the effect of P_1 may be ignored and (3) may be approximated by

$$\Omega = -\frac{1}{2r_2} . \quad (5)$$

Show that (1) and (2) then have analytic solutions with the spacecraft in a circular orbit of radius a about P_2 , where a can take any value. [The approximation (5) is of course only valid if a is small.]

Modify your program to solve (1) and (2) with Ω specified by (5) instead of (3). Demonstrate (for one value of a) that the modified program can reproduce the analytic solutions.

Question 3 Now return to the original problem (1)–(3) with $\mu = 0.5$, and take initial conditions $x = 0.2$, $y = 0$, $\dot{x} = 0$, $\dot{y} = v_0$ with $v_0 = -0.50, -1.00, -1.04, -1.18$, and $-1.25, -1.50$ in turn. For each case, use your programs to integrate from $t = 0$ to $t = 15$, and display the trajectory plotted with the allowed region $\Omega(x, y) \leq J$ on the same plot. (One may alternatively shade in the excluded region $\Omega(x, y) > J$.) Give an indication of the accuracy of your numerical integrations.

Comment on the trajectories, and how these and the allowed region change as v_0 increases. Is the allowed region a useful guide to the size of the trajectory? How does a trajectory behave close to the boundary of the allowed region? Which value of v_0 would be most suitable to travel from the neighbourhood of P_2 to the neighbourhood of P_1 ?

3 Lagrange points and asteroids

In this part of the project do not restrict attention to the case $\mu = 0.5$.

Question 4 By examining contour plots of Ω show that the system (1)–(3) generally has five equilibrium points. (Three lie on the x -axis and the other two — the *Lagrange points* — at the third vertex of an equilateral triangle whose other two vertices are at P_1 and P_2 .) Display contour plots for three values of μ , with the equilibrium points marked.

Investigate the stability of the different equilibrium points numerically, i.e., by starting trajectories a small distance away from the equilibrium point and integrating forward in time. Display plots of some representative trajectories in the (x, y) plane to illustrate your results. (You may also find it useful to look at plots of x or y against t .)

What do you conclude about the stability of the equilibrium points on the x -axis? Do the stability properties depend on μ ? Confirm your findings by performing a linearised stability analysis about such points. You should be able to deduce any necessary information about the second derivatives of Ω by considering the shapes of the contours, rather than by detailed calculation.

Question 5 Continue with a numerical investigation of the stability of the equilateral Lagrange points for parameter values $\mu = 0.008, 0.022, 0.044, 0.08$ and 0.5 . Illustrate the results in your write-up with one trajectory picture for each.

How do the stability properties change with μ ? Either by numerical experimentation, or by performing a linearised stability analysis, find (to two significant figures) the critical value μ_c dividing values of μ for which the point is stable from those for which it is unstable. [This time the linearised stability analysis does require calculation of the second derivatives of Ω .] For the stable cases give a qualitative description of the motion.

Question 6 The Trojans are a group of asteroids observed at the Sun–Jupiter equilateral Lagrange point. For the Sun–Jupiter system $\mu = 9.54 \times 10^{-4}$. Is the persistence of the Trojans at this point consistent with your findings above? The Earth–Moon system has $\mu = 0.012141$ but no analogue of the Trojans is observed. Can you suggest why?

2.2 Parallel Plate Capacitor

This project may be attempted with knowledge of the Part IB Electromagnetism lecture course.

1 Introduction

In elementary electromagnetic theory, the relation between charge Q and potential V of a parallel plate capacitor is found to be

$$Q = C V \quad \text{where} \quad C = \frac{\epsilon_0 A}{d} \quad (1)$$

is the capacitance, A is the area of the plates, and d their separation. V is the potential difference between the plates. The derivation is usually justified by the argument that the electric field E is uniform between the conducting plates and zero elsewhere, and edge effects are ignored.

In this project the plates are assumed to be long, thin and rectangular. The z axis is taken parallel to the longest sides and the x and y axes form a cross-sectional plane in which it is then assumed that the 2-dimensional Laplace equation holds for the potential function $\phi(x, y)$:

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = 0 \quad (2)$$

with boundary conditions

$$\phi(x, y) = \begin{cases} +V/2 & \text{on top plate} \\ -V/2 & \text{on bottom plate} \\ 0 & \text{as } |x| \rightarrow \infty \text{ and } |y| \rightarrow \infty \end{cases} \quad (3)$$

By solving this 2-dimensional problem it is possible to get some idea of the validity of the approximation that E is uniform between the plates and zero elsewhere.

