MATH 420/CS 310, Fall 2015

Numerical Analysis

Professor: Carl Toews
Office: Thompson 390H
Phone: x3839

Email: ctoews@pugetsound.edu
Office Hours: MWF 2-3, or by appointment, or drop in if my door is open.

Class Website: math.pugetsound.edu/~ctoews/teaching/fall15/numanal/

Schedule: MTWF 1:00-1:50, Thompson 374 (keep Tuesday 12:30-1:00 free for tests!)

Introduction:

The subject of numerical analysis has traditionally focused on formulating and analyzing algorithms for computing such quantities as matrix inverses, definite integrals, and initial value problems. Analyzing an algorithm typically entails answering two questions: one, how long does it take for the algorithm to terminate, and two, how accurate is its output? Generally, the answers to these questions are connected: the more accuracy one desires, the longer the algorithm usually needs to run. Claims of the form ``the algorithm will take no more than *X* iterations to be at least as accurate as *Y*" are thus very typical in numerical analysis.

While powerful mathematics is often required to formally establish such claims, the real proving ground for these algorithms is the computer. One of the reasons for this is that theory generally lags well behind practice. Many algorithms actually perform much better than their ``worst case" bounds, and it is useful to implement the algorithms and test them on real problems to see if this is the case. A second reason is that many of the computational problems that numerical analysts deal with come from science and engineering, and an algorithm that works well on one problem may not work well on another. The numerical analyst is thus charged with demonstrating the relevance of his or her algorithm to the broader scientific world, and in practice almost always couples theory with numerical examples. A good numerical analyst thus needs to be competent both as a theoretician and as a programmer.

The intimate interdependence of theory and practice has gradually caused traditional numerical analysis to give way to the broader subject of scientific computing. Broadly speaking, scientific computing is the process of combining powerful mathematics with good programming practices and scientific knowledge to solve real-world problems. Since it is rare for a single individual to excel in all three categories, teamwork is the usual paradigm in scientific computing, but it is nonetheless crucial that team members understand the often subtle ways in which the three elements interact.

The goal of this course is to introduce students to some of the techniques, practices, and problems that characterize modern day scientific computing. The course is cross-listed in mathematics and computer science, and will involve elements of both these disciplines, but it is less about disciplinary specificity than about how to solve real-world problems. As such, it will often focus on ``case studies", i.e. real problems from science and engineering, developing theory and techniques as necessary to provide intelligible solutions to these problems. Students will be challenged not just to solve mathematical problems, write clean code, and process real data, but also to write up and discuss their solutions in clear and convincing language.

Course Catalog Description:

Students learn about numerical solutions to linear systems; numerical linear algebra; polynomial approximations (interpolation and quadrature); numerical differentiation and integration. Students also learn about error analysis and how to select appropriate algorithms for specific problems. A portion of the course will be devoted to learning how to implement these ideas on a computer. No formal programming experience is required, but as this a course that explores the interface between mathematics and computer science, students should be prepared both to prove theorems and program computers.

Specific Learning Goals:

After successfully completing this course, you should be able to:

1. identify the four main sources of error in scientific computing
2. write functional, well-documented code in the Python language
3. articulate what a mathematical model is and know how to formulate and work with such models in solving scientific problems
4. write down and analyze algorithms using such basic terms as *convergence, stability,* and *relative error*
5. learn how to produce informative scientific figures
6. learn how to incorporate such figures into well-written and persuasive scientific report

More broadly, this class should improve your capacity to think critically about where, how, and why numerical analysis is used, to become a deft problem solver, and to become an effective technical communicator.

Required Text:

- "Elements of Scientific Computing", by A. Tveito, H. Langtangen, B. Nielsen, and X. Cai.

Other Requirements:

- Laptop capable of running iPython. (I will ask you to bring this class sometimes.)
- A laboratory notebook (blue, hardbound, numbered, 9.5x7.5in, available from the bookstore.)

