COL726 Assignment 3

15 February – 5 March, 2022

Note: All answers should be accompanied by a rigorous justification, unless the question explicitly states that a justification is not necessary.

1. As you may know, a matrix is said to be in reduced row echelon form (RREF) if the first nonzero entry in every row (i) is strictly to the right of the first nonzero entry in previous rows, (ii) is a 1, and (iii) has all 0's above it, for example:

$$\begin{bmatrix} 1 & 0 & * & 0 & * \\ & 1 & * & 0 & * \\ & & & 1 & * \end{bmatrix}$$

   (a) Design an algorithm to compute a factorization of a given wide matrix, $\mathbf{A} \in \mathbb{C}^{m \times n}$ with $m \leq n$, such that the rightmost factor is in RREF. To account for floating-point error, consider intermediate values to be zero if they are smaller in absolute value than a given parameter ϵ (however, the output matrices should contain exact zeros).

   (b) Implement your algorithm as a Python function `rref_factorize(A, eps)` which returns all the factors described in your written answer.

2. Consider the system of equations

$$\underbrace{\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^* & \mathbf{0} \end{bmatrix}}_{\mathbf{M}} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{p} \\ \mathbf{q} \end{bmatrix},$$

   where $\mathbf{A} \in \mathbb{C}^{m \times m}$ is Hermitian positive definite and $\mathbf{B} \in \mathbb{C}^{m \times n}$ is full rank with $m \geq n$; correspondingly, $\mathbf{p} \in \mathbb{C}^m$ and $\mathbf{q} \in \mathbb{C}^n$.

   (a) Show that the matrix $\mathbf{M}$ is indefinite, i.e. there exist vectors $\mathbf{u}$ and $\mathbf{v}$ such that $\mathbf{u}^*\mathbf{M}\mathbf{u} > 0$ and $\mathbf{v}^*\mathbf{M}\mathbf{v} < 0$.

   (b) Give a method to solve this system efficiently as possible. In particular, your method should take fewer flops than naively applying LU or QR factorization to $\mathbf{M}$.

   Hint: Consider eliminating $\mathbf{x}$ and solving for $\mathbf{y}$ alone first.

3. Suppose $\mathbf{A} \in \mathbb{C}^{m \times m}$ is a Hermitian positive definite matrix.

   (a) Show that decreasing the magnitude of any off-diagonal entries preserves positive definiteness. That is, if $\mathbf{B}$ is a matrix whose one particular entry b_{ij} is arbitrary with $|b_{ij}| \leq |a_{ij}|$, its symmetrical entry is $b_{ji} = \overline{b_{ij}}$, and all other entries are $b_{ij} = a_{ij}$, then $\mathbf{B}$ is also positive definite.

(b) Given the original Cholesky factorization $\mathbf{A} = \mathbf{LL}^*$, describe an algorithm to compute the Cholesky factorization of $\mathbf{B}$ in $O(m^2)$ flops.

4. Recall that the conjugate gradient method ensures a monotonic decrease in the $\mathbf{A}$-norm of the error $\|\mathbf{e}_n\|_{\mathbf{A}}$, but not necessarily in other quantities related to convergence, such as $\|\mathbf{e}_n\|_2$ and $\|\mathbf{r}_n\|_2$. Give upper bounds for both of these, as a function of n and $\mathbf{A}$.

5. Let $\mathbf{A} \in \mathbb{R}^{m \times m}$ be a symmetric, nonsingular, indefinite matrix. Suppose I nevertheless apply conjugate gradients to solve $\mathbf{Ax} = \mathbf{b}$ for some vector $\mathbf{b} \in \mathbb{R}^m$. Explain which of the following properties fail to hold (even in exact arithmetic) and why.

   (a) $\mathcal{K}_n = \langle \mathbf{x}_1, \dots, \mathbf{x}_n \rangle = \langle \mathbf{p}_0, \dots, \mathbf{p}_{n-1} \rangle = \langle \mathbf{r}_0, \dots, \mathbf{r}_{n-1} \rangle = \langle \mathbf{b}, \dots, \mathbf{A}^{n-1}\mathbf{b} \rangle$.

