EECS 127/227AT Optimization Models in Engineering UC Berkeley Spring 2024

Homework 6

This homework is due at 11 PM on February 23, 2024.

Submission Format: Your homework submission should consist of a single PDF file that contains all of your answers (any handwritten answers should be scanned).

1. **Condition Number**

   In lecture, we examined the sensitivity of solutions to linear system $A\vec{x} = \vec{y}$ (for nonsingular/invertible square matrix A) to perturbations in our measurements $\vec{y}$. Specifically, we showed that if we model measurement noise $\Delta\vec{y}$ as a linear perturbation on $\vec{y}$, resulting in a linear perturbation $\Delta\vec{x}$ on $\vec{x}$ — i.e., $A(\vec{x} + \Delta\vec{x}) = \vec{y} + \Delta\vec{y}$ — we can bound the magnitude of the solution perturbations $\Delta\vec{x}$ as

$$\frac{\|\Delta\vec{x}\|_2}{\|\vec{x}\|_2} \leq \kappa(A)\frac{\|\Delta\vec{y}\|_2}{\|\vec{y}\|_2}, \tag{1}$$

   where $\kappa(A) = \frac{\sigma_{\max}\{A\}}{\sigma_{\min}\{A\}} = \|A\|_2 \left\|A^{-1}\right\|_2$ is the condition number of A, or the ratio of A's maximum and minimum singular values. In this problem, we will establish a similar bound for perturbations on A.

   (a) Consider the linear system $A\vec{x} = \vec{y}$ above, where $A \in \mathbb{R}^{n\times n}$ is invertible (i.e., square and nonsingular). Let $\Delta A \in \mathbb{R}^{n\times n}$ denote a linear perturbation on matrix A generating a corresponding linear perturbation $\Delta\vec{x}$ in solution $\vec{x}$, i.e.,

$$(A + \Delta A)(\vec{x} + \Delta\vec{x}) = \vec{y}. \tag{2}$$

   Show that

$$\frac{\|\Delta\vec{x}\|_2}{\|\vec{x} + \Delta\vec{x}\|_2} \leq \kappa(A)\frac{\|\Delta A\|_2}{\|A\|_2}. \tag{3}$$

   (b) Note that Equations (1) and (3) above bound two slightly different quantities: $\frac{\|\Delta\vec{x}\|_2}{\|\vec{x}\|_2}$ and $\frac{\|\Delta\vec{x}\|_2}{\|\vec{x} + \Delta\vec{x}\|_2}$, respectively. In general, we wish to establish these bounds because we want to characterize the size of $\Delta\vec{x}$ under different sizes of perturbation. Which of these two bounds better serves this purpose? Consider the following two cases. (i) $\Delta\vec{x} \gg \vec{x}$ (ii) $\vec{x} \gg \Delta\vec{x}$ and answer whether Equation (1) or Equation (3) is better.

2. Ridge Regression for Bounded Output Perturbation

We will first solve the ridge regression problem in the case where our output measurements $\vec{y}$ are perturbed and we have some bounds on this perturbation, as well as some specific knowledge about data matrix A.

Let square matrix $A \in \mathbb{R}^{n \times n}$ have the singular value decomposition $A = U\Sigma V^\top$, and let its smallest singular value be $\sigma_{\min}\{A\} > 0$.

(a) Is A invertible? If so, write the singular value decomposition of A^{-1}.

(b) Consider the linear equation $A\vec{x} = \vec{y}_p$, where $\vec{y}_p \in \mathbb{R}^n$ is a perturbed measurement satisfying

$$\|\vec{y}_p - \vec{y}\|_2 \leq r \tag{4}$$

for some vector $\vec{y} \in \mathbb{R}^n$ and $r > 0$. Let $\vec{x}^\star(\vec{y})$ denote the solution of $A\vec{x} = \vec{y}$.

