HOMEWORK
David F. Gleich
November 25, 2025

Homework 6

Please answer the following questions in complete sentences in a clearly prepared manuscript and submit the solution by the due date on Gradescope. The homework is due 11:59pm Dec 1st due to Purdue regulations around quiet period. You have an automatic extension to the morning of December 5th. You need to have it in by December 3rd to be guaranteed to have it graded before the exam. We may discuss solutions on December 5th, so no homework can be submitted after that day. (If the class makes a motion to avoid discussing homework in class, I can give you an extra day or so if you'd like more time.)

Remember that this is a graduate class. There may be elements of the problem statements that require you to fill in appropriate assumptions. You are also responsible for determining what evidence to include. An answer alone is rarely sufficient, but neither is an overly verbose description required. Use your judgement to focus your discussion on the most interesting pieces. The answer to "should I include 'something' in my solution?" will almost always be: Yes, if you think it helps support your answer.

Problem 0: Homework checklist

- Please identify anyone, whether or not they are in the class, with whom you discussed your homework. This problem is worth 1 point, but on a multiplicative scale.
- Make sure you have included your source-code and prepared your solution according to the most recent Piazza note on homework submissions.

Problem 1: Experimenting with Lanczos-based methods.

1. Implement a Lanczos-based MINRES code that explictly builds $\boldsymbol{V}_k, \boldsymbol{T}_k$ and then finds the minimum residual vector within the Krylov subspace.
2. Compare the first 25 residuals from the Lanczos-based CG code we wrote in class that explictly builds $\boldsymbol{V}_k, \boldsymbol{T}_k$, with the a standard implementation of CG from: http://www.cs.purdue.edu/homes/dgleich/cs515-2025/homeworks/cg.jl for the linear system

   ```
   n = 100
   on = ones(Int64,n)
   A = A = spdiagm(-1=>-2*on[1:end-1],0=>4*on,1=>-2*on[1:end-1])
   b = ones(n)
   ```

   as well as the MINRES code you developed above.
3. Using the `cg.jl` function, look at how many iterations are required for CG to converge to a tolerance of 10^{-8} for the matrix in the last part. Determine how this scales with n.

Problem 2: Orthogonality of Lanczos

Let $\lambda_1 = 0.1$ and $\lambda_n = 100$, $\rho = 0.9$, $n = 30$.
Consider the n-by-n matrix with diagonal elements

$$d_i = \lambda_1 + (i-1)/(n-1)(\lambda_n - \lambda_1)\rho^{n-i}.$$

(Some call this the Strakoš matrix.)

1. Use the Lanczos method starting from a a random vector $\mathbf{v}_1$ and the vector $\mathbf{v}_1 = \mathbf{e}/\sqrt{n}$ and then plot the quantity $\log(\|\boldsymbol{V}_k^T\boldsymbol{V}_k - \boldsymbol{I}\| + 10^{-20})$ for $k = 1$ to 30. Describe what you SHOULD find and what you actually find. Do your results depend on the starting vector?
2. Plot $\log(|\mathbf{v}_1^T\mathbf{v}_k| + 10^{-20})$ for the $k = 1$ to 30. Also plot $\log(|\mathbf{v}_{k-2}^T\mathbf{v}_k| + 10^{-20})$ for $k = 3$ to 30.
3. What is β_{31}? What should it be?
4. Plot $\log(\|\boldsymbol{A}\boldsymbol{V}_k - \boldsymbol{V}_{k+1}\boldsymbol{T}_{k+1}\| + 10^{-20})$ for $k = 1$ to 60.

Problem 3: Krylov methods

1. Suppose that $\boldsymbol{A} = \boldsymbol{I} + \mathbf{v}\mathbf{v}^T$ is non-singular. And suppose we use MINRES to solve the system $\boldsymbol{A}\mathbf{x} = \mathbf{b}$. In exact arithmetic, how many steps will MINRES take to converge to the exact solution? Be sure to explain *how you determine what the answer is. Note, your result should be self contained and not utilize the theory discussed in class – this is good practice for the final; if you do use a theorem to justify your answer, you may lose a few points.*
2. Compare the work involved with the method in step 1 to methods that would involve estimating $\mathbf{v}$ and then using a specialized solver explicitly for this type of structure.
3. It turns out that the GMRES solution to $\boldsymbol{A}\mathbf{x} = \mathbf{b}$ in step k, $\mathbf{x}^{(k)}$, can be viewed as constructing a degree k polynomial, $q(x)$, such that $||q(\boldsymbol{A})\mathbf{b}||_2$ is minimized. Think of it this way: since $\mathbf{x}^{(k)}$ is in $\mathbb{K}_k(\boldsymbol{A}, \mathbf{b}) = \text{span}\{\mathbf{b}, \boldsymbol{A}\mathbf{b}, \ldots, \boldsymbol{A}^{k-1}\mathbf{b}\}$., we know $\mathbf{x}^{(k)} = c_0\mathbf{b} + c_1\boldsymbol{A}^1\mathbf{b} + \ldots + c_{k-1}\boldsymbol{A}^{k-1}\mathbf{b}$ for some scalars c_j. If we define the polynomial $q(x) = c_0 + c_1x^1 + \ldots + c_{k-1}x^{k-1}$, then we can express $\mathbf{x}^{(k)} = c_0\boldsymbol{I}\mathbf{b} + c_1\boldsymbol{A}^1\mathbf{b} + \ldots + c_{k-1}\boldsymbol{A}^{k-1}\mathbf{b} = q(\boldsymbol{A})\mathbf{b}$.