2 Numerical Method

You are to use the *finite difference method* to numerically solve for the potential.

Question 1 Show that

$$\frac{1}{a^2} [\phi(x-a, y) + \phi(x+a, y) + \phi(x, y-a) + \phi(x, y+a) - 4\phi(x, y)] = 0 \quad (4)$$

is a discretized Laplace equation.

Consider the mesh of spatial points which are at the sites of a square lattice in the (x, y) plane. Take the lattice to be finite so that we consider the points $\{(x, y) | x = aX; y = aY\}$ for integers $X \in [1, N_x]$ and $Y \in [1, N_y]$. The constant a is the lattice spacing, which we scale out of the problem by working with the dimensionless quantities X and Y . We will also work with a dimensionless potential $\Phi(X, Y) = \phi(x, y)/(a^2 K)$, where K is a constant with units of charge density over permittivity.

In this coordinate system, let the cross section of the thin bottom plate coincide with the points $(X_L, Y_B), \dots, (X_R, Y_B)$ and those of the thin top plate with $(X_L, Y_T), \dots, (X_R, Y_T)$. The specific integer values of X_L, X_R, Y_B , and Y_T should be able to be varied. The value of the field Φ

at these points should satisfy the boundary conditions (3). Assume that in our dimensionless units that the potential difference between the plates is $V = 1$. As an approximation to the third boundary condition in (3) impose Dirichlet boundary conditions by setting $\Phi = 0$ along the edges of the square lattice.

We will solve (4) using a variant of *Gauss-Seidel iteration*. In any iteration scheme, some initial values are given $\Phi(X, Y) = \Phi^{(0)}(X, Y)$ (consistent with the boundary conditions, of course) and successive iterations are performed: $\Phi^{(k+1)}$ is determined from $\Phi^{(k)}$ (where k indexes the iteration).

Question 2 Explain in a few sentences why

$$\Phi^{(k+1)}(X, Y) = \frac{1}{4}[\Phi^{(k)}(X-1, Y) + \Phi^{(k)}(X+1, Y) + \Phi^{(k)}(X, Y-1) + \Phi^{(k)}(X, Y+1)] \quad (5)$$

should iteratively approach a solution to Laplace's equation.

Eq. (5) is an implementation of Jacobi iteration and converges slowly. A better method is Gauss-Seidel iteration which replaces old with new values of the function as one sweeps through the lattice sites. Assuming that Φ is updated in order of increasing X and Y , we use

$$\Phi^{(k+1)}(X, Y) = \frac{1}{4}[\Phi^{(k+1)}(X-1, Y) + \Phi^{(k)}(X+1, Y) + \Phi^{(k+1)}(X, Y-1) + \Phi^{(k)}(X, Y+1)]. \quad (6)$$

Further improvement may be obtained via *overrelaxation*:

$$\begin{aligned} \Phi^{(k+1)}(X, Y) = & (1 - \omega)\Phi^{(k)}(X, Y) + \frac{\omega}{4}[\Phi^{(k+1)}(X-1, Y) + \Phi^{(k)}(X+1, Y) \\ & + \Phi^{(k+1)}(X, Y-1) + \Phi^{(k)}(X, Y+1)]. \end{aligned} \quad (7)$$

where $0 < \omega < 2$ is the overrelaxation parameter, less than 2 for stability reasons. Typically $\omega > 1$ will improve convergence significantly compared to (6). This last algorithm is often abbreviated *SOR*, for *Successive Over-Relaxation*. More details can be found in elementary numerical methods texts, such as those listed as references below.

Programming Task:

Write a program to solve this problem using the finite-difference scheme and the *SOR* (Successive Over-Relaxation) method. Define the residual

$$r_k = \frac{1}{N_x N_y} \sum_{X, Y} |\Phi^{(k)} - \Phi^{(k-1)}|, \quad (8)$$

which is used in determining when to stop iterations. The k -th iteration is deemed sufficient if r_k is less than a given tolerance, e.g. $10^{-2}, 10^{-4}, \dots$. You should be able to plot lines of constant potential Φ (as a contour plot).

[Before beginning coding, please read below to have in mind what else your program(s) will be expected to do.]