Show that

$$\max_{\vec{y}_p : \|\vec{y}_p - \vec{y}\|_2 \leq r} \|\vec{x}^\star(\vec{y}_p) - \vec{x}^\star(\vec{y})\|_2 = \frac{r}{\sigma_{\min}\{A\}} \tag{5}$$

(c) What happens if the smallest singular value of A is very close to zero? Why is this problematic for finding our solution vector $\vec{x}^\star$?

(d) Now assume that we find optimal value $\vec{x}^\star$ via ridge regression, i.e., we compute

$$\vec{x}^\star_\lambda(\vec{y}_p) = \operatorname*{argmin}_{\vec{x}} \left\{ \|A\vec{x} - \vec{y}_p\|_2^2 + \lambda \|\vec{x}\|_2^2 \right\} \tag{6}$$

for some chosen value $\lambda \geq 0$. Compute $\vec{x}^\star_\lambda(\vec{y}_p)$, our optimal solution vector (now parameterized by λ), by solving this optimization problem. You may use the solution from class for this part.

(e) Show that for all $\lambda > 0$,

$$\max_{\vec{y}_p : \|\vec{y}_p - \vec{y}\|_2 \leq r} \|\vec{x}^\star_\lambda(\vec{y}_p) - \vec{x}^\star_\lambda(\vec{y})\|_2 \leq \frac{r}{2\sqrt{\lambda}}. \tag{7}$$

How does the value of λ affect the sensitivity of your solution $\vec{x}^\star_\lambda(\vec{y})$ to the perturbation level in $\vec{y}$? *HINT: For every $\lambda > 0$, we have*

$$\max_{\sigma > 0} \frac{\sigma}{\sigma^2 + \lambda} = \frac{1}{2\sqrt{\lambda}}. \tag{8}$$

You need not show this; this optimization can be solved by setting the derivative of the objective function to 0 and solving for σ.

3. Linear Regression with Weights

In this problem, we discuss multiple interpretations of weighted linear regression.

Let $A \in \mathbb{R}^{m\times n}$ be a data matrix whose data points are the m rows $\vec{a}_1^\top, \ldots, \vec{a}_m^\top \in \mathbb{R}^n$. Suppose $m \geq n$ and A has full column rank. Let $\vec{y} \in \mathbb{R}^m$ be a vector of outputs, each corresponding to a data point. Let $\vec{w} \in \mathbb{R}^m_{++}$ be a vector of *positive* real numbers, also called weights, each corresponding to a data point—output pair. We are interested in the following least-squares type optimization problem:

$$\min_{\vec{x}\in\mathbb{R}^n} \sum_{i=1}^{m} w_i(\vec{a}_i^\top \vec{x} - y_i)^2. \tag{9}$$

In general, assigning a high weight w_i means that we want our learned linear predictor $\vec{a}_i^\top \vec{x}$ to achieve a close value to y_i; that is, we believe this data point is significant or important to get right.

(a) Show that the problem in Equation (9) is equivalent to the problem:

$$\min_{\vec{x}\in\mathbb{R}^n} \left\|W^{1/2}(A\vec{x} - \vec{y})\right\|_2^2 \tag{10}$$

where $W \doteq \operatorname{diag}(\vec{w}) \in \mathbb{R}^{m\times m}$ is a diagonal matrix whose diagonal entries are the entries of $\vec{w}$.

(b) Using Equation (10), compute the gradient (with respect to $\vec{x}$) of the objective function

$$f(\vec{x}) \doteq \left\|W^{1/2}(A\vec{x} - \vec{y})\right\|_2^2. \tag{11}$$

(c) Show that the optimal solution to Equation (10) is given by

$$\vec{x}^\star_{\text{WLR}} \doteq (A^\top W A)^{-1} A^\top W \vec{y}. \tag{12}$$

HINT: There are multiple ways to do this problem; one uses the gradient that you just computed, and another finds the least squares solution of a particular linear system.

