

IMPERIAL

Computational Partial Differential Equations
(CPDEs)
MATH60025/70025

Spring Term 2: 2024-2025
live document
(updated January 8, 2025)

MATH60025/70025: Computational Partial Differential Equations

Lecturer: Dr Shahid Mughal

Copyright Notice :

©M. S. Mughal (2024) These notes are provided for the personal study of students taking this module. The distribution of copies in part or whole is not permitted.

The lecture notes are an adaptation of earlier notes (2018) on the course developed by Professor J. Mestel, Mathematics Department, Imperial College London.

Email : s.mughal@imperial.ac.uk

Office : Department of Mathematics, Huxley 734

Assessment : 100% by projects

Lectures : 26 + Surgery hours

Module Summary

This module will introduce a variety of computational approaches for solving partial differential equations, focusing mostly on finite difference methods. Students will gain experience implementing the methods and writing/modifying short programs in Matlab or another programming language of their choice. Applications will be drawn from problems arising in areas such as Mathematical Biology and Fluid Dynamics.

Prerequisites : None, other than the core modules from years 1 & 2, but Programming ability in Python is required – MATH50008 module *Partial Differential Equations in Action* will provide some background. You will be expected to produce and/or modify programs in (say) Python or any other language of your choice.

Module Content :

There is some flexibility as to the exact module content and ordering, but the following should be very close:

1. Introduction: How can we solve PDEs on a computer? Finite Difference Methods. Basic types and Classification of PDEs. Well-posedness and the importance of Boundary Conditions.
2. Parabolic equations: Explicit & Implicit Schemes. Maximum principle analysis.
3. Elliptic equations: Iterative methods: How can they be made faster? Jacobi, Gauss-Seidel, relaxation techniques. Multigrid methods and motivation and implementation.
4. Hyperbolic equations: characteristics, up-winding, Lax-Wendroff schemes. Non-reflecting boundary conditions, perfectly matched layers (PML).
5. Combinations, Extensions and Applications: e.g. advection/diffusion and Navier-Stokes equations. Magnetic Induction and heat transport equations. Domain decomposition, operator splitting.

Intended Learning Outcomes :

- Appreciate the physical and mathematical differences between different types of PDEs;
- Outline a theoretical approach to testing the stability of a given algorithm;
- Determine the order of convergence of a given algorithm;
- Demonstrate familiarity with the implementation and rationale of multigrid methods;
- Develop finite-difference based software for use on research level problems;
- Use numerical methods, as an alternative to analytical approaches, to solve mathematical problems with a physical underpinning.

Programming and Coding Proficiency : This course does **NOT** assess your programming/coding abilities. It is expected that you will already be proficient in a programming language like Python, Fortran, C or Matlab. If you have **NO experience in programming** **you should think wisely as to whether this course is for you** – it is possible to learn programming as you go along, but you may find this quite challenging particularly due to time constraints of submitting your assessed work.

Philosophy of the Module : Most of the universe is governed by Partial Differential Equations (PDEs), but the techniques for solving PDEs analytically are few. Yet nowadays we have incredible computer power, and we can obtain accurate approximations to solutions to most physically relevant problems. An understanding of analytical techniques really should be accompanied by the ability to find numerical solutions when required. This module is important for those considering a future in research, but also for those considering getting a job in a scientific & technical fields (including Finance) – these almost invariably involve solving some form of PDE or sets of PDEs with a computer.

It is moderately easy to write inefficient code which solves simple problems adequately. This skill should not be under-rated - many problems you will encounter can be dealt with effectively by a "quick and dirty" approach. Yet, if you want to be able to solve important problems quickly, or difficult problems at all, a good understanding of numerical techniques is vital, and is a well-valued talent. This is what this module aims to engender. At its conclusion, you should be able to make a decent stab at previously unsolved Research-level problems.

Assessment : The module will be assessed solely by projects. The plan is as follows: A relatively short project, worth 25% will be set early on, and returned to you with comments. You will be committed to completing the module on submission of 25%. A final project will be set in week 8, with an additional project released for the Mastery (MSc, MSci). These projects should be submitted electronically, via Blackboard.

