



hw0.pdf

PDF 227.09KB

Start by solving Question 2. For each part of question 2 explain show all the work needed to arrive at the answer.

### ⌚ Thought for 63 seconds ↴

We are asked to solve Question 2 from the homework. Question 2 is about vector calculus.

We have to show several derivative identities. The convention is that the derivative of a scalar with respect to a column vector is a row vector. So if  $x$  is a column vector, then  $\partial f / \partial x$  is a row vector.

Given:

$$x, c \in \mathbb{R}^n, A \in \mathbb{R}^{n \times n}.$$

We need to compute the following:

(a)  $\partial / \partial x (x^T c) = c^T$

(b)  $\partial / \partial x \|x\|_2^2 = 2 x^T$

(c)  $\partial / \partial x (A x) = A$

(d)  $\partial / \partial x (x^T A x) = x^T (A + A^T)$

(e) Under what condition is the previous derivative equal to  $2 x^T A$ ?

Let's go step by step.

Recall:  $x$  is a column vector, so  $x^T$  is a row vector. The derivative of a scalar function  $f(x)$  with respect to  $x$  is a row vector:  $[\partial f / \partial x_1, \partial f / \partial x_2, \dots, \partial f / \partial x_n]$ .

We'll compute each part.

Part (a): Show  $\partial / \partial x (x^T c) = c^T$ .

$x^T c$  is a scalar:  $\sum_{i=1}^n x_i c_i$ .

So,  $\partial / \partial x_j (x^T c) = c_j$ .

Thus, the derivative row vector is  $[c_1, c_2, \dots, c_n] = c^T$ .

So indeed,  $\partial / \partial x (x^T c) = c^T$ .

Part (b): Show  $\partial / \partial x \|x\|_2^2 = 2 x^T$ .

$\|x\|_2^2 = x^T x = \sum_{i=1}^n x_i^2$ .

$\partial / \partial x_j (x^T x) = 2 x_j$ .

So the derivative is  $[2 x_1, 2 x_2, \dots, 2 x_n] = 2 x^T$ .

Part (c): Show  $\partial / \partial x (A x) = A$ .

Wait, careful: Here  $f(x) = Ax$ , which is a vector. So the derivative of a vector function with respect to a vector is a matrix. Specifically, if  $f: \mathbb{R}^n \rightarrow \mathbb{R}^n$ ,  $f(x) = Ax$ , then the derivative  $\frac{\partial f}{\partial x}$  is an  $n \times n$  matrix. The convention: The derivative matrix has entries  $(\frac{\partial f_i}{\partial x_j})$ . Since  $f_i = \sum_k A_{ik} x_k$ , so  $\frac{\partial f_i}{\partial x_j} = A_{ij}$ . So the derivative matrix is exactly  $A$ . But note: The problem statement says "identify what the derivative looks like (is it a scalar, vector, or matrix?)". For part (c),  $f(x) = Ax$  is a vector, so its derivative is a matrix. And indeed, we have  $\frac{\partial}{\partial x}(Ax) = A$ .

But wait: In the problem it is written as  $\frac{\partial}{\partial x}(Ax)$ . Possibly they mean the derivative of the vector-valued function. Yes, so it's a matrix.

Part (d): Show  $\frac{\partial}{\partial x}(x^T Ax) = x^T(A + A^T)$ .

Here  $f(x) = x^T Ax$  is a scalar. So derivative is a row vector.

Compute:  $f(x) = \sum_{i,j} x_i A_{ij} x_j$ .

$$\frac{\partial f}{\partial x_k} = \frac{\partial}{\partial x_k} (\sum_{i,j} x_i A_{ij} x_j) = \sum_{i,j} A_{ij} \frac{\partial}{\partial x_k} (x_i x_j).$$

$$\text{We have } \frac{\partial}{\partial x_k} (x_i x_j) = \delta_{ik} x_j + x_i \delta_{jk}.$$

$$\text{So } \frac{\partial f}{\partial x_k} = \sum_{i,j} A_{ij} (\delta_{ik} x_j + x_i \delta_{jk}) = \sum_j A_{kj} x_j + \sum_i A_{ik} x_i.$$