HINT: If using the gradient method, you may assume that f is minimized at any $\vec{x}^\star$ such that $\nabla_{\vec{x}} f(\vec{x}^\star) = \vec{0}$; this is because f is convex, as we will see a little later in the course.

(d) Now we will look at this problem from a probabilistic interpretation. Suppose our output value $\vec{y}$ is noisy, and in particular there is some $\vec{x}_0$ such that for every i we have $y_i = \vec{a}_i^\top \vec{x}_0 + u_i$, where u_i is a random variable. Here we assume the u_i are independent but *not* identically distributed. In particular, we assume that for each i we have that u_i is distributed according to a Gaussian $\mathcal{N}(0, \sigma_i^2)$ where $\sigma_i > 0$ is a known noise parameter for each data point i.

To recover $\vec{x}_0$ given data A and $\vec{y}$, as well as the σ_i^2, we want to compute the maximum likelihood estimator (MLE). Show that the maximum likelihood problem

$$\operatorname*{argmax}_{\vec{x}\in\mathbb{R}^n} p(\vec{y} \mid A, \vec{x}) \tag{13}$$

is equivalent to the weighted linear regression problem:

$$\operatorname*{argmin}_{\vec{x}\in\mathbb{R}^n} \sum_{i=1}^{m} w_i(\vec{a}_i^\top \vec{x} - y_i)^2. \tag{14}$$

for some choice of $\vec{w}$. What choice of $\vec{w}$ makes them equivalent?

Note: Refer to Sec.4.5 of the course reader for a discussion on MLE.

(e) In addition to assigning weights to data points to place higher importance on getting those points right, sometimes we want to make sure that our learned $\vec{x}$ is close to some value $\vec{z} \in \mathbb{R}^n$. This could be true, for example, if we had prior information that said $\vec{x}$ is close to $\vec{z}$.

In particular, we want to make sure that the quantity

$$\sum_{i=1}^{n} s_i(x_i - z_i)^2 \tag{15}$$

is small, where $\vec{s} \in \mathbb{R}^n_{++}$ is a vector of *positive* weights. We may add this term to the weighted least squares objective function, with a regularization parameter $\lambda \geq 0$, to create a modified objective function

$$g(\vec{x}) \doteq \left\| W^{1/2}(A\vec{x} - \vec{y}) \right\|_2^2 + \lambda \left\| S^{1/2}(\vec{x} - \vec{z}) \right\|_2^2 \tag{16}$$

where $S \doteq \operatorname{diag}(\vec{s}) \in \mathbb{R}^{n \times n}$ is a diagonal matrix whose entries are the entries of $\vec{s}$. This formulation is called *Tikhonov regression*.

Compute the gradient (with respect to $\vec{x}$) of g, and show that the optimal solution is

$$\vec{x}^\star_{\mathrm{TR}} \doteq (A^\top W A + \lambda S)^{-1}(A^\top W \vec{y} + \lambda S \vec{z}). \tag{17}$$

HINT: You may assume that g is minimized at any $\vec{x}^\star$ such that $\nabla_{\vec{x}} g(\vec{x}^\star) = \vec{0}$; this is because g is convex, as we will see a little later in the course.

4. Quadratics and Least Squares

In this question, we will see that every least squares problem can be considered as minimization of a quadratic cost function; whereas not every quadratic minimization problem corresponds to a least-squares problem. To begin with, consider the quadratic function, $f : \mathbb{R}^2 \to \mathbb{R}$ given by:

$$f(\vec{w}) = \vec{w}^\top A\vec{w} - 2\vec{b}^\top \vec{w} + c \tag{18}$$

where $A \in \mathbb{S}^2_+$ (set of symmetric positive semidefinite matrices in $\mathbb{R}^{2\times 2}$), $\vec{b} \in \mathbb{R}^2$ and $c \in \mathbb{R}$.

(a) Assume $c = 0$, and assume that setting $\nabla f(\vec{w}) = 0$ allows us to find the unique minimizer. Give a concrete example of a matrix $A \succ 0$ and a vector $\vec{b}$ such that the point $\vec{w}^\star = \begin{bmatrix} -1 & 1 \end{bmatrix}^\top$ is the unique minimizer of the quadratic function $f(\vec{w})$.