Question 3 For a fixed tolerance, try at least 2 different initial conditions for the potential $\Phi^{(0)}(X, Y)$. Does this have an effect on the number of iterations necessary to reach the stopping condition?

Question 4 Produce and include in your report contour plots of $\Phi(X, Y)$ for the following inputs

set	N_x	N_y	X_L	X_R	Y_B	Y_T
A	8	8	2	6	3	5
B	16	16	4	12	6	10
C	32	32	8	24	12	20
D	32	32	12	20	14	17

Make sure you have reduced the tolerance until you observe very little dependence on the tolerance in the resulting plots.

Question 5 Try different values of $\omega \in (1, 2)$. What is the dependence on N_x (keeping $N_y = N_x$) of the optimal ω , for which convergence to the stopping condition takes the fewest iterations?

Question 6 Explain how the charge on one of the plates could be computed (approximately) using an expression like

$$\begin{aligned}
 Q = & \sum_{X=X_1}^{X_2} [\Phi(X, Y_2) - \Phi(X, Y_2 - 1) - \Phi(X, Y_1) + \Phi(X, Y_1 - 1)] \\
 & + \sum_{Y=Y_1}^{Y_2} [\Phi(X_2, Y) - \Phi(X_2 - 1, Y) - \Phi(X_1, Y) + \Phi(X_1 - 1, Y)] \quad (9)
 \end{aligned}$$

for appropriate values of $X_{1,2}, Y_{1,2}$.

Programming Task: Write code to compute the charge on one of the plates, given a solution $\Phi(X, Y)$ produced by your first program. [Use your judgement regarding whether to have a separate program which does this or whether you will just add this functionality to code already written.] You probably want to maintain some flexibility so that you can also compute the charge elsewhere.

Question 7 Compute the charge Q on one of the plates for sets A, B, C as tabulated in Question 4. How does this computation give us an indication of the effects of using a discrete lattice of points vs. what one would measure in a laboratory experiment (i.e. in the continuum)? You may wish to carry out computations with larger values of N_x, N_y . Keep in mind that precision of 2-3 significant digits should be sufficient. Therefore, you might be able to increase the tolerance in order to speed up the computations, especially as you experiment. Mention any checks performed that your computation of Q using (9) is correct.

Question 8 Compute Q with additional inputs to those tabulated in Question 4 in order to demonstrate the dependence of Q on the length L of the plates in the x -direction and on their separation d . How do your calculations compare with Eq. (1)?

Question 9 What is the effect of using the boundary condition $\Phi(X, Y) = 0$ along the lattice edges compared to the infinite volume boundary condition that the potential vanishes at large $|x|$ and $|y|$? It would be appropriate to include another contour plot or 2 to illustrate your answer.

References

1. S. D. Conte and C. de Boor, *Elementary Numerical Analysis*, McGraw-Hill, 1980.
2. C.-E. Fröberg, *Numerical Mathematics*, Benjamin-Cummings, 1985.

2.3 Continued Fractions

This project requires an understanding of the Part IA course Numbers and Sets.

1 Introduction

Every rational number x has a (finite) continued fraction expansion

$$x = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \cdots + \frac{1}{a_{N-1} + \frac{1}{a_N}}}}$$

where a_0 is an integer, and a_1, \dots, a_N are positive integers with $a_N > 1$. It is customary to use the short-hand $x = [a_0, a_1, \dots, a_N]$. The continued fraction algorithm starts with $x_0 = x$ and computes the sequence of *partial quotients* a_n by the transformations

$$\begin{aligned} a_n &= \lfloor x_n \rfloor \\ x_{n+1} &= \frac{1}{x_n - a_n} \end{aligned}$$

where as usual $\lfloor x \rfloor$ denotes the greatest integer $\leq x$. The algorithm terminates when $x_n = a_n$.

The *convergents* p_n/q_n are defined for $n \geq 0$ by

$$\begin{aligned} p_n &= a_n p_{n-1} + p_{n-2} \\ q_n &= a_n q_{n-1} + q_{n-2} \end{aligned}$$

with initial conditions $p_{-2} = 0$, $p_{-1} = 1$ and $q_{-2} = 1$, $q_{-1} = 0$. It can be shown by induction that $p_n/q_n = [a_0, a_1, \dots, a_n]$.

