Project: Krylov Subspace Methods, Due: Monday, 2/25

This project (report and accompanying code) is due on **Monday, February 25**, by 11:59 PM. Your report should be well-organized, **typed** (using LaTeX or Microsoft Word) and saved as a .pdf for submission on Canvas. You must show all of your work within the report to receive full credit. For portions of the project requiring the use of MATLAB code, remember to also submit your .m-files on Canvas as a part of your completed project. Your code should be appropriately commented to receive full credit.

Project Description

The Jacobi, Gauss-Seidel, and SOR iterative methods for approximating the solution to $\mathsf{A}\mathbf{x} = \mathbf{b}$ are known as *stationary* iterative methods, since each method can be written as

$$\mathbf{x}^{(k)} = \mathsf{T}\mathbf{x}^{(k-1)} + \mathbf{c}$$

where the iteration matrix T is constant and does not depend on the iteration k. Alternatively, *Krylov subspace methods* are nonstationary iterative methods which do not have iteration matrices. Instead, these methods aim to minimize the residual $\mathbf{r}^{(k)} = \mathbf{b} - \mathsf{A}\mathbf{x}^{(k)}$ with respect to the vectors in the *Krylov subspace*

$$\mathcal{K}_k(\mathsf{A}, \mathbf{b}) = \text{span}\left\{\mathbf{b}, \mathsf{A}\mathbf{b}, \mathsf{A}^2\mathbf{b}, \ldots, \mathsf{A}^{k-1}\mathbf{b}\right\}.$$

Two standard Krylov subspace methods are the conjugate gradient (CG) method and the generalized minimal residual (GMRES) method. The CG method is designed for solving systems where the coefficient matrix A is symmetric positive definite, while GMRES works for nonsymmetric systems. The goal of this project is to explore the use of the CG and GMRES iterative methods in approximating the solution to $\mathsf{A}\mathbf{x} = \mathbf{b}$.

Problems

$\boxed{1}$ (14 points) In your own words, briefly describe the main ideas behind the CG and GMRES algorithms. For instance, each method minimizes the residual in a different way, so you'll want to highlight the differences. Your description of each method should be clear and concise, including the key points and defining factors of each method, and should total 1-2 pages in length (no more than 2 pages).

As you research the algorithms, you may notice that Section 7.6 in the text gives a derivation of the CG method, but it does not discuss GMRES or Krylov subspaces, so you'll also want to refer to supplementary sources. For example, you may find the following references useful:

- L. N. Trefethen and D. Bau (1997) *Numerical Linear Algebra* – Lecture 35 on GMRES, Lecture 38 on CG (available online as an ebook through Gordon Library)

- C. T. Kelley (1995) *Iterative Methods for Linear and Nonlinear Equations* – Chapter 2 on CG, Chapter 3 on GMRES (available in hard copy at Gordon Library and online at `https://www.siam.org/books/textbooks/fr16_book.pdf`)

You may use additional references as you find helpful. Cite any references (including the course textbook) you use in a bibliography at the end of your description. For example, if you use [1] as a reference, then you should include the following bibliography entry at the end of your description:

[1] L. N. Trefethen and D. Bau (1997) *Numerical Linear Algebra.* SIAM: Philadelphia.

You should **cite at least 2 references**. (Wikipedia does not count, so please do not cite it!)

$\boxed{2}$ Consider the numerical implementation of the CG and GMRES methods.

(a) (6 points) Explore the built-in MATLAB functions `pcg.m` and `gmres.m` implementing the CG and GMRES methods, respectively. For each function, describe the inputs, outputs, options relating to the stopping criteria, etc. In particular, what are the default settings relating to the maximum number of iterations and tolerance? How do you change these settings if you need to? What does the `p` in `pcg` stand for, and what does this mean in terms of the algorithm? You may use the MATLAB `help` documentation to your advantage.

(b) (4 points) **BONUS:** Write your own MATLAB **function** implementing the conjugate gradient method. Name your function `myCG.m`. Your function should input the matrix A, righthand-side vector $\mathbf{b}$, stopping tolerance, and maximum number of iterations, and it should output the solution vector $\mathbf{x}$ and the number of iterations performed. Demonstrate that your function works by testing it on the system described in Problem 3 below.

$\boxed{3}$ Consider the linear system $\mathsf{A}\mathbf{x} = \mathbf{b}$ where the coefficient matrix A is an $n \times n$ tridiagonal matrix with ones along the main diagonal and $-1/2$ all along the upper and lower subdiagonals. The righthand-side vector $\mathbf{b}$ is an $n \times 1$ vector with $1/2$ as its first entry and zeros as all other entries. For example, when $n = 3$, the system is as follows:

$$\begin{bmatrix} 1 & -\frac{1}{2} & 0 \\ -\frac{1}{2} & 1 & -\frac{1}{2} \\ 0 & -\frac{1}{2} & 1 \end{bmatrix} \mathbf{x} = \begin{bmatrix} \frac{1}{2} \\ 0 \\ 0 \end{bmatrix}$$

(a) (6 points) Show that A is symmetric positive definite when $n = 3$, as above. Is this true for all n? Explain.

(b) (30 points) For $n = 5$, 10, 50, 100, and 500, use MATLAB implementations of the following iterative schemes to approximate the solution to the linear system described, starting with $\mathbf{x}^{(0)} = (0, 0, \ldots, 0)^{\mathsf{T}} \in \mathbb{R}^n$ and stopping when

$$||x^{(k)} - x^{(k-1)}||_\infty < 10^{-6}$$

- Jacobi's method, using `jacobi.m`.
- Gauss-Seidel, using `gauss_seidel.m` from HW4.*
- SOR with the **optimal** choice of ω, using `SOR.m` from HW5.*
- CG, using MATLAB's `pcg.m`. (Note that the default settings will not be sufficient, so you will need to modify them in order for the method to converge.)
- GMRES, using MATLAB's `gmres.m`. (You will also need to modify the default settings for this method to converge.)

* **Note:** You'll want to make sure your `gauss_seidel.m` and `SOR.m` functions are working correctly from the previous homework assignments!

For each n, use the numerical solution obtained by the command `A\b` as the "true" solution, and report the number of iterations it took each of the methods to converge (if they converge at all). Check that the stationary iterative methods will converge beforehand by computing $\rho(\mathsf{T})$ – report these values for each stationary method for each n. Please do **NOT** include the resulting solution vectors in your project write-up, as they will be too large to look at for $n > 10$.

Summarize your results in a table (or multiple tables) organized in a clear and concise fashion, and discuss your findings. In particular, discuss how the number of iterations and computational time for each method to converge changes with n. For the SOR method, does the optimal choice of ω change with n? Discuss.

Note: For any of the above problems for which you use MATLAB to help you solve, you must submit your code/.m-files as part of your work. Your code must run in order to receive full credit. If you include any plots, make sure that each has a title, axis labels, and readable font size, and include the final version of your plots as well as the code used to generate them.