STAT 102B: Sample Exam I Questions

George Michailidis

1 Problems that require calculations

Problem 1:

Let $f(x) : (0, +\infty) \rightarrow \mathbb{R}$ with

$$f(x) = x - \log(x),$$

where $\log(\cdot)$ denote the **natural base** e logarithm

1. Show that $f(x)$ has a **unique global minimum** in $(0, +\infty)$. Justify your answer.
2. Let $x_0 = 2$ be the initial point used in the gradient descent algorithm. What will x_1 be based on the gradient descent algorithm, if the step size is set to $\eta = 0.5$?
3. Derive the range of values for the step size parameter η, so that gradient descent is convergent? Justify your answer.
4. Suppose that somebody that does not know how to derive the range of eligible step sizes η, decided to use an initial $\eta = 2$ for initial point $x_0 = 2$. Explain what calculations the backtracking line search algorithm will check to select and appropriate step size η_0 to proceed calculating the next update x_1.

Answer:

(1) $\frac{df}{dx} = 1 - \frac{1}{x}$. Setting $df/dx = 0$ yields $\hat{x} = 1$. This is a global minimum since $\frac{d^2 f}{dx^2} = \frac{1}{x^2} > 0$ throughout the domain of the function f.

(2) $x_1 = x_0 - \eta f'(x_0) = 2 - 0.5 \times (1 - \frac{1}{2}) = 2 - 0.5 \times 0.5 = 1.75$.

(3) Let $x_0 \in (0, \infty)$ be the initial value used in the gradient descent algorithm.

We have calculated that $d^2 f/dx^2 = \frac{1}{x^2}$. Note that for any $x_0 > 1$,

$$\frac{d^2 f}{dx^2} = \frac{1}{x^2} < 1.$$

Hence, for **any** $x_0 > 1$, any $\eta \in (0, \frac{1}{1}) = (0, 1)$ would guarantee convergence; i.e., the Lipschitz constant for the function under consideration is $L = 1$ in the interval $(1, \infty)$.

Then, note that the gradient descent update will be:

$$x_{k+1} = x_k - \eta\left(1 - \frac{1}{x_k}\right) = x_k + \eta\left(\frac{1}{x_k} - 1\right)$$

For **any** $x_k \in (0, 1)$ (and hence x_0), it can be seen that the term in parenthesis will be positive and hence $x_{k+1} > x_k$. It may be the case that x_{k+1} will become greater than 1 and therefore the previous analysis will dictate a value of $\eta \in (0, 1)$.

Hence, any value of $\eta \in (0, 1)$ would guarantee convergence irrespective of the initial value x_0.

(4) The backtracking line search algorithm will check the first Armijo condition; namely

$$h(\eta) \leq h(0) + \epsilon \eta h'(0),$$

with $\epsilon \in (0, 1)$ and $h(\eta) = f(x_k - \eta f'(x_k))$.

If the condition is satisfied, then the current value of $\eta_{\text{current}} = 2$ will be retained, otherwise a new value $\eta_{\text{new}} = \tau \eta_{\text{current}}$ will be tested ($\tau \in (0, 1)$).

With $\eta = 2$ and $x_0 = 2$, the left hand side of the condition becomes

$$h(\eta) = f(x_0 - \eta f'(x_0)) = x_0 - \eta(1 - \frac{1}{x_0}) - \log\left(x_0 - \eta(1 - \frac{1}{x_0})\right) =$$

$$2 - 2(1 - \frac{1}{2}) - \log\left(2 - 2(1 - \frac{1}{2})\right) = 1 - \log(1) = 1$$

For the right hand side of the condition the following calculations are helpful:

$$h(0) = x_0 - \eta(1 - \frac{1}{x_0}) - \log(x_0 - \eta(1 - \frac{1}{x_0})) = 2 - \log(2).$$

$$h'(\eta) = \frac{d}{d\eta}\left[x_0 - \eta\left(1 - \frac{1}{x_0}\right) - \log\left(x_0 - \eta(1 - \frac{1}{x_0})\right)\right] =$$
$$-\left(1 - \frac{1}{x_0}\right) + \frac{1}{x_0 - \eta(1 - \frac{1}{x_0})}(1 - \frac{1}{x_0})$$