2 Continued fractions of rational numbers

Question 1 Write a program to compute the continued fraction of a rational number u/v . Your program should use integer arithmetic only. Compute the partial quotients and convergents for

$$\frac{112}{27}, \quad \frac{4803}{3395}, \quad \frac{276272}{72159} \quad \text{and} \quad \frac{1168347147}{863334886}.$$

Estimate the complexity of your algorithm in terms of u and v .

We consider the 2×2 matrices $M_n = \begin{pmatrix} p_n & (-1)^{n-1} p_{n-1} \\ q_n & (-1)^{n-1} q_{n-1} \end{pmatrix}$.

Question 2 Find a matrix identity relating M_n and M_{n-1} . Use this identity to compute $p_n q_{n-1} - p_{n-1} q_n$.

Explain how your program from Question 1 can be modified to compute the highest common factor d of integers u and v , together with integers r and s such that $ru + sv = d$. Makes these changes to your program, and run it on the following examples:

$$\begin{array}{lll} u = 108783374 & u = 942456774 & u = 3022406801 \\ v = 282566945 & v = 528611791 & v = 2773136137. \end{array}$$

3 Solving linear congruences

Euclid's algorithm is an efficient way to compute the highest common factor of two integers. Importantly, it also provides a means to find integers r and s such that $ru + sv = \text{hcf}(u, v)$.

Question 3 How is Euclid's algorithm related to the algorithm in Question 2?

Question 4 Describe clearly how Euclid's algorithm can be used to find all the solutions in the unknown x to the linear congruence $ax \equiv b \pmod{m}$.

Question 5 Implement a routine to solve the linear congruence $ax \equiv b \pmod{m}$. Find all solutions to each of the following congruences. If none exist, state why not.

$$219661x \equiv 196725 \pmod{235948}$$

$$177279x \equiv 395413 \pmod{869233}$$

$$733180x \equiv 598101 \pmod{853631}$$

4 Generators for the group $\text{SL}_2(\mathbb{Z})$

Let $\text{SL}_2(\mathbb{Z})$ be the group of 2×2 integer matrices with determinant 1, under matrix multiplication. In the next question you will show that $\text{SL}_2(\mathbb{Z})$ is generated by

$$S = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \quad \text{and} \quad T = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

Question 6 Show that if A_1 and A_2 are matrices in $\text{SL}_2(\mathbb{Z})$ with the same first column, then $A_1 = A_2 T^n$ for some integer n .

Explain how your program from Question 1 can be modified to write an arbitrary element of $\text{SL}_2(\mathbb{Z})$ as a word in S , T and T^{-1} , for example

$$\begin{pmatrix} 48 & 59 \\ 13 & 16 \end{pmatrix} = T^3 S T^{-1} S T^2 S T^{-4} S T.$$

Make these changes to your program, and run it on the following examples:

$$\begin{pmatrix} 5617 & 6248 \\ 1540 & 1713 \end{pmatrix} \quad \begin{pmatrix} 224841 & 883274 \\ 111079 & 436367 \end{pmatrix} \quad \begin{pmatrix} 1563177 & 9222253 \\ 1616948 & 9539485 \end{pmatrix}.$$

You should check your answers by multiplying out.

5 Continued fractions of real numbers

The partial quotients a_n and convergents p_n/q_n of an irrational number x are computed as described in the introduction, except that the algorithm will no longer terminate. It can be shown that $p_n/q_n \rightarrow x$ as $n \rightarrow \infty$, so it makes sense to write $x = [a_0, a_1, a_2, \dots]$.

Your computer almost certainly represents real numbers internally as a special sort of rational (usually with a denominator which is a power of 2), which thus represents all the real numbers in a certain interval. Hence most real numbers cannot be stored exactly, but only to a certain

relative precision, known as the “machine epsilon” ϵ . Here ϵ is the smallest positive real number such that, when calculated by the computer, $1 + \epsilon$ is found to be not equal to 1. (Beware that `eps` in MATLAB uses a slightly different definition.) You will need to find out what ϵ is for the programming language and machine you are using: state its value in your write-up, and also show how you found it (or how you checked that a published value, e.g., `eps/2` in MATLAB, was correct).

