

Introduction to Numerical Methods
and MATLAB Programming for Engineers

Todd Young and Martin J. Mohlenkamp
Department of Mathematics
Ohio University
Athens, OH 45701
youngt@ohio.edu

April 29, 2021

Copyright © 2008, 2009, 2011, 2014, 2016, 2017, 2018, 2020, 2021 Todd R. Young and Martin J. Mohlenkamp.

Original edition 2004, by Todd R. Young.

Thanks go to many students who have pointed out typos and mistakes. Thank you to Dr. Yaqin Feng for suggestions to improve the exercises.

This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-sa/4.0/>.



Preface

These notes were developed by the first author in the process of teaching a course on applied numerical methods for Civil Engineering majors during 2002-2004 and was modified to include Mechanical Engineering in 2005. The materials have been periodically updated since then and underwent a major revision by the second author in 2006-2007.

The main goals of these lectures are to introduce concepts of numerical methods and introduce MATLAB in an Engineering framework. By this we do not mean that every problem is a “real life” engineering application, but more that the engineering way of thinking is emphasized throughout the discussion.

The philosophy of this book was formed over the course of many years. My father was a Civil Engineer and surveyor, and he introduced me to engineering ideas from an early age. At the University of Kentucky I took most of the basic Engineering courses while getting a Bachelor’s degree in Mathematics. Immediately afterward, I completed a M.S. degree in Engineering Mechanics at Kentucky.

While working on my Ph.D. in Mathematics at Georgia Tech I taught all of the introductory math courses for engineers. During my education, I observed that incorporation of computation in coursework had been extremely unfocused and poor. For instance during my college career I had to learn 8 different programming and markup languages on 4 different platforms plus numerous other software applications. There was almost no technical help provided in the courses and I wasted innumerable hours figuring out software on my own. A typical, but useless, inclusion of software has been (and still is in most calculus books) to set up a difficult ‘applied’ problem and then add the line “write a program to solve” or “use a computer algebra system to solve”.

At Ohio University we have tried to take a much more disciplined and focused approach. The Russ College of Engineering and Technology decided that MATLAB should be the primary computational software for undergraduates. At about the same time members of the Department of Mathematics proposed an 1804 project to bring MATLAB into the calculus sequence and provide access to the program at nearly all computers on campus, including in the dorm rooms. The stated goal of this project was to make MATLAB the universal language for computation on campus. That project was approved and implemented in the 2001-2002 academic year.

In these lecture notes, instruction on using MATLAB is dispersed through the material on numerical methods. In these lectures details about how to use MATLAB are detailed (but not verbose) and explicit. To teach programming, students are usually given examples of working programs and are asked to make modifications.

The lectures are designed to be used in a computer classroom with students working MATLAB

examples during the lecture or with students reading the notes and working the examples after a brief introduction. At Ohio University we have had good success with this Lecture/Lab format.

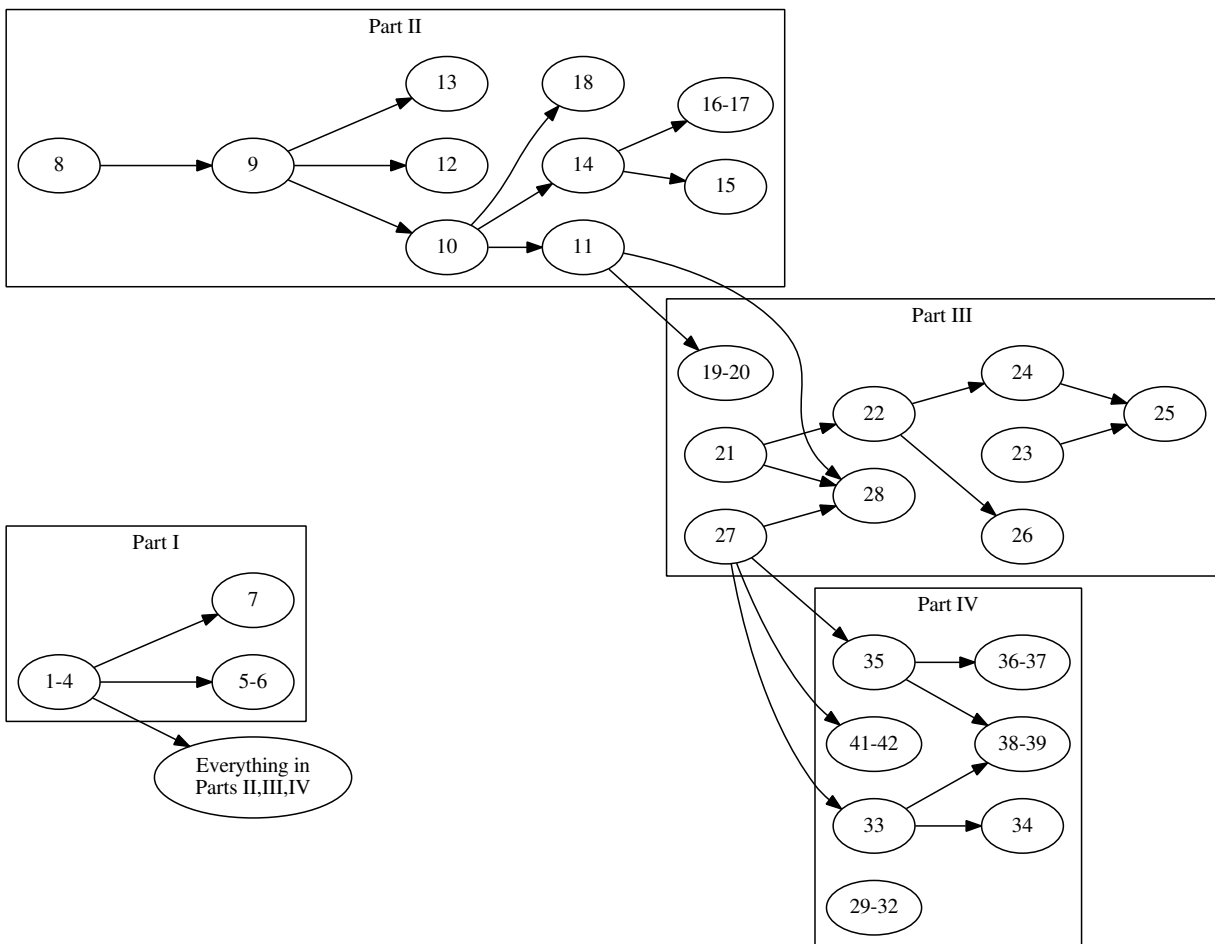
The lectures are divided into four Parts with a summary provided at the end of each Part. Typically we will give an exam covering each of Parts I, II, and III and a comprehensive final exam.

