ENMG 616 Advanced Optimization & Techniques

Homework Assignment 2

Due Date: October 15, 2023

You need to show the steps followed to get the final answer (Do not just give the final result). The homework should be submitted to Moodle as ***one*** *pdf file. Please insert a copy of your Matlab code in the submitted file.*

1 Linear Regression using Real Dataset

Simple linear regression is a linear predictive modeling technique that aims at finding significant relationship between a dependent feature $\mathbf{x}$ a target variable y. The relationship is modeled using the following linear predictor function

$$\widehat{y}_{\boldsymbol{\theta}}(\mathbf{x}) = \theta_0 + \theta_1\, x = \begin{bmatrix} x & 1 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_0 \end{bmatrix} = \mathbf{x}^T \boldsymbol{\theta},$$

where

$$\mathbf{x} = \begin{bmatrix} x \\ 1 \end{bmatrix}, \quad \text{and} \quad \boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \theta_0 \end{bmatrix}$$

Our goal is to find the best fitting line that minimizes the sum of squared errors (SSE) or mean squared error (MSE) between our target variable (y) and our predicted output across all samples. More specifically, we aim at minimizing

$$SSE = \frac{1}{n}\sum_{i=1}^{n} (\text{predicted } \widehat{y}_i - \text{actual } y_i)^2 = \frac{1}{n}\sum_{i=1}^{n} (\widehat{y}_{\boldsymbol{\theta}}(\mathbf{x}_i) - y_i)^2 = \frac{1}{n}\sum_{i=1}^{n} \left(\mathbf{x}_i^T \boldsymbol{\theta} - y_i\right)^2.$$

We will solve the optimization problem detailed above on a dataset that collected to study the relationship between income (one unit = \$ 10000) and happiness (range between 0 and 10). In the following steps, we will use Matlab to apply gradient descent for solving the corresponding optimization problem.

1. Download the data files *Xtrain* and *Ytrain*. ✓
2. Import the dependent feature *Xtrain* and target variable *Ytrain* using the following Matlab function: X = importdata('Xtrain.mat'), Y = importdata('Ytrain.mat'). ✓
3. Using the first order optimality condition, compute the optimal solution of the above quadratic optimization problem. ✓
4. Compute the Lipschitz constant L by finding the largest eigenvalue of $\mathbf{X}\,\mathbf{X}^T$; i.e. max(eig($\mathbf{X}*\mathbf{X}'$)). ✓

5. Implement classical gradient descent algorithm with constant step-size $1/L$, find the optimal $\boldsymbol{\theta}$ and compare it with the optimal solution computed in part 4. ✓

6. Plot the scatter of points along with the linear model, i.e scatter(Xtrain, Ytrain); hold on; plot([0:0.5:10],θ_0 + θ_1*[0:0.5:10]) ✓

7. Comment on the results. ✓

1.1 Effect of the Lipschitz Constant

In this section, we will normalize the data to reduce the largest eigenvalue of the quadratic problem and then run the same experiment.

1. Compute the maximum value of *Xtrain*. ✓

2. Divide both *Xtrain* and *Ytrain* by the maximum value. ✓

3. Compute the resulting Lipschitz constant. ✓

4. Implement classical gradient descent algorithm with constant step-size, find the optimal $\boldsymbol{\theta}$. ✓

5. Comment on the effect of the Lipschitz on the choice of step-size, condition number, and convergence of the algorithm. ✓

2 Gradient Descent

Consider the following quadratic optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} = \frac{1}{2}\mathbf{x}^T\,\mathbf{Q}\,\mathbf{x} + \mathbf{q}^T\,\mathbf{x}.$$

1. Randomly generate $\mathbf{Q}$ and $\mathbf{q}$ according to the following procedure:

   (a) Set $n = 50$;

   (b) Generate a random matrix $\mathbf{B} \in \mathbb{R}^{n\times n}$ where each element is drawn from standard normal distribution. Matlab Function: $\mathbf{B} = \text{randn}(n, n)$.