From now on we shall assume that any real number stored or calculated in the computer is only accurate to within a relative error of ϵ , so that when a number x is input, the stored value \mathbf{x} actually represents the interval $(\mathbf{x1}, \mathbf{x2})$, where $\mathbf{x1} = \mathbf{x} - \epsilon * \mathbf{x}$ and $\mathbf{x2} = \mathbf{x} + \epsilon * \mathbf{x}$; the true value of x lies in that range. When we use an inbuilt operation (such as addition, division, etc.) or an intrinsic function (`exp(x)`, `sqrt(x)`, etc.) this inaccuracy may be compounded with a further relative error; for example, if in a program \mathbf{y} is set equal to `exp(x)`, we can assume that the *true* value of e^x lies in the range $(\mathbf{y1} - \epsilon * \mathbf{y1}, \mathbf{y2} + \epsilon * \mathbf{y2})$ where $\mathbf{y1} = \mathbf{exp}(\mathbf{x1})$ and $\mathbf{y2} = \mathbf{exp}(\mathbf{x2})$. (This works because `exp` is an increasing function.)

Question 7 Write a program to develop the continued fraction expansion of a real number x , giving the partial quotients and convergents, as far as is justified. Your program should stop as soon as it is not possible to be sure of the next partial quotient.

Use your program to tabulate the partial quotients (and in a few cases the convergents) for the real numbers π , 2π , π^2 , $e - 1$, \sqrt{e} , $\tan(1)$, $\sqrt{2}\tan(1/\sqrt{2})$, $\frac{4+\sqrt{37}}{7}$ and \sqrt{d} for $d = 1, \dots, 20$ and $d = 100, \dots, 120$. You will need to keep track of accumulated errors, as suggested above, remembering to give details in your report.

Comment on any structure you see in these continued fractions expansions. In some cases you may be able to guess how the expansions continue. Comment on finding a good rational approximation to π , 2π and π^2 .

References

- [1] Baker, A. *A concise introduction to the theory of numbers*, CUP, 1984.
- [2] Davenport, H. *The higher arithmetic*, CUP, 1992.

2.4 Computational Hypothesis Testing

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

In the standard framework for hypothesis testing, we compare the observed value of a test statistic to what we would expect it to be under the null hypothesis. For some tests, like the t -test or the χ^2 test, we can look up statistical tables to help us perform this comparison. For other tests, tables are not available; in this project we will see how to use simulation instead of tables.

Student's t -test

To illustrate the idea, consider the t -test. Let X_1, \dots, X_m be independent random variables, all $N(\mu_1, \sigma^2)$, and let Y_1, \dots, Y_n be independent random variables, all $N(\mu_2, \sigma^2)$. Suppose σ^2 is unknown. We wish to test

$$H_0 : \mu_1 = \mu_2 \quad \text{against} \quad H_1 : \mu_1 \neq \mu_2.$$

The standard method for testing these hypotheses can be expressed as a likelihood ratio test, which says “Reject H_0 if the likelihood ratio $\text{LR}(H_1, H_0)$ is greater than C ”, for some suitable value of C .

Question 1 Calculate $\text{LR}(H_1, H_0)$. Show that the likelihood ratio test can be expressed as “Reject H_0 if $|T| > D$ ”, for some suitable value of D , where

$$T = \frac{(\bar{X} - \bar{Y})(m^{-1} + n^{-1})^{-1/2}}{\sqrt{S_{XX} + S_{YY}(m+n-2)^{-1/2}}}, \quad \bar{X} = \frac{1}{m} \sum_{i=1}^m X_i, \quad S_{XX} = \sum_{i=1}^m (X_i - \bar{X})^2$$

and \bar{Y} and S_{YY} are defined similarly.

Recall that $\bar{X} \sim N(\mu_1, m^{-1}\sigma^2)$ and $S_{XX} \sim \sigma^2 \chi_{m-1}^2$, and that \bar{Y} and S_{YY} have similar distributions, and that these four random variables are independent.

The *size* of the test is the probability of a Type I error, i.e., the probability that we reject H_0 when it is true. Under the standard framework for hypothesis testing, we choose C (or equivalently D) so that the test has size α , typically $\alpha = 5\%$. The *power* of the test, given values for $\mu_1 \neq \mu_2$ and σ^2 , is the probability that a Type II error does not occur, i.e., the probability that we reject H_0 given these parameter values.

Question 2 Show that under H_0 the distribution of T does not depend on μ_1 , μ_2 or σ^2 . Explain how one can use this fact to choose D so that the test has a given size α . Use tables to find the critical value D such that for $m = 15$ and $n = 20$ the test has size $\alpha = 5\%$.

