

Introduction

1 General

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

1.1 Introduction

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

As in Part IB **the course is examined entirely through the submission of project reports**; there are no questions on the course in the written examination papers. The definitive source for up-to-date information on the examination credit for the course is the Faculty of Mathematics Schedules booklet. 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 60 marks and 3 quality marks (alphas or betas), which is the same as that available for a 16-lecture course. Credit obtained on the Computational Projects is added to that 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 CV).

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

1.3 Timetable

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

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

Your write-ups must be submitted in the Easter Term (see §6.2 below). **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 submission date.

1.4 Programming language[s]

As for IB the Faculty of Mathematics is supporting MATLAB for Part II. 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

You should have collected your copy of MATLAB (for Windows, MacOS or Linux) either in the Easter Term of Part IA or during your Part IB year. 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.

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_2p1.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

¹ Note that your computer will need a DVD reader.

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

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, 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.

- 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.

1.4.4 Computer Algebra Systems

Some projects require the use of a Computer Algebra System (CAS). At present we recommend Maple for these; possible alternatives include the Symbolic Math Toolbox in MATLAB, or Mathematica.

Maple is installed on the PWF. Reference manuals and other information are kept in the library in the CATAM room (see §3). Maple is also a sensible choice for several projects other than the ones for which a CAS is actually required, and you should feel free to use it for *any* of the projects, but you should be aware of a few points:

³ There is no need to consult the CATAM Helpline.

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

2 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.

2.1.1 Examination credit: algorithm applied to project marks

As indicated in the Undergraduate Schedules and §1.1, the maximum credit available in Part II for the Computational Projects course is 60 marks and 3 alphas, and a total of 30 units is required to be able to achieve this maximum. Each project unit therefore equates to a maximum of 2 *Tripos marks* and 0.1 *Tripos alphas or betas*.

For each write-up, the project mark (out of 20) that has been awarded, and also its associated project quality mark, are weighted according to the unit allocation for the project; hence for a n -unit project, both the project mark and project quality mark are scaled by $n/10$. These weighted marks are then summed.

Fractional project marks resulting from the weighting process are rounded up or down to the nearest integer, with exact halves being rounded *either* up or down according to the overall

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

quality of the actual raw marks. Quality marks are not rounded,⁵ and hence it is possible to score multiples of a tenth of an alpha or beta in the Computational Projects course.

If you submit n units, where $n > 30$ (i.e. if you submit more the maximum number of units), then each of your marks is automatically multiplied by a factor $30/n$ to ensure that you can score no more than the overall maximum allowed; this scaling also applies to quality marks (which, if need be, are rounded to the nearest tenth).⁶

2.2 Project write-ups: advice

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

- Your comments on the results of the programs should go beyond a rehearsal of the program output and show an understanding of the mathematical and, if relevant, physical points involved. The write-up should demonstrate that you have noticed the most important features of your results, and understood the relevant mathematical background.
- When discussing the computational method you have used, you should distinguish between points of interest in the algorithm itself, and details of your own particular implementation. Discussion of the latter is usually unnecessary, but if there is some reason for including it 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*.⁷ 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.

⁵ Unless you submit more than the maximum number of units: see the next paragraph.

⁶ Needless to say, if you submit fewer than 30 units no upwards scaling applies.

⁷ See also the paragraph on *Citations* in §5

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

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

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 **eight sides of A4 of text**⁸ should be plenty for a clear concise report of a seven or eight unit project.⁹ You will need to add tables, graphs, printouts etc. to this total. Do not however include every single piece of output you generate:

⁸ Excluding inline graphs, tables, etc.

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

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.¹⁰ 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.

2.3 Project write-ups: technicalities

- As emphasised above, elaborate write-ups are not required. You may use a word processor if you wish,¹¹ 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

¹⁰ Recall that graphs should not as a rule be printed one to a page.

¹¹ 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.

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 Easter Term as the submission deadline approaches. You are strongly advised to complete all your computing work by the end of the Easter vacation if at all possible, since the submission deadline is early in the Easter term.

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/>.¹² 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*.

¹² 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.

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.

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-II* mailing list is one of your current lists.

If you are not subscribed to the correct mailing list, then this can be corrected by contacting the Faculty 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 II' as well as the project number and title or other topic, such as 'MATLAB query', to which your email relates. Please also **restrict each email to one question or comment** (use multiple emails if you have more than 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 II Q&A Web Pages. Before asking the *Helpline* about a particular project, please check the *CATAM Part II 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.

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 II students as they do to copying from the work of students in your own year. You should not allow any present or future Part II student access to the work you have undertaken for your own CATAM projects, even after you have submitted your write-ups. If you knowingly give another student access to your CATAM work, 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.¹³

Timing. *Viva voce examinations* and *Investigative Interviews* are a formal part of the Tripos examination, and if you are summoned then you must attend. These will usually take place during the last week of Easter Full Term. *Viva voce examinations* are likely to take place on the Monday of the last week (i.e. Monday 11 June 2012), while *Investigative Interviews* may take place any time that week. If you are required to attend a *Viva voce examinations* or a *Investigative Interviews* 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;

¹³ See <http://www.admin.cam.ac.uk/offices/exams/examiners/plagiarism.pdf> for more information.

- 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.

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

The submission date is

Wednesday 2nd May 2012, 10am–4pm.

No submissions can be accepted before this time, and **4pm on 2nd May is the final deadline**. After this time, 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.¹⁴ 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.

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.

¹⁴ 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,¹⁵ **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).¹⁶ 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

¹⁵ In an average year, something less than 10% of projects submitted do not use the supported programming language.

¹⁶ 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.¹⁷ **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),¹⁸ 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/>.

¹⁷ 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

¹⁸ 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.¹⁹ 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/>.²⁰ 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/>.

¹⁹ As such it may not be the ideal language for a beginner to learn.

²⁰ 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.²¹ 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

²¹ 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 plagiarism²² 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.

²² 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 II MATHEMATICAL TRIPOS 2011-12 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. In particular, that means working out the credit units total in advance.
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	Credit Units
	Total Credit Units	

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