Computer Science Department, Ben Gurion University of the Negev

Optimization methods with Applications - Spring 2023

Home assignment 2

Due on 18/05/2023.

May 2, 2023

1. **The efficiency of different iterative methods for solving a linear system.**

   (a) Implement programs for the four methods: Jacobi, Gauss-Seidel, Steepest (or Gradient) Descent (SD) and Conjugate Gradient (CG). See the CG Algorithm in the NLA-6 notes.

   (b) Define the random symmetric positive definite matrix A below (which we define as sparse, see section 4.3.6). In Julia (after adding the `SparseArrays` package):

```
using SparseArrays;
n = 256;
A = sprandn(n,n,5/n);
A = A'*spdiagm(0=>rand(n))*A + 0.1*spdiagm(0=>ones(n));
```

   In python:

```
import numpy as np
from scipy.sparse import random
import scipy.sparse as sparse
n = 256
A = random(n, n, 5 / n, dtype=float)
v = np.random.rand(n)
v = sparse.spdiags(v, 0, v.shape[0], v.shape[0], 'csr')
A = A.transpose() * v * A + 0.1*sparse.eye(n)
```

   For $\mathbf{x}^{(0)} = 0$, and a random $\mathbf{b}$, (`b = rand(n)` in Julia or `b = rand(n,1)` in Matlab) solve the system $A\mathbf{x} = \mathbf{b}$ with each of the methods, with at most 100 iterations. For Jacobi, use the weights $\omega = 1.0$ (standard Jacobi) and if it doesn't converge, search a reasonable $0 < \omega < 1$ that does lead for convergence most of

the times (remember, it is a random experiment).
Use the `semilogy()` plotting function to plot a convergence graph of $\|A\mathbf{x}^{(k)} - \mathbf{b}\|_2$. Also plot the convergence factor $\frac{\|A\mathbf{x}^{(k)}-\mathbf{b}\|_2}{\|A\mathbf{x}^{(k-1)}-\mathbf{b}\|_2}$, using `plot()`.

2. **Convergence properties**

(a) Show that for any symmetric positive definite matrix $A \in \mathbb{R}^{n\times n}$, the Richardson method

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \frac{1}{\|A\|}\left(\mathbf{b} - A\mathbf{x}^{(k)}\right),$$

converges to the solution of $A\mathbf{x} = \mathbf{b}$, where $\|\cdot\|$ is any induced matrix norm.

(b) Show that if A in the previous section is indefinite (has both positive and negative eigenvalues), then the Richardson method diverges.

(c) We will now prove the convergence of Steepest Descent with optimal choice of $\alpha^{(k)} = \alpha_{opt}$ as shown in the NLA-6 notes.

i. Show that (recall: $A \succ 0$)

$$f(\mathbf{x}^{(k+1)}) = f(\mathbf{x}^{(k)}) - \frac{1}{2}\frac{\langle \mathbf{r}^{(k)}, A\mathbf{e}^{(k)}\rangle^2}{\langle \mathbf{r}^{(k)}, A\mathbf{r}^{(k)}\rangle} < f(\mathbf{x}^{(k)}),$$

where $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}^* - \mathbf{x}\|_A^2$, defined in the NLA-6 notes. The strict inequality holds as long as that $\mathbf{r}^{(k)} \neq 0$.

ii. Using the previous section, find a scalar factor $C^{(k)}$ such that

$$f(\mathbf{x}^{(k+1)}) = C^{(k)} \cdot f(\mathbf{x}^{(k)})$$

where $C^{(k)} < 1$ (this upper bound is achieved from the inequality above).

iii. Now, using the fact that for every symmetric matrix $A \in \mathbb{R}^{n\times n}$:

$$\forall \mathbf{v} \in \mathbb{R}^n : \lambda_{min} \leq \frac{\mathbf{v}^\top A\mathbf{v}}{\mathbf{v}^\top \mathbf{v}} \leq \lambda_{max},$$

(this is called a Rayleigh quotient) show that $C^{(k)} \leq 1 - \frac{\lambda_{min}}{\lambda_{max}} < 1$.

iv. Conclude that SD converges: $\lim_{k\to\infty} f(\mathbf{x}^k) = 0$, and hence $\lim_{k\to\infty} \mathbf{x}^k = \mathbf{x}^*$

