Assignment 2

MT 3X03 & CS/SE 4X03: Scientific Computation

Due at 11:59 PM on Friday, November 15

Fall 2024

Submission Guidelines

Submit the following files on Avenue:

1. A PDF called `<FIRSTNAME>_<LASTNAME>_a2.pdf` (e.g., `matthew_giamou_a2.pdf`) containing all of your plots and written answers to mathematical and discussion questions (no need to include Julia code here). Show all steps in your solutions.

2. A file called `<FIRSTNAME>_<LASTNAME>_a2.jl` containing all of your Julia code solutions (we will run this with autograding scripts, so be sure to test it carefully). Use the `a2_template.jl` file we have provided as boilerplate: it contains function signatures for you to implement. Include all helper functions you implement as part of your solution in this file. **Do not use any additional `import` or `using` statements beyond what is provided in `a2_template.jl` - these will be detected and you will receive a grade of zero.**

The PDF can be any combination of typed/scanned/handwritten, so long as it is legible (you will receive a grade of zero on any section that cannot be easily understood). **Do not** submit the plotting scripts or test script provided to you. Read the syllabus to find the MSAF policy for this assignment.

Use of Generative AI Policy

If you use it, treat generative AI as you would a search engine: you may use it to answer general queries about scientific computing, but any specific component of a solution or lines of code must be cited (see the syllabus for citation guidelines).

This is an individual assignment. All submitted work must be your own, or appropriately cited from scholarly references. Submitting all or part of someone else's solution is an academic offence.

Problems

There are 10 problems worth a total of 100 marks. Please read all of the files included in the Assignment 2 handout as they contain useful information.

Symmetric and Iterative Linear System Solvers

This section continues our exploration of linear system solvers. You may find some of your functions from Assignment 1 useful here (feel free to reuse them as helper functions in your submission). You will also need to make use of the `norm` and `opnorm` functions in the `LinearAlgebra` package for the Euclidean norm (i.e., $p = 2$). As in Assignment 1, using additional packages or built-in functions and operators like \ that directly solve a problem for you is **prohibited**, and you will receive a mark of zero for attempting to do this.

Problem 1 (15 points)**:** Implement the $LDL^\top$ decomposition **without** partial pivoting for a symmetric matrix $A \in \mathbb{S}^n$ in the function signature `ldl_decomposition()`. This will be tested for accuracy and speed by a script similar to `test_a2.jl`.

Problem 2 (5 points)**:** Implement `ldl_solve()` function signature which uses the output of your $LDL^\top$ decomposition implementation to solve $Ax = b$. This will be tested by a script similar to `test_a2.jl`.

Problem 3 (15 points)**:** Implement the Jacobi method for a symmetric matrix $A \in \mathbb{S}^n$ in the function signature `jacobi_method()`. Additionally, implement the Gauss-Seidel method for a symmetric matrix $A \in \mathbb{S}^n$ in the function signature `gauss_seidel()`. Run the `plot_symmetric_solvers.jl` script and include the resulting figure in your PDF submission. Answer the following questions about this plot:

a) (5 points) Compare the convergence of the Jacobi and Gauss-Seidel methods. Do they both obey the upper bounds on error produced by your implementation?

b) (10 points) the Jacobi and Gauss-Seidel methods should both converge to a similar value. Explain why this occurs, and analytically derive an approximate expression for this value.

Computing Extremal Eigenvalues

Problem 4 (5 points)**:** Implement the power method from class for real symmetric matrices in the function template `power_method_symmetric`. Use the Bauer-Fike theorem for symmetric matrices with the input parameter `tol` as your termination criterion. This function will be tested by `test_a2.jl` with slightly different inputs. Do not change the function signature.

Problem 5 (10 points)**:** Derive an analytical expression for the eigenvalues and eigenvectors of the matrix $V \triangleq vv^\top$ for $v \in \mathbb{R}^n$ such that $\|v\|_2 = 1$.

Problem 6 (5 points)**:** Use your implementation of the power method and the result from Problem 5 to develop a method for determining the eigenpairs of $A \in \mathbb{S}^n$ for the k eigenvalues with the greatest absolute value. Implement your solution in the `extremal_eigenpairs()` function signature. This function will be tested by `test_a2.jl` with slightly different inputs. **Hint**: consider modifying A after finding each eigenpair in order of descending eigenvalue magnitude.

Newton's Method

This section applies Newton's method to estimation problems involving distance measurements. This is a common state estimation task, and the formulation we use here is a simplified version of what Global Navigation Satellite Systems (GNSS) like the Global Positioning System (GPS) use to provide accurate position measurements.

Problem 7 (10 points)**:** Consider the problem of localizing a receiver using idealized (noiseless) range measurements to transmitters with known positions $p_i \in \mathbb{R}^n$. Unless we have at least n transmitters in a non-degenerate configuration, there will be infinitely many solutions. This problem is equivalent to solving the following system of linear equations:

$$
\begin{aligned}
f_1(x) &= \|x - p_1\| - d_1 = 0 \\
f_2(x) &= \|x - p_2\| - d_2 = 0 \\
&\vdots \\
f_n(x) &= \|x - p_n\| - d_n = 0,
\end{aligned}
\tag{1}
$$

where $x \in \mathbb{R}^n$ is the unknown position of the receiver, and d_i is the noiseless measurement of the distance between x and p_i. If $F(x) = 0$ (where $F : \mathbb{R}^n \to \mathbb{R}^n$) is the system of equations in Eq. 1, find the Jacobian $J(x) = D_x F$ of $F(x)$. Include your derivation of the Jacobian as part of your PDF submission.

Problem 8 (10 points)**:** Implement Newton's method for the range-only localization (or *triangulation*) problem described in Problem 5 in the `newton()` function template in `template_a2.jl`. This function will be tested by `test_a2.jl` with slightly different inputs. Do not change its function signature.

Problem 9 (15 points)**:** Next, consider the realistic scenario where the distance measurements d_i are corrupted by noise and we have $m \geq n$ transmitting beacons. For independent zero-mean Gaussian distributions of the noise in d_i, we can estimate the position x by solving the nonlinear least-squares optimization problem

$$\min_{x \in \mathbb{R}^n} \sum_{i=1}^{m} |f_i(x)|^2 \tag{2}$$

with Newton's method, where the residuals f_i are the same as in Eq. 1. We will denote the objective function of Eq. 2 with

$$f(x) \triangleq \sum_{i=1}^{m} |f_i(x)|^2. \tag{3}$$

Find the gradient ∇f and Hessian $\nabla^2 f$ of f with respect to x. Include your derivations as part of your PDF submission. **Hint**: you may find tools like `https://www.matrixcalculus.org/` useful for checking your derivations.

Problem 10 (10 points)**:** Implement Newton's method for the unconstrained optimization problem in Eq. 2 in the `newton_optimizer()` function template in `template_a2.jl`. Run `plot_newton.jl` and include the plot as part of your PDF submission. Provide the following information in your PDF submission:

a) (5 points) numerical evidence that your implementation of Newton's method has converged to a critical point; and

b) (5 points) numerical evidence that this critical point is a local minimum.