That is:  $(Ax)_k + (x^T A)_k$ ? Actually careful:  $\sum_j A_{kj} x_j = (Ax)_k$ . And  $\sum_i A_{ik} x_i = (A^T x)_k$ ? Because  $(A^T)_{ki} = A_{ik}$ , so  $\sum_i A_{ik} x_i = \sum_i (A^T)_{ki} x_i = (A^T x)_k$ . So  $\frac{\partial f}{\partial x_k} = (Ax)_k + (A^T x)_k$ .

Thus, the derivative row vector is  $(Ax + A^T x)^T = x^T A^T + x^T A$ ? Wait:  $(Ax)^T = x^T A^T$ , and  $(A^T x)^T = x^T A$ . So the row vector is  $x^T A^T + x^T A = x^T (A^T + A)$ . So indeed,  $\frac{\partial}{\partial x}(x^T Ax) = x^T(A + A^T)$ .

Alternatively, we can compute using product rule. Since  $f$  is scalar, we can think of it as  $x^T(Ax)$ . Let  $g(x) = x$  and  $h(x) = Ax$ . Then  $f(x) = g(x)^T h(x)$ . Derivative:  $\frac{\partial(g^T h)}{\partial x} = (\frac{\partial g}{\partial x})^T h + g^T (\frac{\partial h}{\partial x})$ . But careful with conventions. Alternatively, we can use the formula:  $\frac{\partial(x^T Ax)}{\partial x} = x^T(A + A^T)$ . Yes.

Part (e): Under what condition is the previous derivative equal to  $2x^T A$ ?

We have  $\frac{\partial}{\partial x}(x^T Ax) = x^T(A + A^T)$ . This equals  $2x^T A$  if and only if  $x^T(A + A^T) = 2x^T A$ , i.e.,  $x^T A + x^T A^T = 2x^T A$ , which simplifies to  $x^T A^T = x^T A$ , i.e.,  $x^T(A^T - A) = 0$ . This holds for all  $x$  if and only if  $A^T = A$ , i.e.,  $A$  is symmetric. But the condition might be for a given  $x$ ? The question likely asks: For what condition on  $A$  does the equality hold for all  $x$ ? Probably  $A$  symmetric. But it says "under what condition", so likely  $A$  symmetric.

Alternatively, it might be that the equality holds for the specific  $x$ ? That wouldn't be interesting. So I'll say: When  $A$  is symmetric, i.e.,  $A = A^T$ .

Thus, the answer for (e) is: When  $A$  is symmetric.

Now, we need to present the solutions for (a) through (e) with explanation. The problem says: "before taking any derivatives, identify what the derivative looks like (is it a scalar, vector, or matrix?) and how we calculate each term in the derivative. Then carefully solve for an arbitrary entry of the derivative, then stack/arrange all of them to get the final result."

So for each part, we should:

1. Identify the type of the derivative.
2. Compute the derivative entry by entry.
3. Assemble the entries into the final form.

Let's write up the solutions.

Part (a):

- $f(x) = x^T c$  is a scalar.
- Derivative  $\partial f / \partial x$  is a row vector of size  $1 \times n$ .
- Compute  $\partial f / \partial x_j$ :  $f = \sum_{i=1}^n x_i c_i$ , so  $\partial f / \partial x_j = c_j$ .
- So the derivative row vector is  $[c_1, c_2, \dots, c_n] = c^T$ .

Part (b):

- $f(x) = \|x\|_2^2 = x^T x$  is a scalar.
- Derivative is a row vector.
- $\partial f / \partial x_j = \partial / \partial x_j (\sum_i x_i^2) = 2 x_j$ .
- So derivative is  $[2x_1, 2x_2, \dots, 2x_n] = 2 x^T$ .