   Then since GMRES constructed $\mathbf{x}^{(k)}$ to minimize $||\mathbf{b} - \boldsymbol{A}\mathbf{x}^{(k)}||_2$, we know that the polynomial $q(x)$ is the degree $k-1$ polynomial that minimizes $||\mathbf{b} - \boldsymbol{A}q(A)b||_2$, since $\mathbf{b} - \boldsymbol{A}\mathbf{x}^{(k)} = \mathbf{b} - \boldsymbol{A}q(A)\mathbf{b}$.

   Suppose GMRES converged to the exact solution $\mathbf{x}$ to the system $\boldsymbol{A}\mathbf{x} = \mathbf{b}$ in t iterations, i.e. it constructed $\mathbf{x}^{(t)}$ in $\mathbb{K}_t(\boldsymbol{A}, \mathbf{b})$ such that $\mathbf{x}^{(t)}$ is the exact solution to $\boldsymbol{A}\mathbf{x} = \mathbf{b}$. Show that there exists a degree t polynomial $p(x) = -1 + \sum_{j=1}^{t} p_j x^j$ such that $p(A)\mathbf{b} = 0$
4. When we derived the form of CG from orthogonal polynomials, we started from the goal that we wanted the residuals to be orthogonal. Consider the Lanczos perspective on CG. Solve $\bar{\boldsymbol{T}}_k\mathbf{y}_k = \|\mathbf{b}\|\mathbf{e}_1$ and let $\mathbf{x}^{(k)} = \boldsymbol{V}_k\mathbf{y}_k$. Show that the residuals from this perspective *are indeed* orthogonal.

Problem 4: Solving non-symmetric systems.

Note that there are few ways to turn a non-symmetric linear system into a symmetric linear system. The first is to use the normal equations $\boldsymbol{A}^T\boldsymbol{A}\mathbf{x} = \boldsymbol{A}^T\mathbf{b}$. The second is to use a set of augmented equations:

$$\begin{bmatrix} \boldsymbol{I} & \boldsymbol{A} \\ \boldsymbol{A}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{r} \\ \mathbf{x} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ 0 \end{bmatrix}.$$

1. Show that the solution of the augmented system of equations exists for any square, full-rank non-symmetric matrix $\boldsymbol{A}$.
2. Try and use CG and MINRES to solve the Chutes and Ladders linear system from the beginning of class using these approaches. Do they work? If so, how many matrix-vector products do they take compared with your Neumann series-based solver to converge to a 2-norm relative residual of 10^{-8}.
3. Also report on the same measures for the PageRank system from Homework 2 on the Wikipedia matrix.

Problem 5: Using GMRES with Preconditioners

For this problem, you must use Julia.

It's been updated now, it should be good.

It's based on an assignment from Michael Saunders' iterative methods class.

Download `poisson2D-data.csv` and `poisson2D-rhs.csv` and make sure you can load them via the following script. These files were originally MAT files from https://sparse.tamu.edu/FEMLAB/

```
using DelimitedFiles
using SparseArrays
data = readdlm("poisson2D-data.csv",',')
A = sparse(Int.(data[:,1]), Int.(data[:,2]),(data[:,3]))
A = (A + A')./2
b = vec(readdlm("poisson2D-rhs.csv"))
```

The matrix A is symmetric-positive definite. But we are going to look at solving $(\boldsymbol{A} - \sigma \boldsymbol{I})\mathbf{x} = \mathbf{b}$ where $\sigma = 1.7 \times 10^{-2}$. We'll declare a method converged if the relative residual has 2-norm less than 10^{-6}.

1. Plot the relative residuals for 100 iterations of the MINRES (`minres` from `Krylov.jl`) method and the GMRES (`gmres` from `Krylov.jl`) method restarted every 30 iterations on a semilogy scale. Describe what you observe, in particular, do the methods converge? For convenience, here is some code to help you get started:

   ```
   using Krylov
   x, hist_min = minres(A,b; rtol = mytol,  itmax = mymaxiter)
   x, hist_gm = gmres(A,b;restart = true, memory = myrestart, rtol = mytol, itmax = mymaxiter)
   # look at hist_min.data[:resnorm]
   ```

2. If you use an incomplete cholesky `ichol` or incomplete LU `ilu` preconditioner with GMRES, it will converge. How many iterations does it take? Does MINRES benefit from an incomplete Cholesky preconditioner? We provide you with the `ichol` and `ilu` code. Download the file `precond.jl` from here http://www.cs.purdue.edu/homes/dgleich/cs515-2025/homeworks/precond.jl. There are some peculiarities with the Krylov package. We will help you set that up. Here is the code to help you get started

