Program Cover Document -MAT 341: Computational Mathematics

I. Basic Course Information

MAT 341: Computational Mathematics is an upper-level course. It has two 80-minute meeting periods each week. The prerequisites are MAT 200 (Proof Writing through Discrete Mathematics), MAT 205 (Linear Algebra: Theory and Applications) and CSC 220 (Computer Science I: Computational Problem Solving) or CSC 250 (Accelerated Computer Science I, II). It is an option for all majors in the Department of Mathematics and Statistics and is one of two ways students in the Applied Mathematics specialization can fulfill their second computer programming requirement.

Computing is an essential part of modern mathematics. The partnership of applied mathematics, mathematics, and computational mathematics brings the tools of modeling, simulation, and data analysis to bear on real-world problems, producing solutions with the power to predict and explain complex phenomena. Computational methods are used in a wide variety of areas in mathematics, computer science, business, engineering, the natural sciences, and the social sciences. As a result, Computational Mathematics combines the beauty and logic of mathematics with the application of computing to solve mathematically modeled problems.

This course will help students develop the computational skills required to solve real-world problems. Significant work on topics drawn from core courses in mathematics that students have taken will be covered, but from a computer solution point of view. Students completing the course will be well prepared for the following opportunities:

- More advanced undergraduate study of computationally based mathematical topics
- Further training in professional masters or doctoral programs in applied mathematics and mathematics.
- Careers that require the ability to integrate computation and mathematical skills.

The goal of computational mathematics, put simply, is to find or develop algorithms that solve mathematical problems computationally (i.e. using computers). There are multiple programming languages, including but not limited to C++, Java, Python, Mathematica, Matlab, Maple, SAS, R and Excel/VBA, that facilitate solving computational mathematics problems. The choice of the best, or most appropriate, software platform in which to do programming should be completely determined by the applications being studied and the intended student audience. The primary focus of MAT 341 is on the mathematical algorithms and their implementation, and not on learning additional computer languages.

For Undergraduate Bulletin: Computational Mathematics combines the beauty and logic of mathematics with computing. In Computational Mathematics, students will learn how to develop and implement mathematical algorithms that can be utilized to solve real-world problems in many disciplines. Much of the course content will draw on topics from earlier mathematics courses, but these topics will be covered from a computer solution point of view.

II. Learning Goals

The primary learning goals of this course are for students to a) learn how to make a computer either solve a mathematical problem; b) gain insights through simulation into how one might solve a problem; c) use computation to gather data to help formulate and refine a mathematical conjecture; and d) understand how the computational complexity of an algorithm affects the usefulness of a particular computational approach. This course will build on mathematical topics students have been taught in core courses within their major from a theoretic approach and reexamine these same topics from a computational/algorithmic point of view. Specific objectives for the course are:

1. Students should gain an appreciation for the role of computers in mathematics, science, engineering and economics as a complement to analytical and experimental approaches.
2. Students should develop a knowledge of numerical approximation techniques, know how, why, and when these techniques can be expected to work, and have ability to program numerical algorithms
3. Students should have learned what computational mathematics is about: designing algorithms to solve scientific problems that cannot be solved exactly; investigating the robustness and the accuracy of the algorithms and/or how fast the results from the algorithms produce solutions. These items includes a basic understanding of computer arithmetic and round-off errors, how to avoid loss of significance in numerical computations, and computational complexity.
4. Students should be able to use and evaluate alternative approaches and algorithms for the solution to a computational problem, including the use of recursive algorithms, iterative algorithms, and where appropriate, closed form solutions.
5. Students should appreciate and demonstrate skills in oral, written and graphical communication, and know the importance of each.

The specific content goals of the course will be determined by the instructor, but it is expected that many of the following topics will be covered in the course:

- Functional and imperative programming languages.
- Functions and Programming; Differences between programming recursive, iterative, and closed form functions; Subroutines, user-defined Objects
- Computational Complexity
- Lists and Sets
- Recursive and Iterative Algorithms
- Numerical Algorithms and Accuracy. Specifically, Newton's methods, calculation of eigenvalue/vectors, determinants, solving differential equations, Euler's methods,
- Graph and Tree Algorithms. Specifically, Euler and Hamiltonian circuits, Traveling Salesperson problem, Greedy algorithm, Sorted edges methods, Kruskal's method.
- Search and Sorting Algorithms. Insertion, Bubble, Quick, and Heap Sorts; Bisection Algorithm
- Time Series Analysis

III. Student Assessment

To assess student understanding of the mathematical and computing topics covered in the course, feedback will be given to students through any of the following mechanisms: commented and/or graded homework, projects, computer programs, examinations, and in-class work.

