DS BMED 200, Winter 2024
Problem Set 1: Linear Algebra
Due October 25th, 2024 at 11:59pm PST

Submission instructions

- Submit your solutions electronically on the course Gradescope site as PDF files.
- Submit through the BruinLearn course website under the Gradescope tab on the left. You can add yourself to the course Gradescope site by going to gradescope.com, clicking "Add a course" and entering the following entry code: XG54ND.
- Please provide short and concise answers. Long, cumbersome, or unclear answers will not be checked.
- If you plan to typeset your solutions, please use the LaTeX solution template. If you plan to submit scanned handwritten solutions, please use a black pen on blank white paper and a high-quality scanner app.
- **The purpose of this problem set is to practice matrix calculation by hand. Unless the problem states otherwise, please evaluate all necessary quantities in your derivation by hand and implement code from scratch (as opposed to using Python or R's built-in functions).**

1 Matrix Calculation [8 pts]

(a) [2 pts] What is $\left(\left(\begin{bmatrix}1 & 3\\ 2 & 4\\ 0 & 2\end{bmatrix} + \begin{bmatrix}2 & 1\\ 0 & 3\\ 1 & 1\end{bmatrix}\right)\begin{bmatrix}4 & 1\\ 2 & 3\end{bmatrix}\right)\begin{bmatrix}0 & 1 & 0\\ 1 & 0 & 1\end{bmatrix}$

(b) [2 pts] What is the trace of $\boldsymbol{A} = \begin{bmatrix}8 & -4 & 0 & -10\\ -2 & 8 & -19 & -10\\ -2 & -7 & -6 & -7\\ -16 & -7 & 10 & 12\end{bmatrix}$

(c) [2 pts] In the lecture, we have seen several vector norms, including ℓ_1 and ℓ_2. We can further extend this notation to any arbitrary non-negative integer p. Formally, ℓ_p norm is defined as $\|\boldsymbol{x}\|_p = \sqrt[p]{\sum_i |x_i|^p}$. What is $\|\boldsymbol{x}\|_3$, where $\boldsymbol{x} = \begin{pmatrix}2 & -3 & 1 & 1 & 3 & 2 & 2 & 1\end{pmatrix}^T$

(d) [2 pts] What is the Frobenius norm of $\boldsymbol{A} = \begin{bmatrix} 0 & -1 \\ 4 & 8 \end{bmatrix}$

2 Linear Algebra [16 pts]

(a) [3 pts] Are the columns of matrix $\boldsymbol{A}$ linearly independent? $\boldsymbol{A} = \begin{bmatrix} -4 & -5 & 1 \\ 9 & 0 & -6 \\ -2 & 13 & 5 \\ 0 & -6 & -2 \end{bmatrix}$

(b) [3 pts] Show that matrix $\boldsymbol{A}$ is orthogonal but not orthonormal $\boldsymbol{A} = \begin{bmatrix} -1 & 2 & -2 \\ 2 & 2 & 1 \\ -2 & 1 & 2 \end{bmatrix}$.

(c) [2 pts] Find a constant $\alpha \in \mathbf{R}$ such that $\alpha \boldsymbol{A}$ is orthonormal in the previous question.

(d) [3 pts] Find the general representation of the null space of matrix $\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & -2 \\ 0 & 0 & 0 \end{bmatrix}$

(e) [3 pts] Find the general representation of the span of the column vectors in the matrix

$$\boldsymbol{A} = \begin{bmatrix} -1 & 0 & 1 \\ 1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & -1 \end{bmatrix}$$

(f) [2 pts] Explain why the vector $\boldsymbol{x} = \begin{pmatrix} 2 & 7 & 2 & -3 \end{pmatrix}^T$ is in or is not in the span of the matrix $\boldsymbol{A}$ in the previous question.

3 The gradient and hessian [8 pts]

(a) [2 pts] What is the gradient of $f(x_1, x_2) = 2x_1^4 + x_1^2 - 3x_1x_2 + 2x_1^2x_2 - 7$

(b) [2 pts] What is the gradient of $f(x_1, x_2) = \log(3x_1^3 - x_2^2 - 1)$

(c) [2 pts] What is the hessian of $f(x_1, x_2) = 1 - e^{x_1^2} + 2{x_2}^3$

(d) [2 pts] What is the hessian of $f(x_1, x_2) = 10{x_1}^2 - 7x_1x_2^2 + {x_2}^2 - 30$

4 Implementation: Linear Algebra [10 pts]

(a) [6 pts] Using the definition from lecture 1, write a recursive function that calculates the determinant of an n by n matrix. **Please refrain from using any built-in function related to matrix calculation; using matrix data structures (e.g., numpy arrays) is allowed.** Attach a screenshot of your implementation in the blank space below.

