AMSC660. Final exam.

Maria Cameron

December 15, 2023

1 Problem 1

Think about a maze similar to the one in Problem 5 of HW7. *It is helpful to read the entire statement of Problem 5 in HW 7 for solving this problem.*

The maze consists of an $n \times n$ array of cells. The connectivity of the maze is described by an adjacency matrix A of size $N \times N$ where $N = n^2$: $A_{ij} = 1$ if cells i and j are adjacent and there is no wall between them, and $A_{ij} = 0$ otherwise. We assume that the maze is connected, i.e., there is a path between any pair of cells. In particular, there is a path from the top left cell indexed by 1 and the bottom right cell indexed by N. Note that the matrix A is symmetric.

As in problem 5 of HW7, we compute row sums of A and convert the adjacency matrix A into a stochastic matrix P:

$$P = R^{-1}A, \quad \text{where} \quad R = \mathsf{diag}\left\{\sum_{j=1}^{N} A_{1j}, \ldots, \sum_{j=1}^{N} A_{Nj}\right\}. \tag{1}$$

1. **(4 pts)** Prove that all eigenvalues of P are real.

   Hint: Start with the identity $PV = V\Lambda$ where V is the matrix whose columns are the eigenvectors of P and Λ is the diagonal matrix with the corresponding eigenvalues of P along its diagonal.

2. **(2 pts)**The eigenvectors of P are, generally, not orthogonal with respect to the dot product, i.e. $V^\top V \neq I$, where I is the $N \times N$ identity matrix. Find a symmetric positive definite matrix (?) such that $V^\top(?)V = I_{N\times N}$ where V is the matrix whose columns are the eigenvectors of P, i.e., the eigenvectors of P are orthogonal with respect to the (?)-inner product.

3. **(4 pts)** Evidently, the matrix $L := P - I$ is not invertible because its row sums are all equal to zero. However, in problem 5, HW7, we solved a linear system whose

matrix $\tilde{L}$ was obtained from L by removing the first and the last row and the first and the last column of L:

$$\tilde{L} := L_{2:(N-1),(2:N-1)}. \tag{2}$$

Prove that the matrix $\tilde{L}$ is invertible.

Hint: Proceed from converse. Assume that there is $x \in \mathbb{R}^{N-2}$, $x \neq 0$, such that $\tilde{L}x = 0$, i.e., x is the eigenvector corresponding to the zero eigenvalue of $\tilde{L}$. Normalize x so that its maximal entry in absolute value equals 1. Now observe that $\tilde{L}x = 0$ is equivalent to $x = \tilde{P}x$. Recall that P is irreducible, i.e., each pair of sites is connected by a path in the maze (the maze can be thought of as a graph). This means that there is a sequence connecting any pair of sites i and j,

$$(i, k_1),\ (k_1, k_2),\ \ \ldots,\ (k_s, j)$$

such that the entries of P with subscripts in this sequence are positive.

2 Problem 2

Depicting graphs nicely is important as some properties of a graph can be first detected by visual inspection.

(10 pts) We consider an approach based on minimizing an energy function for a graph. An unweighted undirected graph $G(V, E)$ consists of a set of vertices $V = \{1, \ldots, N\}$ a set of edges $E \subset \{(i, j) \mid i, j \in V\}$. A graph $G(V, E)$ can be defined by its adjacency matrix A:

$$A_{i,j} = \begin{cases} 1, & (i,j) \in E \\ 0, & \text{otherwise} \end{cases}, \quad 1 \leq i, j \leq N. \tag{3}$$

Since the graph is undirected, A is symmetric.

In a depiction of a graph, *we want all linked vertices to be at a distance of 1 from each other, while all unlinked vertices at a distance of at least as large as $\sqrt{3}$ of each other.* The value $\sqrt{3}$ is motivated by the distance between unlinked vertices in a right hexagon with side 1. Motivated by this wish, we set up the following energy function

$$\begin{aligned} U(\mathbf{x}, \mathbf{y}) = \frac{1}{2} \sum_{(i,j) \in E} \left[\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} - 1 \right]^2 \\ + \frac{1}{2} \sum_{(i,j) \notin E} \left[\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} - \sqrt{3} \right]_-^2, \end{aligned} \tag{4}$$

where $\mathbf{x}, \mathbf{y} \in \mathbb{R}^N$, and

$$\left[\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} - \sqrt{3} \right]_- = \min \left\{ \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} - \sqrt{3}, 0 \right\} \tag{5}$$

The first sum is the sum of spring energies for all edges of the graph while the second sum is the sum of all repelling spring energies of unlinked vertices. The corresponding force,

$$f(\mathbf{x}, \mathbf{y}) = -\nabla U(\mathbf{x}, \mathbf{y}),$$

is implemented in Matlab and Python functions called `forces`. To avoid large force when the linked vertices are far from each other, these routines cap the absolute value of each component of the attracting forces due to links by 1.