Evaluating the derivative $h'(\eta)$ at $\eta = 0$ yields

$$(1 - \frac{1}{x_0})(\frac{1}{x_0} - 1) = 0.5(-\frac{1}{2}) = -0.25.$$

Hence, the right hand side of the condition becomes

$$2 - \log(2) - 0.25(2)\epsilon$$

Hence, if $\epsilon = 0.5$, then the right hand side of the Armijo condition becomes

$$2 - \log(2) - 0.25 \times 2 \times 0.5 \approx 1.449.$$

Since the left hand side (=1) is smaller than the right hand size (=1.449), the current value for $\eta = 2$ will be retained to determined the first update x_1.

Problem 2:

Let $n \geq 1$ be an integer and let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix (not necessarily positive definite) for which all of its eigenvalues are non-zero. Let $a \in \mathbb{R}^m$ be a given vector and we consider the function $f : \mathbb{R} \to \mathbb{R}$ given by

$$f(x) = \frac{1}{2}(x-a)^\top AA(x-a),$$

1. Does the function $f(x)$ have a minimizer? If yes, derive it, otherwise argue why not. Show your work.
2. If one decides to use the gradient descent algorithm, how should η be selected? Justify your answer.

Answer:

(1) A is an $m \times m$ symmetric matrix, hence $A^\top = A$.

Define $Q \equiv AA \equiv A^\top A$.

From the definition of a positive definite matrix, Q will be positive definite if and only if for any vector $z \in \mathbb{R}^m$ with $z \neq 0$, $z^\top Qz > 0$. Note that

$$z^\top Qz = z^\top A^\top Az = u^\top u > 0,$$

unless $u = Az = 0$, which imply that $z = 0$, which is excluded from the definition of positive definiteness of a matrix.

Hence, $Q \equiv A^\top A \equiv AA$ is positive definite.

Then, $\nabla f(x) = Q(x-a)$ and setting it to zero yields $\hat{x}_{\min} = a$.

The latter is a global minimum, since Q (the Hessian of $f(x)$) is positive definite.

(2) Since $f(x)$ is a quadratic function and Q is positive definite, the interval of admissible constant step sizes η that would guarantee convergence is given by

$$\eta \in \left(0, \frac{2}{\lambda_{\max}(Q)}\right)$$

Problem 3:

Consider the function $f : \mathbb{R} \to \mathbb{R}$ with

$$f(x) = (x+2)^2$$

1. Write pseudo-code that implements the gradient descent algorithm, with **optimal selection** of the step size η.
2. If $x_0 = 1$ was selected as the initial point, what would the value of the optimal η be to calculate x_1 based on the gradient descent algorithm?
3. Based on the optimal step size derived in the previous part, calculate the value of x_1.
4. What are the admissible values of η that would lead to a convergent sequence of iterates x_k for the gradient descent algorithm with fixed step size? Justify your answer.

Answer:

(1) The gradient of $f(x)$ is $f'(x) = 2(x+2)$ and $f''(x) = 2 > 0$, hence a global minimum exists ($\hat{x}_{\text{min}} = -2$).

Let $h(\eta) = f(x_k + -\eta f'(x_k))$.

