Homework 8. Due Wednesday, Oct. 23.

Reference: [NW] J. Nocedal and S. Wright, "Numerical Optimization", Second Edition, Springer, 2006 (available online e.g. via UMD library).

1. **(5 pts)** Prove that the conjugate gradient algorithm, the preliminary version (Algorithm 5.1, page 108 in [NW]), is equivalent to Algorithm 5.2 (CG) (page 112 in [NW]), i.e., that

$$\alpha_k = \frac{r_k^\top r_k}{p_k^\top A p_k} \quad \text{and} \quad \beta_{k+1} = \frac{r_{k+1}^\top r_{k+1}}{r_k^\top r_k}. \tag{1}$$

2. **(5 pts)** Let A be an $n \times n$ matrix. A subspace spanned by the columns of an $n \times k$ matrix B is an invariant subspace of A if A maps it into itself, i.e., if $AB \subset \mathsf{span}(B)$. This means that there is a $k \times k$ matrix C such that $AB = BC$. Prove that if a vector $r \in \mathbb{R}^n$ lies in the k-dimensional subspace spanned by the columns of B, i.e., if $r = By$ for some $y \in \mathbb{R}^k$ (r is a linear combination of columns of B with coefficients y_1, ..., y_k) then the Krylov subspaces generated by r stop expanding at degree k?1, i.e,

$$\mathsf{span}\{r, Ar, ..., A^p r\} = \mathsf{span}\{r, Ar, ..., A^{k+1} r\} \quad \forall p \geq k. \tag{2}$$

3. **(5 pts)** Prove Theorem 5.5 from [NW], page 115. Here are the steps that you need to work out.

   (a) Construct a polynomial $Q(\lambda)$ of degree $k+1$ with roots λ_n, λ_{n-1}, ...,λ_{n-k+1}, and $\frac{1}{2}(\lambda_1 + \lambda_{n-k})$ such that $Q(0) = 1$.

   (b) Argue that $P(\lambda)$ defined as

$$P(\lambda) = \frac{Q(\lambda) - 1}{\lambda} \tag{3}$$

   is a polynomial, not a rational function, by referring to the theorem about factoring polynomials. Cite that theorem.

   (c) Use the ansatz

$$\|x_{k+1} - x^*\|_A^2 \leq \min_{P \in \mathcal{P}_k} \max_{1 \leq i \leq n} [1 + \lambda_i P(\lambda_i)]^2 \|x_0 - x^*\|_A^2. \tag{4}$$

   Argue that

$$\|x_{k+1} - x^*\|_A^2 \leq \max_{1 \leq i \leq n} Q^2(\lambda_i) \|x_0 - x^*\|_A^2. \tag{5}$$

(d) Show that

$$\|x_{k+1} - x^*\|_A^2 \leq \max_{\lambda \in [\lambda_1, \lambda_{n-k}]} \left| \frac{\lambda - \frac{1}{2}(\lambda_1 + \lambda_{n-k})}{\frac{1}{2}(\lambda_1 + \lambda_{n-k})} \right|^2 \|x_0 - x^*\|_A^2. \tag{6}$$

(e) Find the maximum of the function in the right-hand side of the last equation in the interval $[\lambda_1, \lambda_{n-k}]$.

(f) Finish the proof of the theorem.

4. **(5 pts)** *The coding for this problem can be done in Matlab or Python. If you choose Python, you can export the matrix* L_{symm} *and the vector* b_{symm} *to Python and do the rest of the work in Python.* The goal of this problem is to practice the *conjugate gradient algorithm (CG) with and without preconditioning* on a meaningful problem. An explanation for its setup is a bit long, but it is a typical case that a lot of effort is spent on problem setup. I have done the setup for you. Your job will be just to code the CG algorithms.

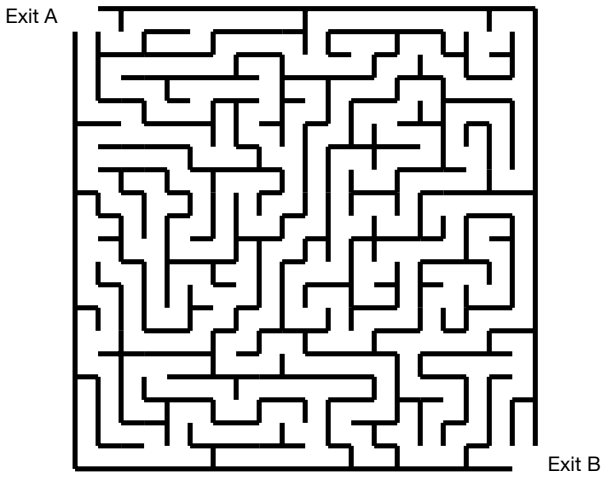

Figure 1: A maze with two exits.

Consider a maze with two exits, A, and B, shown in Fig. 1 taken from this paper by W. E and E. Vanden-Eijnden. This maze consists of a 20×20 array of cells, $N = 400$ cells in total. We will number the cells from 1 to 400 column-wise, i.e., the first column contains cells with indices from 1 to 20, the second one from 21 to 40, and so on. Exit A is at cell 1 while Exit B is at cell 400.