1. **CW 1 :** 25% released 28 January, submission 1pm, 10 February.
2. **CW 2 :** 40% released 14 February, submission 1pm, 3 March.
3. **CW 3 :** 35% released 6 March, submission 1pm, 20 March.
4. **CW 4 :** 20% released 6 March, submission 1pm, 4 April (**Mastery**).

The assessment for each CW is based upon you solving a given problem having written your code, and followed up by a **detailed technical report** which outlines your findings. You should not expect a high mark, if you only submit your code but fail to submit the written report.

During the module, various Matlab codes will be demonstrated in lectures and released on Blackboard. Most of you will choose to modify these codes for the problems set in the projects, but it is quite permissible to write your programs in any language which runs on the Imperial College machines, e.g. Python, C, Matlab, Fortran.

The projects will involve demonstrating that your codes work to the expected accuracy, obtaining solutions to set problems and discussing the results. The projects may build on one another slightly; you may be able to use your codes from the first two projects as part of the final project. The final project will be at a high level, the kind of problem which borders on Research level. At the end of the module you should be able to solve difficult problems using the various techniques covered.

More topics will be covered in lectures that will/may be of direct use in the Projects, just as some topics do not get assessed in examinations.

General Comments on Project Modules :

Most Maths modules are assessed mainly by a summer exam. Some memory is required - in 2-3 hours you demonstrate what you have learned from the lectures and the many hours of revision. In project modules, memory is not a factor, and you can spend a very long time on them if you choose. As a result, average marks on project modules tend to be a little higher. Nevertheless, 100% is not achievable without deep understanding, and you should not expect perfection. In the past, some students have spent too long on their projects and neglected revision of other topics – I can only caution you against this. The Senior Tutor may advise you not to attend too many project modules.

Collaboration :

We do not, of course, wish to discourage discussion between you on any of the mathematical and computational issues underlying this module. However, plagiarism considerations are very important in project modules. The projects really must be produced independently and any help you received MUST be acknowledged in your submissions. You must adhere strictly to the plagiarism guidelines you have been given – if you are in any doubt about what is permitted you should seek clarification from the Senior Tutor, Dr Ford. The College penalties are exceptionally severe for breaches and this can jeopardise your entire degree. Please do be sensible about this.

Support Classes :

Every now and again, problem sessions will occur in lectures. There will also be surgery sessions in office hours. The timings of these will be discussed in lectures.

Recommended Books :

See the complete Reading list on Blackboard, but the following are pretty good.

1. LeVeque, Randall J., *Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems*. (**Core**)
2. Tannehill, John C., *Computational fluid mechanics and heat transfer*.
3. Smith, G. D., *Numerical solution of partial differential equations : finite difference methods*.
4. LeVeque, Randall J., *Finite Volume Methods for Hyperbolic Problems*.
5. Toro, Eleuterio F., *Riemann Solvers and Numerical Methods for Fluid Dynamics*, 3rd edition.

MATH60025/70025: Computational Partial Differential Equations**Contents**

1 Brief Motivation	9
2 PDEs - Primer and Definitions	11
3 Finite Difference Methods (FDMs)	14
3.1 Finite-difference formulae via Taylor series expansions	15
3.2 Difference operators	16
3.3 Truncation error	17
3.4 Higher order and sided-differences	18
3.4.1 4^{th} -Order or $O(h^4)$ accurate finite-differences	18
3.5 Finite-difference formulae via Lagrange interpolation	19
3.6 Our first finite difference method	20
3.6.1 Numerical stability of ODEs	22
3.6.2 General Error discrete form	24
4 Classification of 2nd-order Quasi-linear PDEs in Two Variables	25
4.1 Example: Significance of characteristics	28
4.1.1 General solution of the wave equation D'Alembert's solution	29
4.2 Boundary conditions for well-posed problems	31
4.3 Hyperbolic equations	31
4.4 Elliptic equations	31
4.5 Parabolic equations	32
4.6 Change of type : fluid flow example	33
4.7 Classification of 1st-order quasi-linear PDEs in two variables	37
5 The Explicit Method for the 1-D Diffusion Equation	38
5.1 Maximum Principle Analysis (MPA)	39
6 Explicit Method for the General Non-linear Parabolic Problem in 1-D	41
7 Implicit Scheme for the 1-D Diffusion equation	44
7.1 The Crank-Nicolson method and (nearly) Tridiagonal systems	46
7.2 Aside – Separation of variables solution to Diffusion eqn.	47