Part (c):

- $f(x) = A x$  is a vector (size  $n \times 1$ ).
- Derivative  $\partial f / \partial x$  is a matrix of size  $n \times n$ .
- The  $(i,j)$  entry is  $\partial f_i / \partial x_j$ , where  $f_i = \sum_{k=1}^n A_{ik} x_k$ .
- So  $\partial f_i / \partial x_j = A_{ij}$ .
- Thus, the derivative matrix is  $A$ .

Part (d):

- $f(x) = x^T A x$  is a scalar.
- Derivative is a row vector.
- Compute  $\partial f / \partial x_k$ :  $f = \sum_{i,j} x_i A_{ij} x_j$ .
- $\partial f / \partial x_k = \sum_{i,j} A_{ij} (\partial (x_i x_j) / \partial x_k) = \sum_{i,j} A_{ij} (\delta_{ik} x_j + x_i \delta_{jk}) = \sum_j A_{kj} x_j + \sum_i A_{ik} x_i$ .
- The first sum is  $(A x)_k$ , the second sum is  $(A^T x)_k$ .
- So  $\partial f / \partial x_k = (A x)_k + (A^T x)_k$ .
- Therefore, the derivative row vector is  $(A x + A^T x)^T = x^T A^T + x^T A = x^T (A^T + A) = x^T (A + A^T)$ .

Part (e):

- We have  $\partial / \partial x (x^T A x) = x^T (A + A^T)$ .

- This equals  $2 \mathbf{x}^T \mathbf{A}$  if and only if  $\mathbf{x}^T (\mathbf{A} + \mathbf{A}^T) = 2 \mathbf{x}^T \mathbf{A} \Rightarrow \mathbf{x}^T \mathbf{A}^T = \mathbf{x}^T \mathbf{A} \Rightarrow \mathbf{x}^T (\mathbf{A}^T - \mathbf{A}) = 0$ .
- For this to hold for all  $\mathbf{x}$ , we need  $\mathbf{A}^T - \mathbf{A} = 0$ , i.e.,  $\mathbf{A}$  is symmetric.
- (If it only needs to hold for a particular  $\mathbf{x}$ , then the condition is that  $\mathbf{x}$  is in the left nullspace of  $(\mathbf{A}^T - \mathbf{A})$ . But typically we mean for all  $\mathbf{x}$ , so  $\mathbf{A}$  symmetric.)

Thus, the condition is that  $\mathbf{A}$  is symmetric.

Now, we can present the final answers.

## Question 2: Vector Calculus Review

Deepseek v3.2 correctly identifies the conventions required by the question.

### Convention:

The derivative of a scalar  $f$  with respect to a column vector  $\mathbf{x} \in \mathbb{R}^n$  is a row vector:

$$\frac{\partial f}{\partial \mathbf{x}} = \left[ \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_n} \right].$$

For a vector-valued function  $\mathbf{f}(\mathbf{x})$ , the derivative is a matrix with entries  $(\partial f_i / \partial x_j)$ .

**(a) Show**  $\frac{\partial}{\partial \mathbf{x}} (\mathbf{x}^T \mathbf{c}) = \mathbf{c}^T$ .

- $\mathbf{x}^T \mathbf{c} = \sum_{i=1}^n x_i c_i$  is a scalar.
- Derivative is a  $1 \times n$  row vector.
- For each  $j$ :

$$\frac{\partial}{\partial x_j} (\mathbf{x}^T \mathbf{c}) = \frac{\partial}{\partial x_j} \left( \sum_{i=1}^n x_i c_i \right) = c_j.$$

- Assembling into a row:  $[c_1, c_2, \dots, c_n] = \mathbf{c}^T$ .

$$\frac{\partial}{\partial \mathbf{x}} (\mathbf{x}^T \mathbf{c}) = \mathbf{c}^T$$

**(b) Show**  $\frac{\partial}{\partial \mathbf{x}} \|\mathbf{x}\|_2^2 = 2\mathbf{x}^T$ .

- $\|\mathbf{x}\|_2^2 = \mathbf{x}^T \mathbf{x} = \sum_{i=1}^n x_i^2$  is a scalar.
- Derivative is a  $1 \times n$  row vector.
- For each  $j$ :

$$\frac{\partial}{\partial x_j} \|\mathbf{x}\|_2^2 = \frac{\partial}{\partial x_j} \left( \sum_{i=1}^n x_i^2 \right) = 2x_j.$$