   (b) $\mathbf{r}_n^T \mathbf{r}_j = 0$ and $\mathbf{p}_n^T \mathbf{A} \mathbf{p}_j = 0$ for all $j < n$.

   (c) $\mathbf{x}_n$ minimizes $\phi(\mathbf{x}) = \mathbf{x}^T \mathbf{A} \mathbf{x}$ over all $\mathbf{x} \in \mathcal{K}_n$.

   (d) No divisions by zero can occur before the solution is found.

6. Since GMRES treats the matrix as a black box, it may converge slowly even if the matrix is easy to invert otherwise. This can be addressed using the notion of preconditioning.[1]

   (a) Suppose GMRES is applied to solving $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{A} \in \mathbb{R}^{m \times m}$ is a real diagonal matrix with bounded entries $a_{ii} \in [-u, -l] \cup [l, u]$. Derive an upper bound for $\|\mathbf{r}_n\|_2 / \|\mathbf{b}\|_2$ as a function of n, l, and u, analogous to T&B Theorem 38.5.

   (b) Implement a function `gmres(A_func, b, n)` which takes a linear function $A : \mathbb{R}^m \to \mathbb{R}^m$ and a vector $\mathbf{b} \in \mathbb{R}^m$, runs n iterations of GMRES on the problem $A(\mathbf{x}) = \mathbf{b}$, and returns the matrix of iterates $[\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{m \times n}$.

   The provided code defines functions `A_dict = test_matrix(m)` which creates a random sparse matrix $\mathbf{A} \in \mathbb{R}^{m \times m}$ represented as a Python dictionary, and `y = matvec_dict(A_dict, x)` which computes its matrix-vector products $\mathbf{y} = \mathbf{Ax}$. You are encouraged to look at the code for more details. Your `gmres` solver should be usable with these functions as follows:

```
def solve(A_dict, b, n):
    def A_func(x):
        return matvec_dict(A_dict, x)
    return gmres(A_func, b, n)
```

   (c) The matrix produced by `test_matrix` has diagonal entries typically much larger

[1] See T&B 40 if you want to learn more.

than off-diagonal entries. Take advantage of this fact by implementing a function `solve_prec_l(A_dict, b, n)` which solves $\mathbf{Ax} = \mathbf{b}$ by applying GMRES to the equivalent problem $\mathbf{D}^{-1}\mathbf{Ax} = \mathbf{D}^{-1}\mathbf{b}$, where $\mathbf{D}$ contains only the diagonal entries of $\mathbf{A}$. Similarly, implement `solve_prec_r(A_dict, b, n)` which applies GMRES to $\mathbf{AD}^{-1}\mathbf{y} = \mathbf{b}$, followed by $\mathbf{x} = \mathbf{D}^{-1}\mathbf{y}$. Both functions should return the matrix of iterates $[\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{m\times n}$.

Note: You must not actually form the matrices $\mathbf{D}^{-1}\mathbf{A}$ and $\mathbf{AD}^{-1}$; just implement functions analogous to `A_func` that compute their action on a vector $\mathbf{x}$.

Test all three solver functions on a random $\mathbf{A}$ from `test_matrix` and a random vector $\mathbf{b} \in \mathbb{R}^m$, with $m = 1000$ and $n = 20$. Plot the relative residual norm $\|\mathbf{r}_i\|_2/\|\mathbf{b}\|_2$ versus iteration i with a logarithmic y-axis (include the data point $\|\mathbf{r}_0\|_2/\|\mathbf{b}\|_2 = 1$). Since $\mathbf{A}$ is random, you may see somewhat different behaviour on each run, so try it a few times and pick one (or more) representative plot(s) to include in your PDF.

Also explain in your written answer why the residual may not be monotonically decreasing for `solve_prec_l`, but it is for `solve_prec_r`.

Collaboration policy: Refer to the policy on the course webpage.

If you collaborated with others to solve any question(s) of this assignment, give their names in your submission. If you found part of a solution using some online resource, give its URL.

Submission: This assignment has two submission forms on Gradescope. In one form, you have to submit a PDF of your answers for all questions, and in the other you have to submit your code for Questions 1 and 6. Both submissions must be uploaded before the assignment deadline.

Code submissions should contain two files `a3q1.py` and `a3q6.py` which contain the requested functions and any helper functions.