(b) Assume $c = 0$. Give a concrete example of a matrix $A \succeq 0$, and a vector $\vec{b}$ such that the quadratic function $f(\vec{w})$ has infinitely many minimizers and all of them lie on the line $w_1 + w_2 = 0$.

HINT: Take the gradient of the expression and set it to zero. What needs to be true for there to be infinitely many solutions to the equation?

(c) Assume $c = 0$. Let $\vec{w} = \begin{bmatrix} 1 & 0 \end{bmatrix}^\top$. Give a concrete example of a **non-zero** matrix $A \succeq 0$ and a vector $\vec{b}$ such that the quadratic function $f(\alpha\vec{w})$ tends to $-\infty$ as $\alpha \to \infty$. *HINT: Use the eigenvalue decomposition to write $A = \sigma_1 \vec{u}_1\vec{u}_1^\top + \sigma_2\vec{u}_2\vec{u}_2^\top$ and express $\vec{w}$ in the basis formed by $\vec{u}_1, \vec{u}_2$.*

(d) Say that we have the data set $\{(\vec{x}_i, y_i)\}_{i=1,\ldots,n}$ of data points $\vec{x}_i \in \mathbb{R}^d$ and values $y_i \in \mathbb{R}$. Define $X = \begin{bmatrix} \vec{x}_1 & \ldots & \vec{x}_n \end{bmatrix}^\top$ and $\vec{y} = \begin{bmatrix} y_1 & \ldots & y_n \end{bmatrix}^\top$. In terms of X and $\vec{y}$, find a matrix A, a vector $\vec{b} \in \mathbb{R}^d$ and a scalar c, so that we can express the sum of the square losses $\sum_{i=1}^{n}(\vec{w}^\top\vec{x}_i - y_i)^2$ as the quadratic function $f(\vec{w}) = \vec{w}^\top A\vec{w} - 2\vec{b}^\top\vec{w} + c$.

(e) Here are three statements with regards to the minimization of a quadratic loss function:

i. It can have a unique minimizer.

ii. It can have infinitely many minimizers.

iii. It can be unbounded from below, i.e. there is some direction, $\vec{w}$ so that $f(\alpha\vec{w})$ goes to $-\infty$ as $\alpha \to \infty$.

All three statements apply to general minimization of a quadratic cost function. Parts (a), (b) and (c) give concrete examples of quadratic cost functions where (i), (ii) and (iii) apply respectively. However, notice that statement (iii) cannot apply to the least squares problem as the objective is always positive. The least-squares problem can have infinitely many minimizers though. How? Consider the gradient of the least squares problem in part (d) at an optimal solution $\vec{w}^\star$:

$$\nabla f(\vec{w}^\star) = 2X^\top X\vec{w}^\star - 2\vec{b} = 0. \tag{19}$$

Therefore, the least squares problem only has multiple solutions if $X^\top X$ is not full rank. This means that $\operatorname{rank}(X^\top X) = \operatorname{rank}(X) < d$. Finally, the rank of X is less than d when the data points $\{\vec{x}_i\}_{i=1}^n$ do not span $\mathbb{R}^d$. This can happen when the number of data points n is less than d or when $\{\vec{f}_i\}_{i=1}^d$ are linearly dependent where $\vec{f}_i$ are the columns of X, i.e., the features.

We will see soon that these cases correspond to the *convexity* of the function: if the function is strictly convex, then it has a unique minimizer; and if it is just convex, then it can have multiple minimizers; and in both cases, it can have no minimizers. We will see soon how to prove that the quadratic objective functions we discuss in this problem are convex, strictly convex, or even non-convex.

Indicate below that you have read and understood the discussion above.