- Row vector:  $[2x_1, 2x_2, \dots, 2x_n] = 2\mathbf{x}^T$ .

$$\boxed{\frac{\partial}{\partial \mathbf{x}} \|\mathbf{x}\|_2^2 = 2\mathbf{x}^T}$$

**(c) Show**  $\frac{\partial}{\partial \mathbf{x}}(A\mathbf{x}) = A$ .

- $A\mathbf{x}$  is a vector in  $\mathbb{R}^n$  (assuming  $A \in \mathbb{R}^{n \times n}$ ).
- Derivative is an  $n \times n$  matrix.
- Let  $\mathbf{f}(\mathbf{x}) = A\mathbf{x}$ , so  $f_i = \sum_{k=1}^n A_{ik}x_k$ .
- The  $(i, j)$  entry is:

$$\frac{\partial f_i}{\partial x_j} = \frac{\partial}{\partial x_j} \left( \sum_{k=1}^n A_{ik}x_k \right) = A_{ij}.$$

- Thus, the derivative matrix is exactly  $A$ .

$$\boxed{\frac{\partial}{\partial \mathbf{x}}(A\mathbf{x}) = A}$$

**(d) Show**  $\frac{\partial}{\partial \mathbf{x}}(\mathbf{x}^T A\mathbf{x}) = \mathbf{x}^T(A + A^T)$ .

- $\mathbf{x}^T A\mathbf{x} = \sum_{i,j} x_i A_{ij} x_j$  is a scalar.
- Derivative is a  $1 \times n$  row vector.
- For each  $k$ :

Once again  
Deepseek does  
really well and  
adheres to proper  
proof conventions.

$$\frac{\partial}{\partial x_k}(\mathbf{x}^T A\mathbf{x}) = \frac{\partial}{\partial x_k} \left( \sum_{i,j} x_i A_{ij} x_j \right) = \sum_{i,j} A_{ij} \frac{\partial}{\partial x_k}(x_i x_j).$$

Using  $\frac{\partial}{\partial x_k}(x_i x_j) = \delta_{ik}x_j + x_i \delta_{jk}$ ,

$$= \sum_{i,j} A_{ij}(\delta_{ik}x_j + x_i \delta_{jk}) = \sum_j A_{kj}x_j + \sum_i A_{ik}x_i.$$

The first sum is  $(A\mathbf{x})_k$ , the second is  $(A^T\mathbf{x})_k$ .

- So  $\frac{\partial}{\partial x_k}(\mathbf{x}^T A \mathbf{x}) = (A\mathbf{x})_k + (A^T\mathbf{x})_k$ .
- As a row vector:

$$(A\mathbf{x} + A^T\mathbf{x})^T = \mathbf{x}^T A^T + \mathbf{x}^T A = \mathbf{x}^T(A^T + A) = \mathbf{x}^T(A + A^T).$$

$$\boxed{\frac{\partial}{\partial \mathbf{x}}(\mathbf{x}^T A \mathbf{x}) = \mathbf{x}^T(A + A^T)}$$

**(e) Under what condition is the previous derivative equal to  $2\mathbf{x}^T A$ ?**

- We have  $\frac{\partial}{\partial \mathbf{x}}(\mathbf{x}^T A \mathbf{x}) = \mathbf{x}^T(A + A^T)$ .
- Setting equal to  $2\mathbf{x}^T A$ :

$$\mathbf{x}^T(A + A^T) = 2\mathbf{x}^T A \iff \mathbf{x}^T A^T = \mathbf{x}^T A \iff \mathbf{x}^T(A^T - A) = 0.$$

- For this to hold for all  $\mathbf{x}$ , we require  $A^T - A = 0$ , i.e.,  $A$  is symmetric.

**When  $A = A^T$  (symmetric matrix).**

Similarly solve Question 3.