For this test, the critical value D (and also the power of the test) can be looked up in standard tables. If we do not have tables accessible, or if the distribution of the test statistic is not standard, we can use computer simulation to draw samples from the distribution of T .

Question 3 Estimate the value D such that $\mathbb{P}(|T| > D) = 5\%$ by generating samples of 10,000 independent realisations of T . Compare to your answer to Question 2.

Programming hint: If your programming environment does not support the generation of normal or χ^2 random variables, see the appendix for notes on how to generate them.

Inhomogenous variances

Suppose that (X_1, \dots, X_m) is a sample of independent $N(\mu_1, \sigma_1^2)$ random variables, and that (Y_1, \dots, Y_n) is a sample of independent $N(\mu_2, \sigma_2^2)$ random variables. Suppose all of the parameters are unknown, and we wish to test

$$H_0 : \mu_1 = \mu_2 \text{ and } \sigma_1^2 = \sigma_2^2 \quad \text{against} \quad H_1 : \mu_1 \neq \mu_2 \text{ or } \sigma_1^2 \neq \sigma_2^2.$$

Question 4 Calculate $\text{LR}(H_1, H_0)$. Show that the likelihood ratio test can be expressed as “Reject H_0 if $|T| > D$ ”, where T is a function (which you should find) of \bar{X} , \bar{Y} , S_{XX} and S_{YY} .

Question 5 Show that under H_0 the distribution of T does not depend on μ_1 , μ_2 , σ_1^2 or σ_2^2 . Find the critical value D for a test of size $\alpha = 5\%$ when $m = 15$ and $n = 20$.

We can also use simulation to find the power of the test. Suppose H_1 is true, and the parameters μ_1 , μ_2 , σ_1^2 and σ_2^2 are given. By simulating a random sample of T given these parameters, we can estimate the power of the test, which is just $\mathbb{P}(T > D | H_1)$.

Question 6 Find the power of the likelihood ratio test under the alternative $\mu_1 = 0$, $\mu_2 = 0$, $\sigma_1 = 1$, $\sigma_2 = 3$.

Question 7 With the same μ_1 , μ_2 , σ_1^2 and σ_2^2 , and $\alpha = 5\%$, find the power of the likelihood ratio test for a range of values of m and n . Subject to $m + n = 10$, what values of m and n yield the most powerful test?

Theory says that the distribution of $2 \log \text{LR}(H_1, H_0)$ under H_0 should approach that of a χ^2 random variable with two degrees of freedom. A standard way to compare two distributions visually is with the quantile-quantile plot: given cumulative distribution functions F and G , one plots for each $p \in [0, 1]$ the point $(F^{-1}(p), G^{-1}(p))$. If one does not know F exactly, one can approximate it by the empirical cumulative distribution function \hat{F} of a random sample (Z_1, \dots, Z_N) drawn from F :

$$\hat{F}(z) = \frac{1}{N} \sum_{i=1}^N 1[Z_i \leq z]$$

This has the convenient property that

$$\hat{F}^{-1}(i/N) = Z_{(i)}$$

where $Z_{(i)}$ is the i th order statistic, i.e., the i th smallest value in the sample.

Question 8 Compare the distribution of $2 \log \text{LR}(H_1, H_0)$ to that of a χ^2 random variable with two degrees of freedom by means of a quantile-quantile plot.

Notes on generating random variables

Let U and V be independent uniform random variables on $[0, 1]$. The MATLAB routine **rand** generates such random variables.

Let $X = -\lambda^{-1} \log U$. Then X is an exponential random variable with mean λ^{-1} . If E_1, \dots, E_n are independent exponential random variables with mean λ^{-1} , then $E_1 + \dots + E_n$ is a Gamma random variable $\Gamma(n, \lambda)$.

Let $\theta = 2\pi U$ and $R = -2 \log V$. Then $\sqrt{R} \cos \theta$ and $\sqrt{R} \sin \theta$ are independent standard normal random variables (i.e., with mean 0 and variance 1). If A_1, \dots, A_d are independent standard normal random variables, then $A_1^2 + \dots + A_d^2$ is a χ_d^2 random variable (with d degrees of freedom).

Note that $\chi_d^2 \sim \Gamma(d/2, 1/2)$.