3. In this question we will develop a method called GMRES(1). Assume that $A \in \mathbb{R}^{n\times n}$ is full rank, positive definite, but non-symmetric. We want to solve a linear system $A\mathbf{x} = \mathbf{b}$ using a method that is similar to Steepest Descent (SD), with an optimal step-size per iteration. We define

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha^{(k)}(\mathbf{b} - A\mathbf{x}^{(k)}).$$

and we choose $\alpha^{(k)}$ to minimize the residual norm $\|\mathbf{r}^{(k+1)}\|_2 = \|\mathbf{b} - A\mathbf{x}^{(k+1)}\|_2$ with respect to $\alpha^{(k)}$ (in SD we minimized the $A-$norm of the error $\mathbf{e}^{(k+1)}$ instead of the residual norm, but that norm is relevant for SPD matrices only).

(a) Show that

$$\alpha^{(k)} = \frac{(\mathbf{r}^{(k)})^\top A\mathbf{r}^{(k)}}{(\mathbf{r}^{(k)})^\top A^\top A\mathbf{r}^{(k)}}.$$

(b) (non-mandatory) Show that as in SD, at each iteration we have to compute only one matrix vector multiplication $A\mathbf{r}^{(k)}$.

(c) Demonstrate the convergence of the GMRES method for the following matrix:

$$A = \begin{bmatrix} 5 & 4 & 4 & -1 & 0 \\ 3 & 12 & 4 & -5 & -5 \\ -4 & 2 & 6 & 0 & 3 \\ 4 & 5 & -7 & 10 & 2 \\ 1 & 2 & 5 & 3 & 10 \end{bmatrix}.$$

Choose $\mathbf{b} = [1,1,1,1,1]^\top$, and $\mathbf{x}^{(0)} = [0,0,0,0,0]^\top$, and apply 50 iterations. Plot a graph of the residual norm vs. the iterations using the `semilogy()` plotting function.

(d) The graph that you get in the previous subsection is monotone. Explain why?

(e) Explicitly define the method $GMRES(2)$:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_1^{(k)}\mathbf{r}^{(k)} + \alpha_2^{(k)}\mathbf{r}^{(k-1)}.$$

where the vector $[\alpha_1^{(k)}, \alpha_2^{(k)}]^\top$ is chosen to minimize $\|\mathbf{b} - A\mathbf{x}^{(k+1)}\|_2$.
Guidance: Define a vector $\vec{\alpha}^{(k)} = [\alpha_1^{(k)}, \alpha_2^{(k)}]^\top$, and a $n \times 2$ matrix $R^{(k)} = [\mathbf{r}^{(k)}, \mathbf{r}^{(k-1)}]$, and write in matrix form $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + R^{(k)}\vec{\alpha}^{(k)}$. Find a closed form for $\vec{\alpha}$.

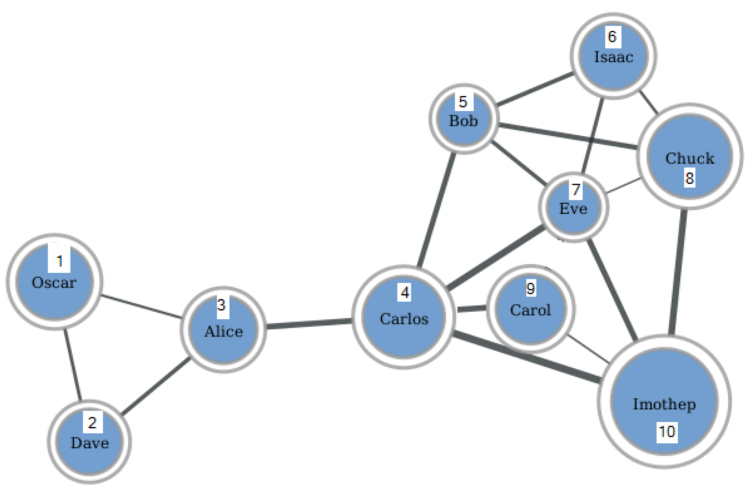
Remark for general knowledge: the method GMRES(1) that approximates only one parameter α stagnates for indefinite systems, choosing $\alpha^{(k)} = 0$ from some iteration on. The general GMRES(m) method approximates m such α's (for m previous search *directions) similarly to Conjugate Gradient and is able to solve indefinite systems as long as there aren't more than m negative eigenvalues. The GMRES(m) method also incudes orthogonalizations in the minimization process for the α's to avoid numerical errors.*