8 Implicit Method for Nonlinearities	48
9 Matrix Representation. Neumann and Periodic Boundary Conditions	49
9.1 Eigenvalues of a Toeplitz matrix	50
9.2 Neumann boundary conditions	51
9.3 Periodic boundary conditions	52
10 Diffusion in More Dimensions; the ADI Method	54
10.1 Implicit ADI	55
10.2 ADI in three dimensions	56
11 Elliptic PDEs : Computer Solution by Iteration	58
11.1 Solution by Iteration	59
11.2 Direct versus iterative techniques	61
12 Iterative Methods for Elliptic Problems	62
12.1 Technical note on convergence	64
12.2 Successive Over-Relaxation	67
13 Introduction to Multigrid Methods	69
13.1 More than 2 grids	71
13.2 Full multigrid	71
14 Grids and Clustering	72
14.1 Grid Stretching, an ODE example	73
14.2 Adaptive Stencils	77
14.3 Elliptic Schemes	79
14.4 Complex 3D meshing	81
15 Hyperbolic PDEs	82
15.1 First-order PDE	82
15.1.1 Example solution to first-order PDE	83
15.2 The one-dimensional linear advection equation and Upwinding	84
15.3 Upwinding for simultaneous equations	87
16 Dissipation and Dispersion	88
16.1 Summary of findings through numerical experiments	88
16.2 Modified Equations	90
16.3 Dispersion-dissipation analysis	92

16.4 Dispersion and dissipation in discretised equations	94
17 The Lax-Wendroff method and conservation equations	96
17.1 Lax-Wendroff schemes for simultaneous conservation equations	97
17.2 Examples of Conservation Forms	98
18 NSE and Conservation Form	100
18.1 Compressible Viscous Navier-Stokes Equations	100
18.1.1 Inviscid Conservation-Law (Euler) Form – Ideal Gas Dynamics	100
18.2 Diagonalisation and Characteristic Variables	102
18.3 Integral Forms of Conservation Laws	105
18.4 Finite Volume Conservation Methods	106
18.5 Shocks : Discontinuous Solutions	107
18.5.1 Shock formation time	109
18.5.2 Rankine-Hugoniot conditions and shock speed	110
18.6 The Riemann Problem	110
18.6.1 Entropy condition and rarefaction waves	111
18.7 Riemann Solution for the Inviscid Burgers Equation	114
19 Finite Volume Methods for Nonlinear Conservation Laws	115
19.1 Godunov's First-Order Upwind Method	115
19.2 Boundary conditions	117
19.3 CFL time step	118
20 The 2D Wave Equation	119
20.1 2D Wave equation: non reflecting boundary conditions	120
21 Perfectly Matched Layers	122
21.1 A more general wave equation	123
21.2 A perfectly matched layer in 2D	124
22 The Method of Characteristics	126
23 Operator Splitting – Fractional Step Methods	130
24 The Navier-Stokes equations in two-dimensions	132
24.1 Nearly inviscid flow, $\nu \rightarrow 0$, High Reynolds Number	133
24.2 Stokes flow: $\nu \rightarrow \infty$, Low Reynolds Number	133
24.3 The general case – intermediate Reynolds number	134

24.4 Rayleigh-Benard Convection	134
24.5 Molten cores of planets and moons	135
25 The Keller-Box method for nonlinear parabolic PDEs	137
25.1 Discretisation and Newton Linearisation	139
26 Multistage Methods	144
26.1 Compact differences	146
26.1.1 Elliptic BVPs and 2D spatial discretisation	147
26.2 Semi-discretisation and the method of lines (MoL)	148
27 Further Reading	150