5. Neural Networks and Backpropagation

Neural networks are parametric functions that have been widely used to fit complex patterns in vision and natural languages. Given some training data of the form $(\vec{x}_i, y_i)$, a neural network $\mathcal{N}$ is trained to minimize a *loss function* on the data. This is often done using *gradient descent*, an optimization method we will cover later in this class. Gradient descent requires us to compute the gradients of the loss function with respect to the parameters of the neural network. In practice, computational frameworks for neural networks compute the gradients automatically and efficiently via *back-propagation*, which uses the chain rule to recursively compute the gradients of the loss function. In this problem, we study a toy neural network trained on a single data point $(\vec{x}, y)$.

In particular, consider the following simplified three-layer neural network $\mathcal{N}$, representing a map from $\mathbb{R}^d$ to $\mathbb{R}$ whose parameters are $(\vec{w}_1, w_2, w_3) \in \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}$:

$$p_1 = \vec{w}_1^\top \vec{x} \tag{20}$$
$$h_1 = \sigma(p_1) \tag{21}$$
$$p_2 = w_2 h_1 \tag{22}$$
$$h_2 = \sigma(p_2) \tag{23}$$
$$z = w_3 h_2, \tag{24}$$

where $\sigma\colon \mathbb{R} \to \mathbb{R}$ is a nonlinear function (also called the "activation function"), whose derivative is denoted by $\sigma'\colon \mathbb{R} \to \mathbb{R}$.

We want the output of the network $z = \mathcal{N}(\vec{x})$ to match the true label y. A natural choice of the loss function encouraging this behavior is the squared loss:

$$L(y, z) \doteq \frac{1}{2}(y - z)^2. \tag{25}$$

In the parts that follow, we will compute the derivative of L with respect to the parameters $\vec{w}_1, w_2, w_3$.

(a) Compute the following gradients and partial derivatives sequentially from left-to-right:

$$\frac{\partial L}{\partial z}, \quad \frac{\partial L}{\partial w_3}, \quad \frac{\partial L}{\partial h_2}, \quad \frac{\partial L}{\partial p_2}, \quad \frac{\partial L}{\partial w_2}, \quad \frac{\partial L}{\partial h_1}, \quad \frac{\partial L}{\partial p_1}, \quad \nabla_{\vec{w}_1} L, \quad \nabla_{\vec{x}} L. \tag{26}$$

Here $\nabla_{\vec{x}} L$ is the gradient whose entries are the derivatives of L with respect to the entries of $\vec{x}$, etc. We compute the first 4 derivatives for you.

$$\frac{\partial L}{\partial z} = z - y, \quad \frac{\partial L}{\partial w_3} = \frac{\partial L}{\partial z} h_2, \quad \frac{\partial L}{\partial h_2} = \frac{\partial L}{\partial z} w_3, \quad \frac{\partial L}{\partial p_2} = \frac{\partial L}{\partial h_2} \sigma'(p_2) \tag{27}$$

Note how $\frac{\partial L}{\partial w_3}$ can be calculated using $\frac{\partial L}{\partial z}$ and $\frac{\partial L}{\partial p_2}$ can be calculated using $\frac{\partial L}{\partial h_2}$. In numerical computation, the result of $\frac{\partial L}{\partial z}$ and $\frac{\partial L}{\partial h_2}$ can thus be reused. This technique of saving computations for calculating the derivatives of a neural network is called *back-propagation*.

Use the chain rule to calculate the 5 remaining derivatives $\frac{\partial L}{\partial w_2}$, $\frac{\partial L}{\partial h_1}$, $\frac{\partial L}{\partial p_1}$,$\nabla_{\vec{w}_1} L$, and $\nabla_{\vec{x}} L$.

