Final Project: Numerical Analysis

We will be doing a final project instead of a final exam. The goal of this project is to gain a better understanding of the concepts we learned in this class and to practice communicating about math. You can work individually or in groups of up to three people total. If you have multiple people, your final result should be more comprehensive than if you're working alone.

There are three main components that will be evaluated:

(i)	project proposal	(10%)
(ii)	written final report	(50%)
(iii)	project presentation / review of your peers' projects	(40%)

Each aspect will be given a rating "does not meet expectations", "satisfactory", "good", and "excellent". These will roughly correspond to letter grades of F, C, B, and A respectively. Based on your project proposal, I will provide you an estimated rating for your final report, assuming that you complete everything outlined in the proposal.

There are several possibilities for the contents of the project. You should pick *one* of the following:

- Pick a standard topic in numerical analysis that we did not cover in much detail. Find one or more textbooks, notes, or other resources which cover this topic, and do a self-guided study of the topic.
- Find a paper on a research topic in numerical analysis. Read (part of) the paper and try to implement some of the ideas from the paper.
- Something else you are interested in! But please talk to me individually prior to submitting the project proposal if you would like to do something besides what is suggested above.

As a rough guide, you should spend about 2-3 homeworks worth of time on the project.

The last several homeworks will be a bit lighter than the ones at the start of the class to provide you time to work on the project. It's my hope that you use this time to work on the projects, but ultimately you are responsible for properly allocating your time. If you would like help with being held accountable for making progress, feel free to include that in your proposal. I'm happy to regularly meet/check-in with you about your progress.

Proposal (due Mar 30)

The first part of the final project is a project proposal, and each group should turn in a single proposal with all all members tagged.

The proposal serves several purposes. First, it forces you to start thinking about the project earlier than the last minute. Second, it gives me a chance to make sure your project is on the right track to satisfy the requirements. Finally, it will give you an outline to provide structure for the rest of the project.

Since the exact format for the project is very flexible, the project proposals are very important in making sure that your proposed project will be sufficient. Submit to Gradescope 1-2 pages outlining your plan for the project. Make sure to answer all of the following:

- Who is in your group?
- What is the specific topic / what will you do for your project? Try to be specific as possible. It's not expected that you will know everything (because often what you do is conditional on what happens while you are working on the project), but the more details you can provide, the easier it will be for me to give you feedback on if your project is sufficient.
- Describe what you will put in the final report. Again, try to be as detailed as possible. Minimum suggested requirements are listed below to use as a starting point. However, your proposal should contain a more detailed outline of your plan.

Written report (due April 27)

Each group will produce a written report summarizing what you did during your project. The exact requirements for the report are flexible since there are many possible projects. Thus, you should lay out a precise plan for what you will put in the written report in your project proposal. Overall, the project is expected to take the time of several homeworks as it carries 30% of the final grade.

I have listed minimum expectations which you can use as a starting point for your proposal. However, your proposal should contain a detailed description of your plan for the project so I can determine whether it is sufficient.

- If you choose to study a "textbook" topic, I expect a roughly 10 page summary of what you learned; i.e. explaining the topic and how it fits into what we learned in the class in your own words. Some of this may be your solutions to the textbook problems. You must demonstrate that you have understood the topic well.
- If you read a research paper, I expect a 5-10 page summary of the topic. The length depends on whether the paper is mostly theoretical, or whether you are able to implement the ideas yourself (e.g. code them up). You should clearly explain the context for the paper (why the paper is important), what problem the paper aims to solve, and what the techniques in the paper are.

Presentations (May 15, 3:30pm-5:30pm in class)

Each group will prepare a poster on their project. The poster in not meant to be the same as the final report. Rather, the goal of your poster is to *help you* to explain what you did for your project to your classmates. It should not be a wall of text which is very hard to read.

Try to think about how to simplify complex mathematical ideas into something which is understandable to people with less background in the area. Remember, you've spent more time thinking about your project than anyone else has, so some things which you might think are "obvious" may actually be something people might have questions about.