⌚ Thought for 145 seconds ▾

We are given Question 3: "Least Squares and the Min-norm problem from the Perspective of SVD". We need to answer each subpart.

We have the equation  $X\mathbf{w} = \mathbf{y}$ , with  $X \in \mathbb{R}^{m \times n}$ , and SVD:  $X = U\Sigma V^T$ , where  $U$  is  $m \times m$  orthogonal,  $V$  is  $n \times n$  orthogonal, and  $\Sigma$  is  $m \times n$  diagonal matrix with singular values  $\sigma_i$  on the diagonal (only for  $i = 1, \dots, r$  where  $r = \text{rank}(X)$ , and possibly zeros beyond that). We define  $\Sigma^\dagger$  as  $n \times m$  matrix with reciprocals  $1/\sigma_i$  on the diagonal for nonzero singular values, and zeros elsewhere.

We consider two cases: (a)  $m > n$  (overdetermined) and (d)  $m < n$  (underdetermined). We need to find the least squares solution and the minimum norm solution using SVD.

Let's go step by step.

**Part (a): For  $m > n$ , we want to solve  $\min_{\mathbf{w}} \|X\mathbf{w} - \mathbf{y}\|^2$ .**

The least squares solution is given by  $\mathbf{w}^* = (X^T X)^{-1} X^T \mathbf{y}$  if  $X^T X$  is invertible (which requires  $X$  to have full column rank, i.e., rank = n). In general, even if not full rank, the pseudo-inverse gives a solution. Using SVD, we have  $X = U \Sigma V^T$ . Then the pseudo-inverse is  $X^\dagger = V \Sigma^\dagger U^T$ . The least squares solution with minimum norm (if multiple) is  $\mathbf{w}^* = X^\dagger \mathbf{y}$ . But here, since  $m > n$ , the system is overdetermined, and if  $X$  has full rank, the unique least squares solution is  $\mathbf{w}^* = (X^T X)^{-1} X^T \mathbf{y} = X^\dagger \mathbf{y}$ . If not full rank, there are infinitely many solutions, but the one with minimum norm is  $X^\dagger \mathbf{y}$ . So we can write the least squares solution as  $\mathbf{w}^* = V \Sigma^\dagger U^T \mathbf{y}$ .

But the question asks: "How do we find the weights w that minimizes the error?" We need to derive it. Possibly, they want the expression using SVD.

**Part (b): Plug in the SVD  $X = U \Sigma V^T$  and simplify.**

We have  $\|X\mathbf{w} - \mathbf{y}\|^2 = \|U \Sigma V^T \mathbf{w} - \mathbf{y}\|^2$ . Since  $U$  is orthogonal, multiplication by  $U^T$  preserves norm:

$$\|U \Sigma V^T \mathbf{w} - \mathbf{y}\|^2 = \|U^T (U \Sigma V^T \mathbf{w} - \mathbf{y})\|^2 = \|\Sigma V^T \mathbf{w} - U^T \mathbf{y}\|^2.$$

Let  $\tilde{\mathbf{w}} = V^T \mathbf{w}$  (so  $\mathbf{w} = V \tilde{\mathbf{w}}$ ) and  $\tilde{\mathbf{y}} = U^T \mathbf{y}$ . Then the problem becomes:

$$\min_{\tilde{\mathbf{w}}} \|\Sigma \tilde{\mathbf{w}} - \tilde{\mathbf{y}}\|^2.$$

Now,  $\Sigma$  is  $m \times n$  diagonal matrix. Write  $\Sigma = \begin{bmatrix} D \\ 0 \end{bmatrix}$  if  $m > n$ , where  $D$  is  $n \times n$  diagonal with singular values. More precisely,  $\Sigma = \begin{bmatrix} \text{diag}(\sigma_1, \dots, \sigma_r) & 0 \\ 0 & 0 \end{bmatrix}$  in general, but if  $m > n$  and full rank, then  $\Sigma = \begin{bmatrix} D \\ 0 \end{bmatrix}$  with  $D$  invertible. But let's do it generally.