Then, $h(\eta) = f(x_k - \eta f'(x_k)) = f(x_k - 2\eta(x_k+2))$ (check slide 7 Lecture 2.1). Then,

$$\eta_k = \text{argmin}_{\eta>0} f(x_k - 2\eta(x_k+2)) = \text{argmin}_{\eta>0} f(x_k(1-2\eta) - 4\eta)$$

To find the optimal η we take the derivative of $h(\eta)$ with respect to η and set it equal to 0; i.e.,

$$\frac{dh}{d\eta} = \frac{d}{d\eta}(x_k(1-2\eta) - 4\eta + 2)^2 \quad \underset{=}{\text{chain rule}} \quad 2(x_k(1-2\eta) - 4\eta + 2)(-2x_k - 4)$$

We then set

$$\frac{dh}{d\eta} = 0 \Longrightarrow 2(x_k(1-2\eta) - 4\eta + 2)(-2x_k - 4) = 0 \Longrightarrow x_k(1-2\eta) - 4\eta + 2 = 0$$

$$\implies \eta = \frac{2 + x_k}{4 + 2x_k} = \frac{1}{2}.$$

Pseudo-code:

1. $f(x) = (x+2)^2$
2. $f'(x) = 2(x+2)$
3. Set $\eta = \frac{1}{2}$
4. Set tolerance=1e-10
5. Initialize gradient descent by selecting x_0
6. While $|f(x_{k+1} - f(x_k)| >$ tolerance do

$$x_{k+1} = x_k - \eta f'(x_k)$$

(2) The optimal $\eta = 1/2$ **for this function**, irrespective of the initial value x_0, as the derivations above show.

(3) $x_1 = x_0 - 1/2 f'(x_0) = 1 - \frac{1}{2} \times 2 \times (1+2) = 1 - 3 = -2$ the theoretical optimal value. Hence, gradient descent will converge in one iteration.

(4) Since $f(x)$ is a quadratic function, the interval of admissible constant step size η that guarantees convergence is determined as follows:
Write $f(x) = (x+2)^2 = x^2 + 2x + 4 = \frac{1}{2}(2x^2) + 2x + 4$, so the Hessian is equal to 2 (a trivial 1×1 matrix) whose maximum eigenvalue is 2. Hence, the required interval is $(0, \frac{2}{2}) = (0, 1)$.

Next, some plots of the behavior of the GD algorithm are given selected from the range $(0, 1)$.

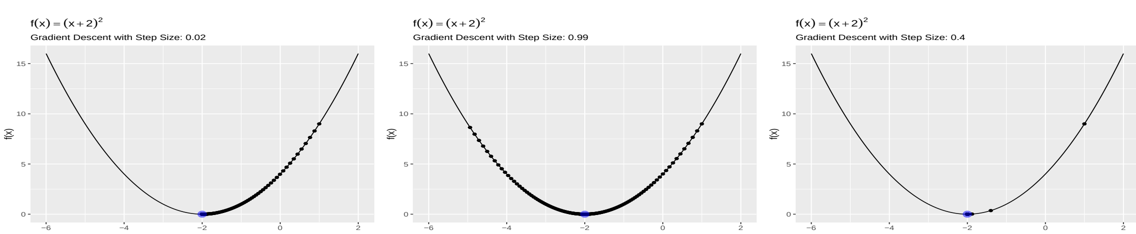

Figure 1: Behavior of GD for different permissible step sizes η

Problem 4:

Somebody run gradient descent for 20 iterations with $\eta = 0.3$ and computes the update x_k and the norm of the gradient $\nabla f(x_k)$; i.e, $||\nabla f(x_k)||_2$ after each iteration.

It is observed that the norm $||\nabla f(x_k)||_2$ decreases quickly and then levels off. Based on this, which of the following would be your recommendation:

- Use a larger value of η; e.g., $\eta = 0.5$.
- Use a larger value of η; e.g., $\eta = 0.1$.
- Keep $\eta = 0.3$.
- Additional information is needed to make a recommendation. What information would you need?

Justify your answer.

Answer:

We need additional information. From the problem description, we know that the step size used in the first 20 iterations is $\eta = 0.3$, but we do not know if that is the result of backtracking line search, or it was actually set to a constant value $\eta = 0.3$.