Posters should be modeled after "research posters". NYU has some general guidelines here: `https://guides.nyu.edu/posters`.

Minimum requirements:

- descriptive title
- names of group members
- clear description of the problem you were trying to solve or your goals with the project, explanation of your methods/techniques, and summary of your results
- at least 18x24in

Blank poster boards are available for several dollars at most convenience stores or art shops. You can then print text onto standard letter paper, and attach it to the board. You're not expected to have your poster printed by a poster shop, as this can be fairly expensive.

On the day of presentations, you will alternate between standing with your poster to present it and going to see other groups' posters. You will write a brief review on several of the projects which you visited, and a form to assist with this will be provided to you on the day of the presentations.

Project Topics

Below I've put some potential topics with brief descriptions. Note that these are *topics*, not projects. You must decide on a project yourself and write the proposal based on what you want to do. This will require a bit of preliminary research to see if you have the background for a topic and whether there are good resources. If you are unsure whether a project is suitable, feel free to reach out to me to discuss.

More on stability and rounding error

What: In class we only very briefly touched on how one might analyze the stability of algorithms in finite precision arithmetic. The stability analysis of some basic algorithm (e.g. summation, matrix-vector products, matrix-matrix products, vector matrix vector, etc.) is within reach given what we covered in class.

Background: Understanding of the model of floating point arithmetic.

Resources: There are a number of good textbooks. The most standard is probably Accuracy and Stability of Numerical Algorithms by Nick Higham.

More topics: There is a fair amount of ongoing research in this area. Some active areas of researach include using multiple precisions within a computation and randomized rounding.

More on QR factorization

What: We saw the Gram–Schmidt and Modified Gram–Schmidt algorithms for computing a QR factorization. Another standard algorithm is the Householder QR algorithm. This one is even more stable than MGS!

Background: Basic linear algebra

Resources: Most textbooks including Numerical Linear Algebra by Trefethen and Bau.

More topics: There is some ongoing research about parallel and communication avoiding variants of QR algorithms which are suitable for use on supercomputers.

Krylov subspace methods

What: Krylov subspace methods are a powerful class of algorithms for solving linear systems of equations and eigenvalue problems as well as things like computing matrix functions. They work by building the subspace

$$\operatorname{span}\{\mathbf{b}, \mathbf{A}\mathbf{b}, \ldots, \mathbf{A}^{k-1}\mathbf{b}\},$$

which can be done accessing $\mathbf{A}$ only through matrix-vector products.

For solving large sparse linear systems, algorithms like conjugate gradient (CG), MINRES, and GMRES are among the most widely used algorithms in practice. These are standard algorithms covered in a grad course on numerical analysis.

Krylov subspace methods are also commonly used for finding eigenvalues and eigenvectors of matrices. We've already seen one example: the power method, but nobody uses the power method in practice. However, other related algorithms such as the Lanczos method and variants such as implicitly restarted Lanczos are used in practice.

Background: Understanding of induction

Resources: The book Krylov Subspace Methods by Liesen and Strakos is a great introduction. In addition, the recent survey 70 years of Krylov subspace methods: The journey continues provides a nice overview of some active research topics.

More topics: First order optimization methods for least-squares problems (gradient and accelerated gradient descent) are equivalent to least squares problems when solving least squares problems or positive definite linear systems. The series of block posts On the Link Between Polynomials and Optimization discuss this connection.

More on approximation

There are many topics in approximation that we won't cover in much detail. A great starting point is Approximation Theory and Approximation Practice by Trefethen.

Remez's algorithm

What: Remez's algorithm is a method to find the best polynomial approximation to a function. Implement it and understanding it's convergence guarantees.

Resources: Wikipedia, Barycentric-Remez algorithms for best polynomial approximation in the chebfun system

Splines

What: Splines are piecewise approximations to some function. These can be used to approximate integrals, and derivatives too.

