Graded Homework 4 Applied Linear Algebra 2 2024

The solutions are due by 11:59PM on Friday April 19, 2024

Programming Exercise

Problem 4.1

4.1.a Coding Task

Implement a general descent method that can:

- choose the direction vector as a general descent direction or more specifically, as the residual at the current iterate x_k, or as a direction that is intentionally taken to be different from the residual;
- choose the optimal local stepsize, α_k^*, for the given direction vector, or select $\alpha_k = \tilde{\alpha}$ as a constant satisfying a convergence sufficient condition, or select α_k based on other specified criteria given below.

Your code should be organized so that it can be used as the basis for an iterative method solver for a symmetric positive definite linear system. For this assignment, it should be capable of running

1. steepest descent (SD);
2. Richardson's stationary method (RS) satisfying $\alpha < 2/\lambda_{max}$; This is just SD with a constant step, i.e., direcdtion vector r_k and fixed $\alpha_k = \alpha$ for all k.
3. a descent method (SDslow) with the direction vector taken as the residual but with the stepsize $\alpha_k = \sigma\alpha_k^*$ where σ is constant and such that convergence is still guaranteed.
4. Conjugate Gradient without preconditioning;

4.1.b Empirical Tasks

- In general, you are to highlight the strengths and weaknesses of the methods and their relative performances.
- A more specific goal is to empirically verify convergence rate bounds, orthogonality of successive residuals for SD, all residuals for CG, A-orthogonality for direction vectors of CG, and the effect of the condition number and distribution of the eigenvalues on the performance of the methods.

- Most of your experiments can be run in the eigencoordinate system but you should provide some evidence that the convergence behavior in the original coordinate system are the same when they should be.

- You should also consider local behavior of the residual and error, i.e., what happens on each step and relate it to the the size of the components of the error or residual in each eigendirection. Pay particular attention to the difference between SD and RS single step differences, e.g., you can save iterates from, say, RS and then examine what happens when you take one step of, say, SD staring from the saved RS iterates.

- The RS method requires that the constant stepsize α satisfy $\alpha < 2/\lambda_{max}$ so that the iteration will converge for all x_0. However, violating this condition **does not mean that the iteration diverges for all x_0.** Create an example where, in exact arithmetic, the iteration will converge even though $\alpha > 2/\lambda_{max}$ and explain the reasoning behind your construction. Does numerical noise due to arithmetic, and perhaps simulated here by random perturbations to the iterates or associated vectors, effect the conclusion?

4.1.c Comments on Test Problems

As noted, the behavior of the algorithms can be demonstrated with respect to convergence, rates etc. in the eigencoordinate system (see the study questions and class notes). So you can work with the system $\Lambda\tilde{x} = \tilde{b}$ and manipulate the choice of the positive diagonal matrix Λ for symmetric positive definite matrices. You should work with large numbers of randomly generated problems from various classes depending on the assumptions you are exploring. Of course, given a diagonal matrix Λ it is easy to generate a symmetric postive definite A or positive semidefinite A by generating an orthogonal matrix $Q \in \mathbb{R}^{n\times n}$ as discussed in first graded homework assignment. For such a matrix you can verify the invariance of the behavior for the algorithm in the eigencoordinates and in the coordinates associated with A. You can also generate symmetric positive definite matrics for which you do not initially specify the eigendecomposition by generating a random lower triangular matrix $L \in \mathbb{R}^{n\times n}$ with positive diagonal elements then forming $A = LL^T$. This allows you to easily determine the true solution for any righthand side vector b (as you can also with the eigendecomposition). You can use any library, e.g., Matlab, to compute the eigendecompostion for $A = LL^T$ for your analysis.

You should include a few tests on what happens when A or Λ is symmetric positive semidefinite, i.e., when the eigenvalues are nonnegative and the matrix may be singular. Be careful with the choice of righthand side vectors for these cases.