Let us examine the other possibilities. The choice $\eta = 0.5$ is probably a poor one, since the norm of the gradient is becoming smaller, which means that smaller step sizes are required to reach the minimum. A larger step size would make gradient descent to overshoot the "target". A smaller $\eta = 0.1$ is probably a safe choice, although it may require more iterations for gradient descent to converge. Keeping $\eta = 0.3$ may still work. For example, if the function under consideration was a quadratic one and $\eta = 0.3$ was within the admissible interval $(0, \frac{2}{\lambda_{\max}(H)})$, with H denoting the Hessian of the function, then keeping $\eta = 0.3$ would be perfectly fine.

Problem 5:

Suppose you have a data set comprising of 1 million observations and 100,000 predictors. You want to use multivariate linear regression to estimate the 100,000 regression coefficients from the data.

1. Should you prefer the closed form least squares solution or use the gradient descent algorithm? Justify your answer.
2. If you decided to use the gradient descent algorithm, would you be able to find the optimal regression coefficients? Justify your answer.
3. What strategy would you use to select the step size in the gradient descent algorithm? Justify your answer.

Answer:

1. To compute the least squares solution involves computing:
   - $X^\top X \in \mathbb{R}^{p\times p}$: $\quad \mathcal{O}(np^2)$
   - $X^\top y \in \mathbb{R}^p$: $\quad \mathcal{O}(np)$
   - Solving the system $X^\top X\beta = X^\top y$: $\quad \mathcal{O}(p^3)$ – requires computing the inverse of $X^\top X$

   Therefore, the total computational complexity is:

   $$\boxed{\mathcal{O}(np^2 + p^3)}$$

   For $n = 10^6$ and $p = 10^5$:

   $$np^2 = 10^6 \cdot (10^5)^2 = 10^{16}, \quad p^3 = (10^5)^3 = 10^{15}$$

   Thus, the dominant term is $\mathcal{O}(np^2) = \boxed{10^{16}}$.
   On the other hand, the gradient of the SSE function is:

   $$\nabla_\beta \|X\beta - y\|_2^2 = X^\top (X\beta - y)$$

   Computational steps:

   - Compute $X\beta$: $\quad \mathcal{O}(np)$

- Compute residual $X\beta - y$: $\mathcal{O}(n)$
- Compute $X^\top(X\beta - y)$: $\mathcal{O}(np)$

Thus, the total computational cost **per iteration** of gradient descent is:

$$\boxed{\mathcal{O}(np)}$$

For $n = 10^6$ and $p = 10^5$:

$$np = 10^6 \cdot 10^5 = \boxed{10^{11}}.$$

With a good choice of the step size, the number of iterations would be of the order of $10^2 - 10^3$ (hundreds to thousands of iterations), so the total computational cost would be at most $\boxed{10^{14}}$.

This is the reason that for large scale regression problems, gradient descent is preferable to the closed form least squares solution.

2. We would be able to find the optimal regression coefficients **provided** $(X^\top X)$ is **positive definite**. Recall that the SSE objective function is a quadratic one, whose Hessian is given by $X^\top X$. If the latter is positive definite, then a global minimum exists and gradient descent with a careful choice of the step size will converge to it. Further, recall that $X^\top X$ is required to be positive definite for the inverse (used in the least squares formula) to exist.

3. Backtracking line search, which does not require any other information. Recall that a step size selected in the interval $(0, \frac{2}{\lambda_{\max}(X^\top X)})$ would guarantee convergence, but calculating the maximum eigenvalue of $X^\top X$ is computationally expensive – requires $\mathcal{O}(p^3)$ computations, which defeats the purpose of using gradient descent in the first place.

Problem 6: Write pseudo-code for a function $f : \mathbb{R}^m \to \mathbb{R}$ to perform gradient descent based on the backtracking line search algorithm.