Rational functions

What: We focus mostly on polynomial approximation theory. There is a lot of theory on approximation with rationals.

Other topics in approximation

- Impossibility of Fast Stable Approximation of Analytic Functions from Equispaced Samples: This paper shows there is an inherent tradeoff between complexity and stability for *any* approximation algorithm which uses equally spaced nodes.

Randomized Numerical Linear Algebra (RandNLA)

Randomized algorithms are increasingly common in all areas of computing. Typically, randomized algorithm are often faster than their deterministic counterparts, but pay for this with a small change of complete failure.

In numerical linear algebra, the use of randomness is increasingly important. This survey provides a good starting point: Randomized Numerical Linear Algebra: Foundations & Algorithms by Martinsson and Tropp.

Randomized Kaczmarz

What: We have seen several algorithms for solving $\min_{\mathbf{x}} \|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2^2$ when $\mathbf{A}$ is $m \times n$. If $n \leq m$ and the system is consistent, then there are at most n important equations (the rest are dependent). Thus, we might hope to have an algorithm which can work without even accessing all of $\mathbf{A}$. The Randomized Kaczmars algorithm is such a method (and is equivalent to stochastic gradient descent (SGD) in a certain sense).

Background: probability (random variables, expectation, variance, concentration inequalities). Some familiarity with gradient descent is also useful.

Sketching / Randomized SVD

What: Computing a rank-k truncated SVD of a $m \times n$ matrix usually takes time $O(mnk)$. Sketching is a powerful technique in randomized numerical linear algebra which allows us to efficiently learn the important singular components of $\mathbf{A}$ in time $O(mn\log(k) + (m+n)k^2)$.

Background: probability (random variables, expectation, variance, concentration inequalities).

Resources: The survey Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions by Halko Martinsson and Tropp is probably the most famous paper on RandNLA. It's quite long, but there are a number of very accessible sections which describe the basic algorithms and provide intuition for how they work.

Trace estimation

What: A number of applications require computing $\text{tr}(\mathbf{A})$, for some $n \times n$ matrix $\mathbf{A}$. If we have $\mathbf{A}$ in hand, this is easy: just read off the diagonal in $O(n)$ time. However, we often can only access $\mathbf{A}$ via matrix-vector products; that is, we pass a vector $\mathbf{v}$ into some black-box, and get out a vector $\mathbf{A}\mathbf{v}$. One can use such a model to compute the trace exactly using n matrix-vector products (think about how), but randomness can let us get an approximation to the trace using far fewer matrix-vector products.

Background: probability (random variables, expectation, variance, concentration inequalities).

Resources: The Martinsson Tropp survey listed above has a chapter on trace estimation. The paper Hutch++: Optimal Stochastic Trace Estimation has two Tandon authors and is probably the most well-known paper on trace estimation recently. This relies heavily on sketching, so at least some background on that topic is needed.

More on Gaussian Elimination

What: We discussed partial pivoting, but there is an even more stable scheme called complete pivoting. There are also more intermediate schemes like rook pivoting. Some of these schemes have probably small growth factors, although there are some big conjectures about what growth factors are actually possible.

Some papers:

- Some New Results on the Maximum Growth Factor in Gaussian Elimination: This paper talks about the famous conjectured growth rate, some numerical experiments showing the conjecture is false, and some new bounds.

More topics: There is still some research on this topic, including incomplete LU/Cholsesky factorizations (and the use of randomized pivoting schemes).

Strassen's method and fast matrix multiplication

What: The standard algorithm for multiplying two $n \times n$ matrices requires $O(n^3)$ operations. In 1969 a new algorithm which runs in time $O(n^{2.8074})$ was introduced by Strassen. Since then, a number of improvements to the exponent have been made. Many people believe it is possible to do this problem in time $O(n^{2+\epsilon})$ for any $\epsilon > 0$.

Background: Familiarity with recursive algorithms

Resources: Wikipedia