Let  $r = \text{rank}(X)$ . Then the singular values  $\sigma_1 \geq \dots \geq \sigma_r > 0$ , and  $\sigma_{r+1} = \dots = 0$ . The matrix  $\Sigma$  has the first  $r$  diagonal entries nonzero, rest zero. Similarly, partition  $\tilde{\mathbf{y}} = \begin{bmatrix} \tilde{\mathbf{y}}_1 \\ \tilde{\mathbf{y}}_2 \end{bmatrix}$  with  $\tilde{\mathbf{y}}_1 \in \mathbb{R}^r$  and  $\tilde{\mathbf{y}}_2 \in \mathbb{R}^{m-r}$ . Also,  $\tilde{\mathbf{w}} \in \mathbb{R}^n$ . Then

$$\Sigma \tilde{\mathbf{w}} = \begin{bmatrix} D_r & 0 \\ 0 & 0 \end{bmatrix} \tilde{\mathbf{w}} = \begin{bmatrix} D_r \tilde{\mathbf{w}}_1 \\ 0 \end{bmatrix},$$

where  $D_r$  is  $r \times r$  diagonal with  $\sigma_1, \dots, \sigma_r$ , and we have partitioned  $\tilde{\mathbf{w}} = \begin{bmatrix} \tilde{\mathbf{w}}_1 \\ \tilde{\mathbf{w}}_2 \end{bmatrix}$  with  $\tilde{\mathbf{w}}_1 \in \mathbb{R}^r$ . Actually careful:  $\Sigma$  is  $m \times n$ . Suppose we write  $\Sigma = \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix}$  where  $\Sigma_r$  is  $r \times r$  diagonal. But then  $\tilde{\mathbf{w}}$  is  $n$ -dimensional. We can write  $\tilde{\mathbf{w}} = \begin{bmatrix} \tilde{\mathbf{w}}_1 \\ \tilde{\mathbf{w}}_2 \end{bmatrix}$  with  $\tilde{\mathbf{w}}_1 \in \mathbb{R}^r, \tilde{\mathbf{w}}_2 \in \mathbb{R}^{n-r}$ .

Then

I noticed Deepseek's  
attention to the dimensions  
of sigma which is important  
to solution of this problem.

$$\Sigma \tilde{\mathbf{w}} = \begin{bmatrix} \Sigma_r \tilde{\mathbf{w}}_1 \\ 0 \\ 0 \end{bmatrix}$$

Actually, the multiplication:  $\Sigma \tilde{\mathbf{w}}$  gives an  $m$ -vector. The top  $r$  entries are  $\Sigma_r \tilde{\mathbf{w}}_1$ , and the remaining  $m - r$  entries are 0. So

$$\|\Sigma \tilde{\mathbf{w}} - \tilde{\mathbf{y}}\|^2 = \|\Sigma_r \tilde{\mathbf{w}}_1 - \tilde{\mathbf{y}}_1\|^2 + \|\tilde{\mathbf{y}}_2\|^2.$$

To minimize this, we choose  $\tilde{\mathbf{w}}_1$  such that  $\Sigma_r \tilde{\mathbf{w}}_1 = \tilde{\mathbf{y}}_1$ . Since  $\Sigma_r$  is diagonal and invertible, we get  $\tilde{\mathbf{w}}_1 = \Sigma_r^{-1} \tilde{\mathbf{y}}_1$ . For  $\tilde{\mathbf{w}}_2$ , it does not appear in the objective, so it can be arbitrary.

However, if we want the minimum norm solution among all least squares solutions, we set

$\tilde{\mathbf{w}}_2 = 0$ . So the minimum norm least squares solution is  $\tilde{\mathbf{w}} = \begin{bmatrix} \Sigma_r^{-1} \tilde{\mathbf{y}}_1 \\ 0 \end{bmatrix}$ . Then transforming back:  $\mathbf{w} = V \tilde{\mathbf{w}} = V \begin{bmatrix} \Sigma_r^{-1} \tilde{\mathbf{y}}_1 \\ 0 \end{bmatrix}$ . But note that  $\tilde{\mathbf{y}}_1$  consists of the first  $r$  components of  $U^T \mathbf{y}$ .