Answer:

Algorithm 1 Gradient Descent with Backtracking Line Search

Require: Function $f : \mathbb{R}^m \to \mathbb{R}$, gradient ∇f, initial point x_0, initial step size $\eta_0 > 0$, parameters $0 < \tau < 1$, $0 < \epsilon < 1$, tolerance `tol`, maximum iterations K

Ensure: Obtain $x_{\min}$

$x \leftarrow x_0$
for $k = 1$ to K **do**
    $g \leftarrow \nabla f(x)$
    if $\|g\| <$ `tol` **then**
        return $x_{\min}$
    end if
    $\eta \leftarrow \eta_0$
    while $f(x - \eta g) > f(x) - \epsilon\, \eta\, \|g\|^2$ **do**
        $\eta \leftarrow \tau \eta$
    end while
    $x \leftarrow x - \eta g$
end for
return $x_{\min}$

Problem 7:

Consider the function $f(x) = \frac{1}{3}\sum_{i=1}^{3} \gamma_i(x-3)^2 + 3$, with $\gamma_1 = 1, \gamma_2 = 5, \gamma_3 = -2$.

Assuming that $x_0 = 0, \eta = 0.1$ and $\xi = 0.5$, calculate what x_3 will be for **stochastic gradient descent** coupled with Polyak momentum, if $I_1 = \{1\}, I_2 = \{2\}$ and $I_3 = \{3\}$.

Answer:

The gradient of $f_i(x)$ is:

$$f_i'(x) = \frac{2}{3}\gamma_i(x-3)$$

We initialize:

$$x_0 = 0$$

Iteration 1: Use $I_1 = \{1\}$

This will be a pure GD update.

$$f_1'(x_0) = \frac{2}{3} \cdot 1 \cdot (0-3) = -2$$
$$x_1 = x_0 - \eta f_1'(x_0) = 0 - 0.1(-2) = 0.2$$

Iteration 2: Use $I_2 = \{2\}$

$$f_2'(x_1) = \frac{2}{3} \cdot 5 \cdot (0.2-3) = \frac{10}{3} \cdot (-2.8) = -\frac{28}{3}$$
$$y_1 = x_1 + \xi(x_1 - x_0) = 0.2 + 0.5 \cdot (0.2 - 0) = 0.3$$
$$x_2 = y_1 - \eta f_2'(x_1) = 0.3 - 0.1 \cdot (-\frac{28}{3}) = 0.3 + \frac{28}{30} \approx 1.2333$$

Iteration 3: Use $I_3 = \{3\}$

$$f_3'(x_2) = \frac{2}{3} \cdot (-2) \cdot (1.2333 - 3) \approx 2.3556$$

$$y_2 = x_2 + \xi(x_2 - x_1) = 1.2333 + 0.5 \cdot (1.2333 - 0.2) = 1.7499$$
$$x_3 = y_2 - \eta f_3'(x_2) = 1.7499 - 0.1 \cdot 2.3556 \approx 1.5144$$

Final Answer:

$$\boxed{x_3 = 1.5144}$$

Problem 8:

Same setting as in Problem 6, but for **stochastic gradient descent** coupled with Nesterov momentum.

Answer:

The gradient of $f_i(x)$ is:

$$f_i'(x) = \frac{2}{3}\gamma_i(x-3)$$

We initialize:

$$x_0 = 0$$

Iteration 1: Use $I_1 = \{1\}$

This will be a pure GD update.

$$f_1'(x_0) = \frac{2}{3} \cdot 1 \cdot (0-3) = -2$$
$$x_1 = x_0 - \eta f_1'(x_0) = 0 - 0.1(-2) = 0.2$$

Iteration 2: Use $I_2 = \{2\}$

$$y_1 = x_1 + \xi(x_1 - x_0) = 0.2 + 0.5 \cdot (0.2 - 0) = 0.3$$
$$f_2'(y_1) = \frac{2}{3} \cdot 5 \cdot (0.3 - 3) = -9$$
$$x_2 = y_1 - \eta f_2'(y_1) = 0.3 - 0.1 \cdot (-9) = 1.2$$

Iteration 3: Use $I_2 = \{3\}$

$$y_2 = x_2 + \xi(x_2 - x_1) = 1.2 + 0.5 \cdot (1.2 - 0.2) = 1.7$$
$$f_3'(y_2) = \frac{2}{3} \cdot (-2) \cdot (1.7 - 3) \approx 1.7333$$
$$x_3 = y_2 - \eta f_3'(y_2) = 1.7 - 0.1 \cdot (1.7333) \approx 1.5267$$

Final Answer:

$$\boxed{x_3 = 1.5267}$$

2 Multiple choice Quiz Questions

Question 1: Which direction does the gradient descent algorithm move in each iteration?