   (c) Let $\mathbf{Q} = \mathbf{B} * \mathbf{B}'$. ✓

   (d) Find the smallest eigenvalue of $\mathbf{Q}$. Matlab Function: MinEig = min(eig($\mathbf{Q}$)).

   (e) To have $\mathbf{Q}$ a PSD matrix, add its smallest eigenvalue plus some uniform random variable. Matlab Function: $\mathbf{Q} = \mathbf{Q}$ + (MinEig + rand(2))*eye(n).

   (f) Generate a random vector $\mathbf{q}$ where each element is drawn from normal distribution $\mathcal{N}(0, 100)$. Matlab Function: $\mathbf{q} = 10 * \text{randn}(n, 1)$.

2. Using first order optimality condition, compute the optimal solution of the above quadratic optimization problem. ✓

3. Implement classical gradient descent algorithm with these step-size selection rules:

   (a) Exact Line Search ✓

   (b) Armijo Rule. ✓

   (c) Diminishing step-size: $\alpha_r = 0.1/r$ ✓

(d) Constant step-size: $\alpha = 1/L$ where L is the Lipschitz constant of the gradient of the objective function. ✓

4. Implement Newton's method. ✓
5. Implement Accelerated Gradient Descent method. ✓
6. Run the above 6 algorithms from the point $\mathbf{x}^0 = 0$ for 1000 iterations, and report the following log-relative error ratio

$$\varepsilon = \log\left(\frac{\|\mathbf{x}^{1000} - \mathbf{x}^*\|}{\|\mathbf{x}^0 - \mathbf{x}^*\|}\right).$$ ✓

7. Repeat the above experiment 5 times (5 random $\mathbf{Q}$ and $\mathbf{q}$) and report ε for each realization each algorithm by filling the tables below:

	Realization 1	Realization 2	Realization 3	Realization 4	Realization 5
Exact Minimization					
Armijo					
Diminishing					
Constant					
Newton					
Nesterov					
Kappa					

Table 1: Log-relative Error for 5 realizations

	Average Log-relative Accuracy
Exact Minimization	
Armijo	
Diminishing	
Constant	
Newton	
Nesterov	

Table 2: Average log-relative error for various algorithms

2.1 Effect of the condition number

Generate $\mathbf{Q}$ according to the following steps $\mathbf{B} = \text{randn}(n, n)$; $\mathbf{Q} = \mathbf{B}*\mathbf{B}'$; $\text{MinEig} = \min(\text{eig}(\mathbf{Q}))$; $\mathbf{Q} = \mathbf{Q} + (\text{MinEig} + 10)*\text{eye}(n)$.

Repeat Problem 2. Notice that this choice of $\mathbf{Q}$ generally leads to a smaller condition number. by filling the tables below:

	Realization 1	Realization 2	Realization 3	Realization 4	Realization 5
Exact Minimization					
Armijo					
Diminishing					
Constant					
Newton					
Nesterov					
Kappa					

Table 3: Log-relative Error for 5 realizations

	Average Log-relative Accuracy
Exact Minimization	
Armijo	
Diminishing	
Constant	
Newton	
Nesterov	

Table 4: Average log-relative error for various algorithms

2.2 Effect of restart in accelerated Nesterov

Generate an instance of Problem 2.1. Plot $\log\left(\frac{\|\mathbf{x}^k - \mathbf{x}^*\|}{\|\mathbf{x}^0 - \mathbf{x}^*\|}\right)$ versus iteration number k for the following algorithms:

1. Accelerate Nesterov method with no restart.
2. Accelerate Nesterov method with restart at every $T = \sqrt{\kappa}$ iterations.
3. Accelerate Nesterov method with restart at every $T = 5\sqrt{\kappa}$ iterations.
4. Accelerate Nesterov method with restart at every $T = 20\sqrt{\kappa}$ iterations.

Comment on the results in few sentences.