   ```
   using LinearOperators
   include("precond.jl")
   Lchol = ichol(A)
   M1 = opInverse(LowerTriangular(ichol(A)))
   x, hist_pgm = gmres(A,b; restart = true, memory = myrestart, rtol = mytol, itmax = mymaxiter,M=M1)
   ```

   And preconditioning MINRES:

   ```
   P1 = opInverse(LowerTriangular(ichol(A)))
   ```

```
Mprecond = P1'*P1;
x, hist_pmin = Krylov.minres(A,b,M=Mprecond,itmax=mymaxiter, rtol=mytol)
```

3. Show the eigenvalues of the matrix before and after preconditioning. Do you observe any clustering?

Problem 6: Using Multigrid to solve Poisson's equation

For this problem, if you don't use Julia, you will have to convert a series of multigrid codes to your own language. This isn't too hard as it's only about 100 lines of code. However, in one step, you will have to use a sparse direct solver. This is simple to do in Julia. If you know how to do it in other languages, then this should be easy. However, you are on your own!

Download the codes http://www.cs.purdue.edu/homes/dgleich/cs515-2025/homeworks/multigrid_functions.jl

1. Using these codes, you can solve Poisson's equation as follows:

   ```
   X = solve_poisson_direct(poisson_setup(nx,ny, (x,y) -> 1))
   ```

   and plot a solution via

   ```
   using Plots
   pyplot()
   surface(X)
   ```

   (which is what a previous homework asked you to do!)

   For this problem, we will always have `n = nx = ny`. How long does it take to solve this for $n = 31, 63, 127, 255, 511, 1023$. (This worked great on my computer in terms of memory usage. But your milage may vary, if you run out of memory, show the largest problem you can solve!)

   Plot the times on a log-log scale. If you double the problem size, about how much longer does the computation need?

2. Explain what the `apply_poisson_jacobi` function does. Where is the matrix?

3. Use this method to implement the Jacobi method to solve a linear system? For a problem with $n = 31$, estimate how much decrease in error occurs at each iteration of Jacobi. That is, if $\mathbf{x}^{(k)}$ and $\mathbf{x}^{(k+1)}$ are sucessive iterates with error $e_k = \|\mathbf{x}^{(k)} - \mathbf{x}\|/\|\mathbf{x}\|$ and e_{k+1}, respectively, then what does the trend in e_{k+1}/e_k show? (Note, you probably have to run a few hundred to a few thousand iterations to see these values start to converge.)

4. Write a function `apply_poisson_gauss_seidel` to implement the Gauss-Seidel without building the matrix. Show the same error trend for Gauss-Seidel compared with Jacobi.

5. Estimate the same rate for Jacobi and Gauss Seidel for a problem with $n = 63$. Given that each iteration of Jacobi and Gauss-Seidel takes $O(n^2)$ time, give a rough estimate of the number of iterations needed to reduce the error to a fixed value of ε.

6. Now we get to play around with Multigrid! The function `interpolate` takes a $n = 31$ vector and returns a $n = 63$ vector. Use `interpolate` to evaluate the error that results from solving a $n = 31$ Poisson problem and interpolating the solution to a $n = 63$ solution. Show a mesh-plot of the absolute value of the error and the norm of the error.

7. What happens when you run a Jacobi iteration after interpolating the solution? Describe how the error changes in terms of the mesh-plot as well as the the norm of the error.

8. Explain what the `simple_multigrid` function does. Estimate the decrease in error for one iteration of the loop for $n = 31$ and $n = 63$.

9. Explain what the `apply_poisson_multigrid` function does and how `poisson_multigrid_error` compares with `simple_multigrid`. Estimate the decrease in error for one iteration of the loop for the same values of n.

10. In practice, we can't use the error as we won't have the true solution. Use the `poisson_multigrid_residual` method to solve Poissons equation with 25 iterations for $n = 31, 63, 127, 255, 511, 1023$. Show the times on a log-log scale along with the times from the direct method in part 1. Can you solve a 10,000-by-10,000 problem? (This is a big problem, you'll need O(16GB) of RAM!)

Problem 7: Eigenvalue and singular value estimates.

1. Use the Lanczos method to estimate the top 5 singular values of the Chutes and Ladders iteration matrix.

2. Does the loss of orthogonality of the Lanczos vectors impact the accuracy of the eigenvectors? Describe your investigation here. (A rough guide, you should present evidence that shows you are familiar with the ideas in class. This could be 1 paragraph up to around a page, and could include figures.)

3. Report if your code from (1) is correct for the 5 largest singular values of the sparse matrix loaded directly from the Wikipedia file via the `load_data()` function in Homework 2. So this is the matrix P, not the PageRank system. Provide evidence to support your statements that the code gets the correct answers or explain what you think goes wrong if it does not. (Your explanation is more important than whether or not it works, although you may lose points if it doesn't work due to the size of the problem as the code from 1. should scale to this level.)