Program 3 Applied Linear Algebra 2 Spring 2024

The solutions are due by 11:59PM on Friday March 29, 2024

Programming Exercise

General Task

This assignment involves implementing and validating the Schur algorithm for displacement rank 2 and 4.

The Codes

1. Implement the Schur algorithm described in Set 5 of the class notes for computing the Cholesky factor R of a symmetric positive definite matrix M that has a rank 2 displacement with respect to the downshift matrix Z.

2. Implement the Schur algorithm described in Set 5 of the class notes for computing the Cholesky factor R of a symmetric positive definite matrix M that has a rank 4 displacement with respect to the downshift matrix Z.

Validation Tasks

1. Validate the rank 2 displacement code on symmetric positive definite Toeplitz matrices starting with the normalized generator $G \in \mathbb{R}^{n\times 2}$ and displacement $\nabla T = G\Phi G^T$, with $\Phi = \begin{pmatrix} e_1 & -e_2 \end{pmatrix} \in \mathbb{R}^2$ described in Set 5.

2. Validate the rank 2 displacement code on symmetric positive definite matrices, M, starting with a generator $G \in \mathbb{R}^{n\times 2}$ and displacement $\nabla M = G\Phi G^T$, with $\Phi = \begin{pmatrix} e_1 & -e_2 \end{pmatrix} \in \mathbb{R}^2$.

3. Validate the rank 4 displacement code on Toeplitz least squares problems $\|b - Tx\|_2^2$ by computing the Cholesky factor of the normal equations matrix $T^T T$ starting with the normalized generator $G \in \mathbb{R}^{n\times 2}$ and displacement $\nabla T = G\Phi G^T$, with $\Phi = \begin{pmatrix} e_1 & -e_2 \end{pmatrix} \in \mathbb{R}^2$ described in Set 5.

4. It is suggested that you first implement an interactive code that allows you to set specific problems, check their definiteness using a library eigenvalue routine, monitor details of the computations, e.g., printing the generator before and after normalization and other debugging prints, and then assess the accuracy of the factorization and solution of the problem defined. After targeted experiments on both codes and all three types of problems, then a more systematic validation code can be done where you take steps to guarantee the definiteness of all matrices that require it.

Generation of Test Problems

A key consideration in this assignment is the generation of test problems.

1. When generating a Toeplitz symmetric matrix, T, for the displacement rank 2 algorithm recall that the normalized generator that has the first column of the Cholesky factor as its first column/row does not guarantee that T is positive definite. Therefore, you should check the eigenvalues of the matrix generated. If the matrix is not definite it is still of interest to apply the Schur algorithm. You should be able to detect the indefiniteness (or nearness to singularity) by the failure of the hyperbolic rotation to exist. Hence, your code should detect and report this in an orderly fashion. Definiteness can be guaranteed by boosting the diagonal value of τ_0^2 so that the matrix is sufficiently large to yield definiteness. Note strict diagonal dominance guarantees this but you may have definiteness before that value.

$$T = \sum_{j=0}^{n-1} Z^j (G\Phi G^T) \left(Z^j\right)^T .$$

   It is suggested you check both using.

2. A similar comment applies to a symmetric rank 2 displacement matrix M. Specifying G and using

$$M = \sum_{j=0}^{n-1} Z^j (G\Phi G^T) \left(Z^j\right)^T .$$

   gives a symmetric matrix of the appropriate displacement rank but it does not guarantee definiteness. This matrix should be formed to verify the Cholesky factor when it is definite and to check the eigenvalues as before. An arbitrary symmetric positive definite matrix does not necessarily have a rank 2 displacement so it will be necessary to boost the diagonal of M. Note however that this does not have the simple correspondence of τ_0^2 to the entire diagonal that is available for the Toeplitz case. So modifying elements of G to boost the diagonal elements can be done by a simple repeated generation of M and checking the eigenvalues. Note also that the diagonal elements of M have summation form, as do the diagonal elements of T (which are known to all be equal to τ_0^2),

$$\mu_{ii} = e_i^T M e_i = \sum_{j=0}^{n-1} e_i^T Z^j (G\Phi G^T) \left(Z^j\right)^T e_i.$$

3. In the case of $T \in \mathbb{R}^{m\times n}$ with $n \leq m$, and the least squares problem, you must specify the first column $c = Te_1 \in \mathbb{R}^m$ and first row $r = T^T e_1 \in \mathbb{R}^n$ where the two respective first elements must be equal. Generically, unless you have several repeated values for across multiple diagonals the matrix will be full column rank. Nevertheless, your code

should check the eigenvalues of $T^T T$ or the singular values of T. (Note the notation used in the description of T, $e_i^T T e_j = \tau_{i-j}$ making

$$c = \begin{pmatrix} \tau_0 \\ \tau_1 \\ \vdots \\ \tau_{m-1} \end{pmatrix} \quad \text{and} \quad r = \begin{pmatrix} \tau_0 \\ \tau_{-1} \\ \tau_{-2} \\ \vdots \\ \tau_{1-n} \end{pmatrix}.$$

4. You should also test the square case, i.e., $m = n$. In this case, T is nonsymmetric nonsingular Toeplitz and to solve $Tx = b$ you solve the normal equations $T^T T x = T^T b$. While this is **not** the preferred way to solve such a system by a Schur-like algorithm it is of interest to check the effectiveness. (There are nonsymmetric displacements that admit pivoting in an associated algorithm.)

5. As with the rank 2 displacement code, you should check for problems defining the hyperbolic rotation and the euclidean reflector or rotation.

6. The code as described in the notes is easily generalized to a displacement rank of $2k$ where each set of k rows is handled as in the algorithm described but the entire first column of each must have all but their first element eliminated before finishing the normalization with a single hyperbolic rotation. You are encouraged to consider implementing such a version.