IV. Learning Activities

In-class learning activities include lectures on mathematical and computer science concepts, discussion, group work, and instruction on programming language syntax and programming techniques. The course will primarily be project based and assignments will be made on each topic covered in the course that involve theory based work, paper and pencil computational work and significant computer programming. Specific activities and work will include the following: 1) Assignments based on each major topic, including written work, programming and in some cases oral presentation; 2) Written and/or oral examinations; 3) As a final assessment tool, either an individual or small group project to be completed, or a formal final cumulative examination will be administered.

Mat370/Computational Mathematics

Grading Policy (Spring 2016)

Course Title: Mat370/Computational Mathematics

Instructor: Dr.Edward Conjura

Text: Course notes and references provided by Dr. Conjura

Meeting Times & Places : Spring 2016 Semester
Section 01 -M/R 11:00AM-12:20PM
Room SC-P201
W 10:00AM-10:50AM
Room SC-P230

SC-P = Science Complex-Math/Physics Wing

Mat370/Computational Mathematics

Who Should Take This Course?

This course is an elective for majors and minors in the Department of Mathematics and Statistics. Please check the requirements for your specialization to determine how credit for the course will be assigned.

Students in this course should have the mathematical maturity of someone who is at least in their fourth semester as a Mathematics, Applied Mathematics, Mathematics Education or Statistics specialization. However, the more math you know the better. In particular, at a minimum, topics covered in this couse will build on those seen in Mat127, Mat128, Mat200 and Mat205.

It is also assumed that anyone taking this course has prior computer programming experience and skills at least at the level of those found in CSC220

This course will use two programming platforms. They are Excel/VBA and Mathematica. Excel/VBA will be used to explore data analysis methods typically used in financial mathematics and other areas that analyze sequences of data. Mathematica will be used as a platform to explore many of the mathematical topics covered in the core courses required

of the various specializations available in the Mathematics and Statistics Department in addition to topics in sequential data series analysis.

Course Philosophy/Main Goals and Objectives

The main focus/goal of this course is to teach students how to write computer programs to solve mathematical problems. Computer Programming is at the heart of this course. Correctness and efficiency of algorithms will be focused on quite a bit. A computer program that runs and solves a problem in seconds is much better than one that takes minutes, hours, days, years, or centuries. Data Structures that are used to store information in a computer will also be covered. Choosing the right data structure is very important and may make the difference in spending 10 hours to write a computer program that solves a problem rather than spending 100 hours by using a different data structure.

Learning how to write computer programs is similar to learning a new foreign language, only harder. If you were studying Spanish you would learn the vocabulary of that language. You would also learn the rules of grammar for the language and then you would combine the skills you have in these two areas to read, write and speak Spanish. The languages you will be learning are Excel/VBA and Mathematica. However, it could have been Java, C++, Visual Basic or any other computer language. Since your goal is to speak these languages to a computer in order to solve a mathematical problem, you will first need to learn the vocabulary and rules of grammar for the languages. Since you already know how to speak a computer langauge like Java or C++, the learning curve for picking up a new language should be short.

As with any computer language, your main objective is to use it as a tool in problem solving. Therefore, you need to describe a method that you will be using in order to solve the problem. Such methods are called algorithms. Once an algorithm has been designed that you will use to solve a particular problem, then you will need to translate the steps of the algorithm into the computer language and have the computer execute it to hopefully solve the problem. Needless to say, doing all of this is not always easy and sometimes things are lost in translation.

The way you learned English as a child was to hear others people speak it and explain it to you. I find that one of the best ways to learn how to speak a computer language is to look at programs that others have written. Once you have learned the basics by looking at the work of others, you will then begin to write your own programs. The computer will do what you tell it to, and nothing more or less. A good philosophy to follow is to never assume that what you have told it to do is correct and so you must constantly monitor what the computer is doing so that

you can determine whether the instructions you have given it are correct or not.

Some people describe computer programming as being as much of an art as a science. For most if not all of you, learning to become a good programmer will require a great deal of hard work and patience. Computers can be frustrating. They don't always explain to you what the problem is in a way that is helpful to your understanding of the problem. In many ways dealing with a computer is like dealing with a baby who has not learned to communicate with adults yet. However, in the end it is what you have told them to do that is causing the problem and so remember that when you point your finger at the computer to assign blame, three fingers are pointing back at you!