Task. Write a code that computes the positions of vertices of a graph and draws it. You need to choose any appropriate deterministic optimizer *except for the gradient descent* out of those that we have studied (BFGS line search, BFGS trust region, Nesterov, Adam, Levenberg-Marquardt). The graph to test your code is defined by an adjacency matrix in the file `Adjacency_matrix.csv`. The graph has $N = 91$ vertices. The initial positions of the vertices can be chosen e.g. as Gaussian random variables with mean zero and standard deviation N. You should obtain something like the graph in Fig. 1. My graph plotting

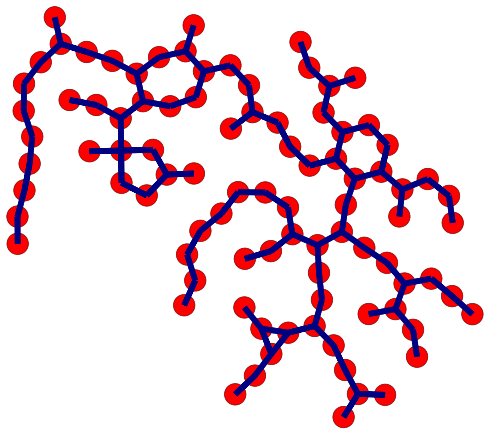

Figure 1: A depiction of the graph in Problem 2. This graph is a sample of the carbon skeleton of a hydrocarbon molecule.

routines are provided in Matlab and Python functions `plot_graph`. Please feel free to write your own plotting routine. In this case, make sure to set an equal aspect ratio and turn the axes off. Also, you are welcome to modify the energy function in any reasonable manner as soon as the resulting graph depiction satisfies the guideline highlighted in italics above.

Submit the resulting figure of the graph, specify which optimizer you implemented, and include the progress report for the optimizer. Plot the norm of the force versus iteration number in log scale along the y-axis and print the norm of the residual every once in a while. Link your code to your pdf file.

3 Problem 3.

(10 pts)

1. Consider a 2D Ising model on a square lattice with periodic boundary conditions. Find the change in energy, $\Delta H = H_{new} - H$, resulting from flipping a spin at site (i, j). Your expression should involve only the spin at (i, j) and its nearest neighbors.

2. Write a code to compute the mean magnetization $\mu(\beta)$ for the 2D Ising model by the Metropolis algorithm. Use a 30×30 lattice with the periodic boundary conditions, i.e., the nearest neighbors of site (i, j) where $0 \le i, j \le 29$ are $(i \pm 1 \mod 30, j \pm 1 \mod 30)$. Your program should compute the running mean and the running variance of the magnetization. These quantities are defined, respectively, by

$$\bar{m}_k = \frac{1}{k}\sum_{i=1}^{k} m_i, \quad [\mathrm{Var}(m)]_k = \frac{1}{k-1}\sum_{i=1}^{k}(m_i - \bar{m}_i)^2,$$

   where k is the current number of iterations. Note that you do not need to store all values of the magnetization. You need only to update $\bar{m}_k$ and $\mathrm{Var}(\bar{m})_k$ using the new value of m and the running mean $\bar{m}_k$ at each iteration.

3. Recall that the analytic expression for the mean magnetization is

$$\mu(\beta) = \begin{cases} \left(1 - [\sinh(2\beta)]^{-4}\right)^{1/8}, & \beta > 1/T_c = 0.4408, \\ 0, & \beta < 1/T_c = 0.4408. \end{cases} \tag{6}$$

   Use your code to calculate μ for the set of values of $\beta =$ `0.2:0.01:1`. Initially, set all spins up for each value of β. Desirably, use $k_{\max} = 10^8$ iterations for each value of β. You can use a smaller number of iterations while debugging and then run your code overnight with $k_{\max} = 10^8$. If your computer is slow, please feel free to reduce $k_{\max}$ appropriately.

   Plot the graph of the computed mean magnetization as a function of β and superimpose it with the graph of $\mu(\beta)$ given by Eq. (6). Also plot $\bar{m} \pm \sqrt{\mathrm{Var}(m}$.

Submit your derivation for Question 1, your plot, and link your code to your pdf file.