Grade Distribution:

Homework	10%
Labs	20%
Participation	10%

Project	15%
Exams	30%
Final Exam	15%

The final grading scale is as follows:

> 93.3	A	73.3 – 76.6	C
90.0 - 93.3	A-	70.0 – 73.3	C-
86.6 - 89.9	B+	66.6 – 69.9	D+
83.3 - 86.6	B	63.3 – 66.6	D
80.0 - 83.3	B-	60.0 – 63.3	D-
76.6 - 79.9	C+	< 59.9	F

Provisional Topic List:

Week	Topic
1-2	Integration
3-4	Ordinary differential equations
5-6	Non-linear algebraic equations
7-8	Linear systems
9-10	Interpolation and least squares
11-12	Special topics
13-14	Student presentations

Class Expectations:

I try to structure a classroom that revolves around the twin pillars of *discovery* and *discussion.* I believe that learning is an active process, and that class works best when it functions as an opportunity for guided inquiry, where the "guides" are both the professor and the peer group. In the wider educational community, my approach is labeled "Inquiry Based Learning", or IBL.

There are a number of ways to structure an IBL class, but salient features of this class will include the following:

- Lectures punctuated by hands-on activities
- An emphasis on group work
- Lots of dialogue and discussion
- Student presentations
- A supportive environment in which to take "risks"
- An emphasis on communication, both oral and written
- A need for trust and confidence, both student-student and student-professor

Productive failure is an idea that lies at the root of our approach. When you're trying to learn something, never making a mistake is generally a sign that you're not pushing yourself hard enough. This class should be a safe and supportive space in which to get things wrong. When talking or presenting, you are challenged to work slightly outside of your comfort zone, to volunteer answers when you have a pretty good idea but aren't 100% certain, to risk a conjecture that might turn out to be off the mark. And when you are listening to fellow students talk, you are challenged to pay strict attention, to flag small errors of language or comprehension, and to politely and respectfully help guide one another towards a clearer and truer picture of the matter at hand. Failure is part of the design spec for this class, and it can be hard, but you will not be struggling alone.

Although the spirit of what I'm shooting for with this IBL style class is probably clear, here is a minimalist list of concrete expectations:

- attend class daily
- do all assigned homework
- do all assigned reading, in a timely fashion
- participate actively in class discussions and class group work activities
- volunteer to present solutions on the board
- volunteer answers to questions I pose, and to ask your own questions when you are confused, uncertain, or simply thinking outside the box
- be courteous and supportive of your fellow learners
- help create a classroom that is a supportive, energetic, respectful place to learn.

More broadly, my basic hope and expectation is that you will engage the spirit of Inquiry Based Learning with enthusiasm, openness, and joy (it *is* fun), and help make this class a fun and supportive place in which to learn.

Lastly, a word about goals and outcomes: the goal in IBL is to produce critical thinkers who have a strong, creative command of the subject material. There is ample research evidence to support the IBL model, and I'm happy to share it with you if you'd care to see it. For me, one of the strongest element of IBL is the host of secondary skills you develop almost "for free", including skills in abstract thinking, working from first principles, working with other people, and communicating your ideas. As my colleague Dana Ernst has noted, "All of the secondary skills you will develop in this course are highly valued by society. Whether you become a teacher, a lawyer, an engineer, or an artist, what differentiates you from your competition is your ability to think critically at a high level, collaborate professionally, and communicate effectively."

Details About Class Activities:

Homework:

I will assign and collect homework weekly, generally assigning on Wednesdays and collecting on Tuesdays. On collection days, we will cover solutions to at least a subset of the problems you turn in. I expect student volunteers to present the solutions, and non-presenters to compare these solutions again their own. New or missing material may be added to your own solutions at this point, but I'll ask that you use special "felt-tipped pens" that I provide so as to distinguish between material that you brought to class and material that you picked up during discussion. I'll collect your homework, and grade it for both style and content according to the following scale:

3	All problems attempted; correct answers, strong effort to articulate thought process
2	Most problems attempted; correct solutions, but not a great articulation of thought process
1	Half or fewer of the problems attempted; incorrect and ill-articulated answers
0	Nothing submitted

You are welcome (indeed, encouraged) to work on your homework in groups. You should, however, turn in your own copy of your work, written up in your own words. As with any mathematical writing, homework solutions are to be written out in complete sentences, and you need to make an effort to articulate your thought process, not merely provide "an answer." The lowest homework score will be dropped at the end of the term.