4. Graph-Laplacians

   Graph Laplacians are a powerful tool is computer science theory and have many applications. You may have a look at the following links if you're interested - there is a large amount of research devoted at these:

   - `https://en.wikipedia.org/wiki/Laplacian_matrix`
   - `https://en.wikipedia.org/wiki/Algebraic_connectivity`
   - `https://sites.google.com/a/yale.edu/laplacian/`
   - `http://www.cs.yale.edu/homes/spielman/eigs/`
   - `https://www.cs.yale.edu/homes/spielman/PAPERS/icm10post.pdf`
   - `https://theory.epfl.ch/vishnoi/Lxb-Web.pdf`

   In many cases listed above, we are needed to solve linear systems or compute eigenvalues/eigenvectors with these graph Laplacians. Below we see a graph of a small social network:



It's corresponding graph Laplacian matrix is:

$$L = \begin{bmatrix} 2 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 3 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 5 & -1 & 0 & -1 & 0 & -1 & -1 \\ 0 & 0 & 0 & -1 & 4 & -1 & -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 3 & -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & -1 & 5 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & -1 & -1 & 4 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 2 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & -1 & -1 & 4 \end{bmatrix}.$$

Note that this matrix is symmetric and positive semi-definite. It has a null-space of a single vector: the constant $[1, 1, 1, 1..., 1]$. Let's define a vector $\mathbf{b} = [1, -1, 1, -1, ..., 1]$. That is a vector with a sum of zero, so there is a solution $L\mathbf{x} = \mathbf{b}$.
Remark: note this null space in your analyses. Since $L\mathbf{1} = 0$, then for any M, $\rho(I - M^{-1}L) = 1$. You have to exclude this vector to get a meaningful analysis when thinking of the section below.

(a) Solve the system using the Jacobi method, up to five digits of accuracy. Show the convergence graph of the residual norm. How many iterations are needed? What is the convergence factor of the method?

(b) We will now accelerate Jacobi with a very powerful block preconditioner, instead of the simple diagonal one in Jacobi (the matrix D). We will partition the unknowns of L into two subgroups: $\{1, 2, 3\}$, and $\{4, 5, 6, 7, 8, 9, 10\}$. According to this partitioning we will define the submatrices $M_1 = L_{1..3,1..3} \in R^{3\times 3}$ and $M_2 = L_{4..10,4..10} \in R^{7\times 7}$. Now we will use the block preconditioner M which can be inverted by inverting the two smaller submatrices (which is cheaper):

$$M = \begin{bmatrix} M_1 & 0 \\ 0 & M_2 \end{bmatrix} \quad M^{-1} = \begin{bmatrix} M_1^{-1} & 0 \\ 0 & M_2^{-1} \end{bmatrix}$$

Repeat section 4a with the iteration $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \omega M^{-1}(\mathbf{b} - L\mathbf{x})$. Does the method converge in fewer iterations than standard Jacobi? What is the convergence factor now? Note that the matrices M_1 and M_2 are not singular like L, so this is well-defined.
Remark: In sections (b) and (c) it is better to use a damping parameter, i.e.: $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \omega M^{-1}(\mathbf{b} - L\mathbf{x})$. *Choose something like* $\omega \approx 0.7$, *which works nicely. You can also search for the optimal one.*

(c) A student argued that M_2 is 7×7 and is still too large compared to the 10×10 in L, and that he wants to add another group (these numbers are not really large - it's just for illustration purposes). Using what we've learned in class, divide the unknowns $\{1, ..., 10\}$ into groups of balanced sizes 4,3,3 so that the iterations using a block diagonal M with the three blocks M_1, M_2, M_3 will be most efficient (try a few options and choose what's best). Repeat section 4a with the new partitioning. Do you see a connection to the structure of the graph? Explain.
Remark: the indices of the groups should not necessarily be sequential, i.e., something like $\{1, 3\}$ *and* $\{2, 4\}$ *is also OK.*