MATH3101/MATH5305 - Course Outline

INFORMATION ABOUT THE COURSE

COURSE AUTHORITY: A/Prof. Thanh Tran
RC-4061
Tel: 9385 7041
Email: thanh.tran@unsw.edu.au

LECTURERS:

- A/Prof. Thanh Tran: Weeks 1--8 and 10--12.
- Dr. William McLean (RC-2085): Week 9.

TUTORS:

- Class tutorials: A/Prof. Thanh Tran.
- Lab tutorials: Dr. William McLean.

CONSULTATION:

- A/Prof. Thanh Tran: Tuesday 4pm--5pm
- Dr. William McLean: TBA

CREDIT: This course counts for 6 Units of Credit (6UOC).

PREREQUISITES:

The prerequisites for MATH3101 are 12UOC in Level 2 mathematics courses, including MATH2011 (Several Variable Calculus) and MATH2120 (Differential Equations) or their higher versions MATH2111 and MATH2130.

For engineering students, alternative prerequisites are

- MATH2019 (DN) and MATH2089 or
- MATH2069 (CR) and MATH2099.

In addition, you must have prior experience with simple procedural programming, such as is provided in MATH2301 Mathematical Computing or MATH2089.
Numerical Methods and Statistics (taken by students in many engineering programs) or COMP1911 or COMP1917.

We assume that you understand, e.g., how variables are used to manipulate data, how algorithms are encapsulated in procedures, how simple input/output works, how if-statements allow branching and how loops allow repetition.

The graduate course MATH5305 has no formal prerequisites but students need to have taken courses roughly equivalent to those mentioned required for MATH3101.

EXCLUSIONS: There is no higher version of this subject and it has no exclusions.

CLASSES:

- **Lectures**: During weeks 1-12 you will have two lectures per week.

<table>
<thead>
<tr>
<th>Day</th>
<th>Room</th>
<th>Week</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monday 11AM-12 noon</td>
<td>TETB-LG03</td>
<td>1-8 and 10-12</td>
</tr>
<tr>
<td></td>
<td>(Tyree Energy Tech. Bldg)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RC-G012C</td>
<td>9</td>
</tr>
<tr>
<td>Wednesday 2PM-3PM</td>
<td>RC-G012C</td>
<td>1-4 and 9</td>
</tr>
<tr>
<td></td>
<td>WEBSTER--250</td>
<td>5-8 and 10-12</td>
</tr>
</tbody>
</table>

- **Tutorial**: During weeks 2-13 you will have one class tutorial per week

<table>
<thead>
<tr>
<th>Day</th>
<th>Room</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wednesday 4PM-5PM</td>
<td>OMB-113</td>
</tr>
<tr>
<td>OR</td>
<td></td>
</tr>
<tr>
<td>Wednesday 5PM-6PM</td>
<td>OMB-113</td>
</tr>
</tbody>
</table>

- **Computer lab class**: During weeks 2-13 you will have one computer lab class per week.

<table>
<thead>
<tr>
<th>Day</th>
<th>Room</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monday 4PM-5PM</td>
<td>RC-G012A</td>
</tr>
<tr>
<td>OR</td>
<td></td>
</tr>
<tr>
<td>Monday 5PM-6PM</td>
<td>RC-G012A</td>
</tr>
</tbody>
</table>

MOODLE: Further information, skeleton lecture notes, and other material will be provided via [https://moodle.telt.unsw.edu.au](https://moodle.telt.unsw.edu.au)
COURSE AIMS

MATH3101 will introduce you to some key ideas and techniques associated with the numerical solution of differential equations, ranging from theoretical questions about the accuracy of finite difference schemes and the efficiency of algorithms, through to implementation in computer codes. The course therefore provides a foundation for postgraduate study and research in many fields that rely on numerical modelling. More than a third of the course is devoted to computer programming for scientific and engineering applications.

We will write programs using a subset of Fortran2008, and introduce a few standard software development tools under Linux.

RELATION TO OTHER MATHEMATICS COURSES:

Two closely related courses are MATH2301 Mathematical Computing and MATH3311 Mathematical Computing for Finance (with its postgraduate version MATH5335 Computational Methods for Finance). Also, in some years we offer an honours course on finite element methods. For engineering students, MATH2089 Numerical Methods and Statistics covers a number of the same topics as MATH2301.