Labs:

In numerical analysis, computers are more than just tools for finding an answer; they are also powerful agents of exploration and understanding. Using a computer as an "experimental laboratory" will be a key idea in this course, and one that we will leverage through regular class computing sessions.

I'll use Python as the main computer language for this class. Most of the class computing sessions will be in the form of iPython Notebooks, a particular Python interface that is well suited to interactive and exploratory work. Part of the computational lab will consist of working through the exercises and coming up with correct code, but another part will consist of thinking, discussing, and writing about the "bigger picture." When you submit a computational lab, you will submit both a code file (generally an iPython Notebook file) and a physical notebook that has your responses to these big picture questions. You should think of this physical notebook as a tool for recording and organizing what you learn via your computer—it is directly analogous to a scientist's laboratory notebook, and serves the same purpose.

Computational labs will generally be done with a partner. You and your partner can turn in one iPython Notebook file, but each of you should submit your own lab notebooks. I will grade each lab on a 0-10 scale, assessing style, correctness, creativity, and attention to expression. The lowest lab score will be dropped at the end of the term.

Participation:

Since active participation by each member of the class is a critical element in the success of the IBL classroom, participation counts as 10% of your total grade in this course. This grade will be evenly apportioned along the following two axes:

- *Presentations of homework solutions*

  Solutions to homework problems will be discussed in detail on Mondays, the day you turn the homework in. I will ask for student volunteers to present these solutions. **At a minimum, you should plan on presenting at least three times over the course of the semester, and at least once between each exam.**

  Some notes on style: though the environment in this class should be informal and friendly, what we're doing is serious business. The presentations made by students are a foundational part of this class, and they should be taken with great seriousness. I ask that the following points be kept in mind during students presentations:

  - ➢ The point of the presentation is not to convince me that you've done the problem, but rather to make the problem and its solution clear to your peers.
  - ➢ When possible, presenters are to write in complete sentences, using proper English grammar.
  - ➢ Presenters should explain their reasoning, not simply show techniques.

- ➢ Fellow students can ask questions at any point, and should pay attention both to what is written on the board and what is said by the speaker. Anything that is unclear or incorrect should be noted and addressed.
- ➢ Since the presentation is directed at students, the presenter should make appropriate eye contact and try to gage how well students are following the argument.

- *Contribute to our Python Page!*
  Most of you will be learning Python for the first time. It can be a tough process, full of "gotchas". Occasionally you will figure out things that are really helpful, and it would be nice to share these things with your peers. Post these things to the Python Page!

- *Other forms of in-class participation*

  Aside from presenting homework solutions, there are a variety of other ways to participate in this class.

  - ➢ Ask questions! Anytime, anywhere, and on any topic. Good options include during student presentations, during my "lectures", within your working groups, before and after class, during office hours, etc.
  - ➢ Answer questions! Sometime I or a student will ask questions, and we'll need answers. If you have ideas, share them!
  - ➢ Bring in numerical analysis tidbits to share with the class. (Factoids, newspaper articles, a summary of how numerical analysis played a role in a science seminar, whatever.)

***Project*:**

There will be one project assigned roughly mid-way through the class. The project will involve independent research, analysis, coding, and a polished, professional looking paper. The purpose of the project is not just to expand your exposure to topics in numerical analysis, but also to give you some research practice and the opportunity to hone your scientific writing. A portion of your project grade will be tied to an in-class presentation that you give the final week of class.

***Exams**:*

I will give two tests during the semester. Each test will have two parts, a take-home part and an in-class part. The take home parts will involve some use of the computer, and you'll generally have a couple of days to complete them. The in-class part will be administered during a regular class session (generally Tuesdays), and neither notes nor technology will be allowed. Material for each test will generally be chosen from the material we have most recently studied, but each test is theoretically "cumulative" in the sense that any material we have covered is fair game for any test.