Let A be the adjacency matrix for this maze. A is 400×400, $A_{ij} = 1$ if cells i and j are adjacent and there is no wall between them, and $A_{ij} = 0$ otherwise. A random walker makes a step from a cell to any adjacent cell not separated by a wall with equal probability. The stochastic matrix (the other names for it are the transition matrix and the Markov matrix) for this random walk can be found as $P = R^{-1}A$ where R is a diagonal matrix with row sums of A along its diagonal. Row sums of P are all equal to 1, and P_{ij} is the probability that the random walker located at cell i will next move to cell j.

Our goal is to compute the *committor* $x \in \mathbb{R}^N$ for this random walk, i.e., the vector of probabilities whose component x_i is the probability that the walker located at cell i will first arrive at Exit B rather than Exit A. This vector of probabilities satisfies:

$$x_1 = 0, \quad x_N = 1, \quad x_i = \sum_{j=1}^{N} P_{ij} x_j, \quad 2 \le i \le N-1. \tag{7}$$

Equation (7) is obtained from the following reasoning. The probability of exiting via B rather than A from cell i is equal to the sum of products of probabilities to move to a cell j from i and exit from j via B rather than A. The sum is over all other cells j. Equation (7) can be written in the matrix form as follows. We denote $P - I$ by L. Then

$$L_{2:(N-1),2:(N-1)} x_{2:(N-1)} = b_{2:(N-1)}, \tag{8}$$

where b is obtained by $b = -Le_N$, where e_N is the vector with entries $1, ..., N-1$ equal to zero and entry N equal to 1. You can check this, or just believe me. Recall that $L = R^{-1}A - I$. It is not a symmetric matrix, but it can be symmetrized as follows. We will mark with tilde all submatrices $(2 : (N-1), 2 : (N-1))$ and all subvectors with indices in $2 : (N-1)$. Then we symmetrize $\tilde{L}$ using the fact that the adjacency matrix A is symmetric, $\tilde{L} = \tilde{R}^{-1}\tilde{A} - \tilde{I}$, and multiplication by $\tilde{R}^{\pm 1/2}$:

$$\tilde{L}\tilde{x} = \tilde{b}, \quad \text{the linear system we need to solve;} \tag{9}$$

$$(\tilde{R}^{-1}\tilde{A} - \tilde{I})\tilde{x} = \tilde{b}, \quad \text{recall what is } L; \tag{10}$$

$$\tilde{R}^{1/2}(\tilde{R}\tilde{A} - \tilde{I})\tilde{R}^{-1/2}\tilde{R}^{1/2}\tilde{x} = \tilde{R}^{1/2}\tilde{b}, \quad \text{multiply by } \tilde{R}^{1/2}, \text{ insert } \tilde{R}^{-1/2}\tilde{R}^{1/2}; \tag{11}$$

$$(\tilde{R}^{1/2}\tilde{A}\tilde{R}^{1/2} - \tilde{I})\tilde{R}^{1/2}\tilde{x} = \tilde{R}^{1/2}\tilde{b}, \quad \text{do some algebra.} \tag{12}$$

The matrix $\tilde{R}^{1/2}\tilde{A}\tilde{R}^{1/2} - \tilde{I} =: L_{\mathsf{symm}}$ is symmetric. $\tilde{R}^{1/2}\tilde{x} =: y$ is a new vector of unknowns. $\tilde{R}^{1/2}\tilde{b} =: b_{\mathsf{symm}}$ is the new right-hand side. It is possible to check (just believe me), that the matrix $-L_{\mathsf{symm}}$ is symmetric positive definite.

Thus, here is the linear system with a symmetric positive definite matrix:

$$\boxed{-L_{\mathsf{symm}} y = -b_{\mathsf{symm}}, \quad x_{2:(N-1)} = \tilde{R}^{-1/2} y.} \tag{13}$$

The Matlab code `random_walk_in_maze.m` visualizes the maze, sets up this linear system, solves it using the built-in solver "\", visualizes the solution, and plots the eigenvalues of $-L_{\mathsf{symm}}$. Task. Modify the code to solve Eq. (13) using the conjugate gradient algorithm with- out and with preconditioning (Algorithms 5.2 and 5.3) in [NW]. Use the incomplete Cholesky preconditioner. The corresponding Matlab command is

```
ichol_fac = ichol(sparse(A));
M = ichol_fac*ichol_fac';
```

Stop iterations when the residual will have a norm less than 10^{-12}. Plot the norm of the residuals after each iteration for the CG algorithm with and without preconditioning in the same figure. Use the logarithmic scale along the y-axis. Visualize the computed solution. Link your code to the pdf file with the homework.

The coding for this problem can be done in Matlab or Python. If you choose Python, you can export the matrix L_{symm} and the vector b_{symm} to Python and do the rest of the work in Python.