(b) **(OPTIONAL)** Now suppose that Equation (24) is written as

$$z' = w_3 h_2 + h_1. \tag{28}$$

That is, we define a new neural network $\mathcal{N}'$ with parameters $(\vec{w}_1, w_2, w_3) \in \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}$ as follows.

$$p_1 = \vec{w}_1^\top \vec{x} \tag{29}$$

$$h_1 = \sigma(p_1) \tag{30}$$
$$p_2 = w_2 h_1 \tag{31}$$
$$h_2 = \sigma(p_2) \tag{32}$$
$$z' = w_3 h_2 + h_1. \tag{33}$$

This introduces a change in the network architecture, called the *skip connection*.

Again, compute the following gradients and partial derivatives with respect to the loss function $L(y, z') = \frac{1}{2}(y - z')^2$:

$$\frac{\partial L}{\partial z'}, \quad \frac{\partial L}{\partial w_3}, \quad \frac{\partial L}{\partial h_2}, \quad \frac{\partial L}{\partial p_2}, \quad \frac{\partial L}{\partial w_2}, \quad \frac{\partial L}{\partial h_1}, \quad \frac{\partial L}{\partial p_1}, \quad \nabla_{\vec{w}_1} L, \quad \nabla_{\vec{x}} L. \tag{34}$$

We compute the first 4 derivatives for you.

$$\frac{\partial L}{\partial z'} = z' - y, \quad \frac{\partial L}{\partial w_3} = \frac{\partial L}{\partial z'} h_2, \quad \frac{\partial L}{\partial h_2} = \frac{\partial L}{\partial z'} w_3, \quad \frac{\partial L}{\partial p_2} = \frac{\partial L}{\partial h_2} \sigma'(p_2). \tag{35}$$

Use the chain rule to calculate the 5 remaining derivatives $\frac{\partial L}{\partial w_2}$, $\frac{\partial L}{\partial h_1}$, $\frac{\partial L}{\partial p_1}$,$\nabla_{\vec{w}_1} L$, and $\nabla_{\vec{x}} L$.

(c) **(OPTIONAL)** In optimizing a neural network using gradient descent, we need the gradient of the loss function with respect to the parameters of the network. Please express $\frac{\partial L}{\partial w_3}$, $\frac{\partial L}{\partial w_2}$, and $\nabla_{\vec{w}_1} L$ for $\mathcal{N}$ and $\mathcal{N}'$ respectively with no dependence on partial derivatives of other variables. We compute $\frac{\partial L}{\partial w_3}$ for you, as follows.

- For $\mathcal{N}$, we have $\frac{\partial L}{\partial w_3} = \frac{\partial L}{\partial z} h_2 = (z - y) h_2$.
- For $\mathcal{N}'$, we have $\frac{\partial L}{\partial w_3} = \frac{\partial L}{\partial z'} h_2 = (z' - y) h_2$.

Express the remaining derivatives $\frac{\partial L}{\partial w_2}$ and $\nabla_{\vec{w}_1} L$ within $\mathcal{N}$ and $\mathcal{N}'$.

(d) **(OPTIONAL)** Many activation functions σ have the property that $\sigma' \leq 1$. For example, the sigmoid function $\sigma(p) = \frac{1}{1+e^{-p}}$ is sometimes used as an activation function. Its derivative, $\sigma'(p) = \frac{e^{-p}}{(1+e^{-p})^2}$ has the range $(0, 1/4]$. That is, $\sigma' < 1$. Consider the case when σ' is much smaller than 1, such that $\sigma'(p)\sigma'(q) \approx 0$ for any p, q, but $\sigma'(p) \not\approx 0$ for any p. Consider the derivatives $\frac{\partial L}{\partial w_3}$, $\frac{\partial L}{\partial w_2}$, and $\nabla_{\vec{w}_1} L$ within the neural network $\mathcal{N}$; with the above approximations, which of them will approximately be zero? Also answer this question for the neural network $\mathcal{N}'$.

NOTE: Some of the above gradients will indeed be approximately zero, and this is called the *vanishing gradient* problem in deep learning.

6. **Homework Process**

   With whom did you work on this homework? List the names and SIDs of your group members.

   NOTE: If you didn't work with anyone, you can put "none" as your answer.