Syllabus: The following is a partial list of topics to be covered in this course:

1. Learning Excel/VBA and applying it to Sequential Data
2. Getting Started in Mathematica
3. Functions and Programming
4. List/Sets
5. Numerical Problems
6. Ordinary Differential Equations
7. Linear Algebra meets Differential Equations
8. Some Number Theory
9. Elementary Number Theory meets the Fibonacci Numbers
10. Searching and Sorting
11. Graphs, Trees and Classic Problems
12. Other topics will be explored based on time and interests.

Grading

This course is graded on a "letter grade" basis and your grade will primarily be determined by your ability to write computer programs that solve mathematical problems. You will also be required to do written homework and give oral presentations to explain your work. You will also have one-on-one meetings with me during which you will be expected to explain your work and demonstrate the outcome of your efforts.

You will also be keeping all of your work in one place, called a Portfolio. I will be looking at your work and providing

you feedback and eventually a grade. You should start buy getting yourself a three ringed binder and some tab seperators in which you will store your portfolio. As the semester goes on you may need to get a bigger binder and more tabs.

Attendance:

I personally believe that it is impossible to do well, or even pass this course, without attending class lectures and recitations.

I also realize that sometimes events occur that may result in you missing or being late for a class. We all get sick, and sometimes things come up that require a change in schedule. If you provide me with an acceptable reason for missed time, I will accept it.

However, as the following policy makes clear, I do reserve the right to lower your final grade for excessive unexcused absences or tardiness due to your lack of regular participation in the course.

Three or more unexcused absences could result in a drop in grade by as much as one letter. The same is true for five or more times being late to class without a valid excuse. An unexcused absence will also count as you being late. For example, if you are late three times and absent twice without acceptable excuse, your course grade could be lowered. However, if you are absent twice and late only once without acceptable excuse, it would not be since your late total would only be three.

If you are sick, a doctors excuse may be required. Not feeling well and stopping by the college infirmary will generally not be accepted as a reason for an excused absense. You must also notify me as soon as possible if you believe you will be unable to complete an assigned task. If possible, a makeup date should be set within 24 hours of the date and time of the event. If an assignment is graded and returned and it is not made up by then, in general a makeup will not be given.

Late or Missed Work:

Work missed because of an unexcused absence will be marked late, with a percentage of the total credit deducted.

Assignments must be submitted in class unless other arrangements have been established, such as submission through a computer interface.

Work left in a mailbox (electronic or otherwise) or slipped

under a door, etc may not be accepted and will be considered late until submitted in the required manner.

If you miss the beginning of class and walk in at any time during class to hand in work, it will be considered late unless a valid excuse is provided. If you are not in class to hand in work, but give it to someone else to submit, the same applies.

Responsibility For Knowing About Assignments:

Each student is responsible for assignments made by the professor whether or not he/she is present during the class period when assignments are made. Assignments will generally be made through the web. However, there will be assignments made during lectures that do not appear on the web.

Basis for Grading:

This course is project based. I will be giving you written homework assignments and programming projects with due dates assigned to them. They will be collected and graded. There will be three grade breakdown periods. The first will be after the first 5 weeks. The second will be at the end of the first 10 weeks, and the third and final one will be at the end of the semester. At the end of each period cutoffs for A, A-, B+, B, B-, C+, C, C-, D+ and D grades will be posted, along with your totals at the end of each period.

Work Submitted and Grading

Standards for Work Submitted

Your work will be graded on the basis of content (correctness) and other academic and professional standards (academic honesty, timeliness, neatness, organization, presentation style and completeness).

For example, paperwork submitted should be stapled or contained in a binder.

Academic Dishonesty

ACADEMIC DISHONESTY WILL NOT BE TOLERATED. Any case of academic dishonesty will be dealt with according to college policy, with minimal recommended punishment generally being a grade of "F" for the course.

Reference to any outside source must be provided with each assignment. Failure to do so will be considered as plagiarism.

Teamwork

In this course you may be required to work in teams on the computer programming assignments and on homework. You will find that working in teams will improve your learning ability and it will develop interpersonal skills that are essential for success in the "real world".

Regardless of whether you work as an individual or in a group, every student must maintain a personal portfolio of all work produced in this course. You are expected to keep the portfolio up to date for possible inspection by me at any time.

How Final Grades are Assigned

Final grades will be based on a 'curve' but cutoffs will not exceed straight percentage (eg cutoff for an A will be no more than 90% of total points, no more than 80% for a B, 70% for a C, and 60% for a D).