STUDENT LEARNING OUTCOMES:

In addition to learning the mathematical content of the course you will also write computer programs and be introduced to some standard numerical libraries. You will see how the performance of a practical code depends on the efficient implementation of stable and accurate numerical algorithms.

The tutorial exercises will provide practice in written presentation of mathematics, and the laboratory exercises will improve your programming skills.

RELATION TO GRADUATE ATTRIBUTES:

The above outcomes are related to the development of the Science Faculty Graduate Attributes, in particular: 1 Research, inquiry and analytical thinking abilities, 4. Communication, 6. Information literacy.
TEACHING STRATEGIES UNDERPINNING THE COURSE

The lectures focus on computing in weeks 1-4, then on numerical methods for problems in linear algebra and ordinary differential equations in weeks 4-8. This material lays the foundation for an introduction to the numerical solution of partial differential equations during weeks 10-12. The lectures in week 9 will cover shared-memory parallel computing with OpenMP.

In the tutorials and labs, you will work on many small problems, developing the skill set needed to piece together a complete numerical simulation.

RATIONALE FOR LEARNING AND TEACHING STRATEGIES:

MATH3101/5305 poses significant challenges for students because of the breadth of knowledge that must come into play when using a numerical method to solve a partial differential equation. In the first two-thirds of the course, you have an opportunity to work on one topic at a time, gaining sufficient mastery of each to use them together as needed in the final third of the course. If you do not work consistently from week 1 then you are likely to struggle more and more as the session progresses. However, if you keep up with the lecture material and consolidate your understanding through the tutorials and labs, then the later parts of the course should fall into place and reinforce your study of the earlier parts.

The distinct tutorial and lab classes will give you the opportunity to work on both mathematical exercises and computing problems. You cannot afford to neglect either.

ASSESSMENT

The course has two minor and three major assessment tasks, weighted as follows:

<table>
<thead>
<tr>
<th>Assessment</th>
<th>Weight (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computing exercise (week 4)</td>
<td>5%</td>
</tr>
<tr>
<td>Written exercise (week 5)</td>
<td>5%</td>
</tr>
<tr>
<td>Lab test (week 10)</td>
<td>25%</td>
</tr>
<tr>
<td>Assignment (week 12)</td>
<td>15%</td>
</tr>
<tr>
<td>Final Exam (2 hours)</td>
<td>50%</td>
</tr>
</tbody>
</table>
During your lab class in week 4, you will complete a simple computing exercise and upload the source code for assessment.

In week 5 you have to hand in a written exercise.

The Lab test in week 10 will require you to write some short programs under exam conditions.

The assignment due in week 12 will have a computing component and a written component. You must submit YOUR OWN WORK, or severe penalties will be incurred. You should consult the University web page on plagiarism

http://www.lc.unsw.edu.au/plagiarism/

**KNOWLEDGE AND ABILITIES ASSESSED:**

- The lab component of the Class Work will assess your ability to write and modify short Fortran programs, testing your knowledge of the language syntax and your understanding of standard programming constructs.
- The tutorial component of the Class Work will assess understanding of the relevant mathematical concepts, and your competence at routine calculations.
- The Assignment will test your skill at synthesising mathematical theory and practical computation, and at interpreting and presenting numerical results.
- The Final Exam will mainly test your mathematical understanding of the numerical methods discussed in lectures, but will also include some questions on scientific computing.

**ASSESSMENT CRITERIA:**

- For computing tasks, the main criteria will be correctness, efficiency and clarity. In other words, does the program produce the correct answers? Does the program run in a reasonable time with reasonable memory requirements? And can a human readily understand (from reading the source code) what the program does?
- For written tasks, you should set out your working clearly and in a logical sequence, with adequate justification for each step. Aim for the most direct answer possible.
DETAILED COURSE SCHEDULE

The table below provides an overview of the course content.

<table>
<thead>
<tr>
<th>Week</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-4</td>
<td>Scientific Computing</td>
</tr>
<tr>
<td>5-6</td>
<td>Numerical Linear Algebra</td>
</tr>
<tr>
<td>7</td>
<td>Initial Value Problems for ODEs</td>
</tr>
<tr>
<td>8</td>
<td>Boundary Value Problems for ODEs</td>
</tr>
<tr>
<td>9</td>
<td>OpenMP</td>
</tr>
<tr>
<td>10-11</td>
<td>Heat Equation</td>
</tr>
<tr>
<td>12</td>
<td>Poisson Equation</td>
</tr>
</tbody>
</table>

Here is a more detailed breakdown of the topics:

- **Scientific Computing (8 lectures)** Compilers, interpreters, Fortran and Matlab, numeric data types, non-numeric data types, control flow, subroutines, functions, modules, arrays, makefiles, libraries.