***Final Exam**:*

There will be a cumulative, "mastery-based" final exam administered during our allotted exam slot in finals week, i.e. **Friday, December 18th, 12:00-2:00pm**. All students must take the final exam to pass the class. The exam will have two parts. Part 1 of the final contains basic problems that all students should be able to do to demonstrate a basic knowledge of the course. Completing Part 1 successfully (i.e. getting all problems correct except maybe a couple of minor errors) allows a student to keep his/her preliminary grade, or get at least a C- in the course. Part 2 contains more challenging problems.

Policies:

Attendance:

Built into the philosophy of inquiry-based learning is the idea that we help one another to learn. As a consequence, your daily attendance in class is very important, not just for your own benefit, but for that of your peers.

I will take role every day and verify attendance. You are allowed to miss up to three days with no penalty. After three missed classes, two things happen: 1) every additional absence causes you to lose 10% of your final participation grade, and 2) I reserve the right to drop you from the class.

Late work:

Late homework will be docked at the rate of one point per day (note that since homework is graded on a 0-3 scale, one point is a lot.) Late computer labs will be also be docked at one point per day (since computer labs are graded on a 0-10 scale, this is less grave.) Late exams lose one letter grade per day late (eg. an A becomes a B, a B becomes a C, etc.) Late projects are also docked one letter grade per day. The final exam cannot be turned in late.

Absences:

When I form grades, I will drop your lowest homework and laboratory scores. The idea behind this policy is that you might get sick once in a while and miss an assignment—if that happens, simply think of the missed work as "dropped" and carry on. If you miss more than one homework or computer lab for legitimate reasons (e.g. death in the family, medical emergency, etc.) talk to me and we'll work out something equitable.

If you need to be absent for some planned family or medical reason, you should contact me in advance. If an emergency arises, contact me as soon as possible after the emergency has passed. Student athletes who need to miss class for games should let me know of this as early as possible.

Classroom policies:

You are welcome to bring a cup of coffee or a bottle of water to class, but please eat your meals outside of class. Please turn off your phones and keep your laptops closed, unless we happen to be doing a computer exercise. You can take a bathroom break if necessary, but please make this the exception, not the rule—in general, I don't want people entering and leaving the room during class.

Academic integrity:

It is your responsibility to understand the academic integrity policy of the university. You can find this policy in the Academic Handbook, and it is also available online at:

> http://www.pugetsound.edu/student-life/student-resources/student-handbook/academic-handbook/academic-integrity/.

Not citing other people's work, turning in the same work to satisfy two different classes, citing false information, or plagiarism are all violations of the academic integrity policy. Such violations are taken very seriously, and will be reported if discovered.

Disabilities:

If you have a physical, psychological, medical or learning disability that may impact your course work, please contact Peggy Perno, Director of Student Accessibility and Accommodation, 105 Howarth Hall, 253-879-3395. She will determine with you what accommodations are necessary and appropriate. All information and documentation is confidential.

Classroom Emergency Response Guidance

Please review university emergency preparedness and response procedures posted at www.pugetsound.edu/emergency/. There is a link on the university home page. Familiarize yourself with hall exit doors and the designated gathering area for your class and laboratory buildings.

If building evacuation becomes necessary (e.g. earthquake), meet your instructor at the designated gathering area so she/he can account for your presence. Then wait for further instructions. Do not return to the building or classroom until advised by a university emergency response representative.

If confronted by an act of violence, be prepared to make quick decisions to protect your safety. Flee the area by running away from the source of danger if you can safely do so. If this is not possible, shelter in place by securing classroom or lab doors and windows, closing blinds, and turning off room lights. Lie on the floor out of sight and away from windows and doors. Place cell phones or pagers on vibrate so that you can receive messages quietly. Wait for further instructions.

Getting Help:

Numerical analysis is an advanced class, and there are probably no tutors available to help you. But I'm around: I have regular office hours, and if you can't make those, then try just dropping by or setting up an appointment (I am generally in my office if I'm not teaching, eating, or working out.) Make a habit of coming to see me to discuss any questions you might have. I imagine that one of the reasons you chose to go to a liberal arts college was to gain access to your professors, and I am very much on board with that vision. You should leverage me as a resource.