(b) [4 pts] Write a function that checks whether an input matrix is orthogonal. If so, your function should further check whether it is orthonormal. **You may use basic built-in matrix calculation and matrix slicing/indexing functions as helper functions.** Attach a screenshot of your implementation in the blank space below.

5 Implementation: gradient descent [15 pts]

We have briefly touched upon the compact matrix notation of systems of linear equations in the first lecture. One common but slightly different case involves solving a system where we have more equations/number of observations than parameters/number of features. This is often cast as a least squares problem, where instead of finding an exact solution for the variables, we are interested in finding a solution that gives us the smallest overall error in terms of least squares.

Let $\boldsymbol{X} \in \mathbf{R}^{n \times m}$ be a matrix where rows indicate samples and columns indicate features, and let the vector $\boldsymbol{y} \in \mathbf{R}^n$ denote a *response variable* (often called the "outcome"). We will use $\hat{y}_i = x_i^T \beta$ as our prediction of y_i, where x_i denotes here the i^{th} row of matrix $\boldsymbol{X}$ and β denotes the coefficients associated with all the features present in matrix $\boldsymbol{X}$. We will use the residual sum of squares (RSS) as the measurement of the discrepancy between the true $\boldsymbol{y}$ and our linear prediction $\hat{\boldsymbol{y}}$. More formally, we want to solve the following minimization problem:

$$\arg\min_{\boldsymbol{\beta}} \text{RSS}(\boldsymbol{\beta}) \tag{1}$$

where

$$\text{RSS}(\boldsymbol{\beta}) = \sum_{i=1}^{i=n} (y_i - x_i^T \boldsymbol{\beta})^2 = \|(\boldsymbol{y} - X\boldsymbol{\beta})\|_2^2$$

One computationally efficient approach to optimize the least squares objective is using gradient descent. Let α denote the step size. The update at iteration $t+1$ is given by

$$\begin{aligned} \boldsymbol{\beta}^{(t+1)} &= \boldsymbol{\beta}^{(t)} - \alpha \frac{\partial \text{RSS}}{\partial \boldsymbol{\beta}} \\ &= \boldsymbol{\beta}^{(t)} - \alpha \sum_{i=1}^{i=n} 2(y_i - x_i^T \boldsymbol{\beta})(-x_i) \end{aligned}$$

In this question, you will implement gradient descent for solving the least squares problem defined in Equation (1). We will use the data provided with this assignment: the samples by features data matrix X is provided in the file X.csv, and the response variable y is provided in the file Y.csv.

Implementation note:

- Your gradient descent algorithm should always start with the vector $\vec{0} = (0, ..., 0)$ as the initial point.
- Your algorithm should have at most 10^4 updates (i.e., iterations). Setting a hard maximum threshold can prevent your code from running into an infinity loop.
- Your algorithm should stop when the difference between the objective function (i.e., $RSS(\beta)$) from the current and previous iteration is smaller than 10^{-8}.
- You may use built-in matrix calculation functions when implementing your algorithm.

You do not need to provide screenshots of your code. Instead, please answer the following questions.

(a) [5 pts] We will now explore the behavior of the gradient descent optimizer under different learning rates α: $10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}$. Plot the RSS from the first 100 iterations under each of the learning rates above in one plot and comment on the differences between the learning rates.

(b) [5 pts] What happens if you use larger learning rates (e.g., 0.05, 0.1, etc.)? Please generate a plot similar to the one in (a) using several larger learning rates, this time **only for the first 10 iterations**. Briefly describe what you observe and provide an explanation. (Hint: It might be useful to plot the RSS in log scale to visualize curves associated with different learning rates.)

(c) [5 pts] Least squares sometimes can be solved analytically using the following closed-form solution:

$$\hat{\boldsymbol{\beta}} = (X^T X)^{-1} X^T \boldsymbol{y}$$

Confirm that in our dataset $X^T X$ is non-singular and thus its inverse exists. You can do so by calling your determinant function (part 4.a). Next, please implement the closed-form solution as well. Compare the result with those you obtained by performing gradient descent using various learning rates. Comment on any discrepancy you observe.