The exercises grow in complexity as the students build their programming skills. At Ohio University we ask students to complete the exercises in groups of 2-3 students and this accounts for a significant portion of the grade (e.g. 30%).

Todd Young

Dependencies

Below we give the dependencies between Lectures. Almost everything depends on Lectures 1–4, so those links are omitted to reduce clutter. Some lectures, marked with * in the table of contents, have not yet been developed.



Contents

Preface	iii
I Matlab and Solving Equations	1
Lecture 1. Vectors, Functions, and Plots in MATLAB	2
Lecture 2. MATLAB Programs	6
Lecture 3. Newton's Method and Loops	10
Lecture 4. Controlling Error and Conditional Statements	14
Lecture 5. The Bisection Method and Locating Roots	18
Lecture 6. Secant Methods	22
Lecture 7. Symbolic Computations	25
Review of Part I	29
II Linear Algebra	33
Lecture 8. Matrices and Matrix Operations in Matlab	34
Lecture 9. Introduction to Linear Systems	39
Lecture 10. Some Facts About Linear Systems	43
Lecture 11. Accuracy, Condition Numbers and Pivoting	46
Lecture 12. LU Decomposition	50
Lecture 13. Nonlinear Systems - Newton's Method	53
Lecture 14. Eigenvalues and Eigenvectors	57
Lecture 15. Vibrational Modes and Frequencies	60
Lecture 16. Numerical Methods for Eigenvalues	62
Lecture 17. The QR Method*	66

<i>CONTENTS</i>	vii
Lecture 18. Iterative solution of linear systems*	68
Review of Part II	69
III Functions and Data	73
Lecture 19. Polynomial and Spline Interpolation	74
Lecture 20. Least Squares Fitting: Noisy Data	78
Lecture 21. Integration: Left, Right and Trapezoid Rules	82
Lecture 22. Integration: Midpoint and Simpson's Rules	87
Lecture 23. Plotting Functions of Two Variables	91
Lecture 24. Double Integrals for Rectangles	94
Lecture 25. Double Integrals for Non-rectangles	98
Lecture 26. Gaussian Quadrature*	101
Lecture 27. Numerical Differentiation	102
Lecture 28. The Main Sources of Error	106
Review of Part III	110
IV Differential Equations	117
Lecture 29. Reduction of Higher Order Equations to Systems	118
Lecture 30. Euler Methods	122
Lecture 31. Higher Order Methods	126
Lecture 32. Multi-step Methods*	129
Lecture 33. ODE Boundary Value Problems and Finite Differences	130
Lecture 34. Finite Difference Method – Nonlinear ODE	134
Lecture 35. Parabolic PDEs - Explicit Method	137
Lecture 36. Solution Instability for the Explicit Method	142
Lecture 37. Implicit Methods	145

Lecture 38. Insulated Boundary Conditions	149
Lecture 39. Finite Difference Method for Elliptic PDEs	154
Lecture 40. Convection-Diffusion Equations*	157
Lecture 41. Finite Elements	158
Lecture 42. Determining Internal Node Values	162
Review of Part IV	165
V Appendices	169
Lecture A. Glossary of Matlab Commands	170