- **Numerical Linear Algebra (4 lectures)** Complexity of matrix operations, Basic Linear Algebra Subroutines, Gaussian elimination via LU-factorization, band matrices, Lapack, simple iterative solvers.

- **Initial Value Problems for ODEs (2 lectures)** Explicit and implicit Euler methods, systems of ODEs, stiff problems, higher-order methods.

- **Boundary Value Problems for ODEs (2 lectures)** Central difference approximation, discrete maximum principle.

- **OpenMP (2 lectures)** Shared vs distributed memory, threads, parallel speedup, basic OpenMP directives.

- **Heat Equation (4 lectures)** Method of lines, implicit and explicit Euler methods, stability, Crank−Nicolson scheme.

- **Poisson Equations (2 lectures)** 5-point difference scheme, direct solvers, iterative solvers.
ADDITIONAL RESOURCES AND SUPPORT

MOODLE:

All course materials will be available online by logging on to https://moodle.telt.unsw.edu.au You should check regularly for new materials, as well as for announcements about assessment tasks etc.

TUTORIAL AND LABORATORY EXERCISES:

We will provide exercise sets for the tutorials and labs. In the labs, you will work independently at your own pace, but the tutor will be available to help when necessary. Take advantage of this help: it can save you a great deal of time.

During the lecture the lecturer will outline what is expected to be covered that week in the tutorial and lab. You should prepare by reading the relevant exercises and at least making a start on some of them before the class.

LECTURE NOTES:

A set of outline lecture notes will be provided. You will need to supplement these with a reference book on Fortran. A list of some titles is given below which you can borrow from the library (probably only for short periods).

- An economical option is to buy an electronic title from http://www.fortran.com/books.html in the form of a pdf:


  This book is aimed at complete beginners, and includes many example programs with solutions, but is available only in electronic form.

- Another book that covers some of the topics in the course is


  You can download a pdf of this work for free


- The Software Carpentry website http://software-carpentry.org/ also has many useful resources.
SOFTWARE:

We will use a range of free software, in particular gfortran, the Gnu Fortran
Compiler, and geany, a programming editor that provides a simple integrated
development environment. This software is installed on the computers in the student
labs. See the course homepage for information about running the software on your
own desktop or laptop PC.

REFERENCE BOOKS:

In addition to the electronic book by Morgan and Schonfelder, you may find the
following library books useful for further information about Fortran:

- Michael Metcalf and John Reid, *The F Programming Language*, Oxford
  University Press, 1996, P005.133/FOR/60.
- Michael Metcalf and John Reid, *Fortran 90/95 Explained*, Oxford University
  Press, 1999, P005.133/FOR/58.
- Michael Metcalf, John Reid and Malcolm Cohen, *Fortran 95/2003 Explained*, Oxford University

If you expect to make heavy use of Fortran beyond this course, and have a
reasonable amount of programming experience, then you should consider buying the
book by Metcalf, Reid and Cohen in place of Morgan and Schonfelder.

For a broader treatment of scientific computing you might look at

- Stefan Goedecker and Adolfy Hoisie, *Performance Optimization of
- Suely Oliveira and David Stewart, *Writing Scientific Software: a guide to good

For extra material on the OpenMP, see


The following library books provide additional material on the numerical solution of
differential equations.


Of course, these books cover many topics that are not part of the syllabus.

**COURSE EVALUATION AND DEVELOPMENT**

The School of Mathematics and Statistics evaluates each course each time it is run. We consider the student responses and their implications for course development. The section on OpenMP was introduced in 2010.

**ADMINISTRATIVE MATTERS**

**SCHOOL RULES AND REGULATIONS:**

Fuller details of the general rules regarding attendance, release of marks, special consideration etc are available via the School of Mathematics and Statistics Web page: see the links under Assessment Policies and Exam Information at


**PLAGIARISM AND ACADEMIC HONESTY:**

Plagiarism is the presentation of the thoughts or work of another as one's own. Issues you must be aware of regarding plagiarism and the university's policies on academic honesty and plagiarism can be found at