- ☐ Random direction
- ☐ Direction of the gradient
- ☑ Opposite to the gradient
- ☐ Along the eigenvectors of the Hessian

Question 2: If the step size (learning rate) for the optimization problem of a statistical model is too large, gradient descent can:

- ☐ Converge slowly
- ☑ Not converge
- ☐ Overfit the data
- ☐ Always converge faster

Question 3: In the context of gradient descent, a "step size schedule" is used to:

- ☐ Randomly choose step sizes
- ☐ Ensure faster convergence
- ☑ Decrease step size over iterations
- ☐ Increase gradient magnitude

Question 4: In the heavy ball (Polyak) momentum method, the new direction is a combination of:

- ☐ Current gradient and noise

- ☑ Previous and current value of the parameter, plus the gradient evaluated at the current value
- ☐ Previous update and current gradient
- ☐ Gradient norm and function value

Question 5:

- ☐ It remains constant for every coordinate of the parameter x
- ☐ It increases with each iterate
- ☑ It decreases more for the coordinates of x with larger accumulated gradients
- ☐ It is chosen randomly at each iteration

Question 6: Why is bias correction used in ADAM?

- ☐ To make convergence faster
- ☑ To compensate for initialization at zero
- ☐ To prevent exploding gradients
- ☐ To add noise to the gradient update

Question 7: In ADAM, the step size is:

- ☐ Fixed
- ☐ Increasing
- ☑ Scaled by gradient history
- ☐ Decreased linearly

Question 8: Mini-batch SGD uses:

- ☐ The entire data set
- ☑ Only one sample
- ☑ A subset of data points
- ☐ Data sorted by the objective function value

Question 9: What is a common downside of very small batch size in SGD?

- ☐ Too slow
- ☐ Too stable
- ☑ High variance in updates
- ☐ Uses entire data set per iteration

Question 10:

Which of the following best describes the mechanics of gradient descent with step size determination based on AdaGrad?

- ☐ It increases the step size over iterations to make faster progress on flat regions of the objective function.
- ☐ It maintains a moving average of past gradients to determine the step size.
- ☑ It scales the step size based on the square root of accumulated squared gradients.
- ☐ It uses a momentum type mechanism to combine gradients from previous iterations.

Question 11:

What is a key characteristic of SGD?

- ☐ It uses a separate step size for each coordinate of the gradient of the objective function

- [ ] It uses information from second order partial derivative
- [ ] It updates the argument of the objective function using a batch of the full data set at each step
- [x] It updates the argument of the objective function using gradient information estimates from a small batch or a single example from the data set

Question 12:

What differentiates ADAM-W from the standard ADAM method?

- [ ] ADAM-W applies weight decay through gradient rescaling.
- [ ] ADAM-W introduces per-parameter adaptive step sizes.
- [x] ADAM-W decouples weight decay from the gradient update.
- [ ] ADAM-W uses an exponentially decaying step size schedule.

Question 13: In the context of binary classification using a Multi-Layer Perceptron (MLP), which of the following is typically used as the final activation function?

- [ ] ReLU
- [ ] Tanh
- [ ] Softmax
- [x] Sigmoid

Question 14:

In the context of training an MLP, what is an *epoch*?

- [ ] A single forward pass through one sample
- [ ] A single backward pass through the network
- [x] One complete pass through the entire training data set
- [ ] The number of neurons in the hidden layer