We can write  $\Sigma^\dagger$  as an  $n \times m$  matrix: is  $m \times n$ , so  $\Sigma^\dagger$  is  $n \times m$ . Typically, if  $\Sigma = \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix}$  with  $D$   $r \times r$ , then has diagonal entries  $\sigma_i$ , then  $\Sigma^\dagger$  has diagonal entries  $1/\sigma_i$  for nonzero  $\sigma_i$ , and zeros elsewhere, but arranged such that  $\Sigma^\dagger$  is  $n \times m$ . So if  $m > n$ , then  $\Sigma$  is tall, and  $\Sigma^\dagger$  is wide. In fact, if  $m > n$  and full rank (rank =  $n$ ), then  $\Sigma = \begin{bmatrix} D \\ 0 \end{bmatrix}$  where  $D$  is  $n \times n$  diagonal, then  $\Sigma^\dagger = [D^{-1} \quad 0]$  ( $n \times m$ ). So indeed,  $\mathbf{w} = V \Sigma^\dagger U^T \mathbf{y}$ .

So the answer for (a): The least squares solution is  $\mathbf{w}^* = (X^T X)^{-1} X^T \mathbf{y}$  if  $X$  has full column rank, or in general  $\mathbf{w}^* = X^\dagger \mathbf{y} = V \Sigma^\dagger U^T \mathbf{y}$ .

For (b): Plug in SVD and simplify as above, leading to  $\mathbf{w}^* = V \Sigma^\dagger U^T \mathbf{y}$ .

Part (c): We'll notice that the least-squares solution is in the form  $\mathbf{w}^* = A \mathbf{y}$ . What happens if we left-multiply  $X$  by our matrix  $A$ ? So  $A = X^\dagger$ . Then  $XA = XX^\dagger$ . For overdetermined case,  $XX^\dagger$  is not necessarily identity. Actually,  $XX^\dagger = U \Sigma V^T V \Sigma^\dagger U^T = U \Sigma \Sigma^\dagger U^T$ . Now  $\Sigma \Sigma^\dagger$  is an  $m \times m$  matrix. If  $m > n$  and rank =  $n$ , then  $\Sigma \Sigma^\dagger = \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix}$ . So  $XX^\dagger$  is the projection onto the column space of  $X$ . So  $XA$  acts as a left-inverse only when restricted to the column space? Actually, a left-inverse would satisfy  $AX = I$ . But here  $A$  is  $n \times m$ , and  $X$  is  $m \times n$ , so  $AX$  is  $n \times n$ . Indeed,  $AX = X^\dagger X = V \Sigma^\dagger U^T U \Sigma V^T = V \Sigma^\dagger \Sigma V^T$ . And  $\Sigma^\dagger \Sigma$  is  $n \times n$  diagonal with ones for the nonzero singular values. So if  $X$  has full column rank (rank =  $n$ ), then  $\Sigma^\dagger \Sigma = I_n$ , so  $AX = I_n$ . Thus  $A$  is a left-inverse of  $X$ . So that's why it's called the left-inverse: it satisfies  $X^\dagger X = I$ .

Part (d): Now consider  $m < n$ , underdetermined. We want the minimum norm solution:  $\min \|\mathbf{w}\|^2$  subject to  $X \mathbf{w} = \mathbf{y}$ . This is given by  $\mathbf{w}^* = X^T (X X^T)^{-1} \mathbf{y}$  if  $X$  has full row rank (rank =  $m$ ). In general,  $\mathbf{w}^* = X^\dagger \mathbf{y}$ , where  $X^\dagger = V \Sigma^\dagger U^T$ . But careful: For underdetermined case,  $X^\dagger = V \Sigma^\dagger U^T$  still, but now  $\Sigma$  is  $m \times n$  with  $m < n$ . Typically, if full row rank,  $\Sigma =$

$[D \ 0]$  where  $D$  is  $m \times m$  diagonal. Then  $\Sigma^\dagger = \begin{bmatrix} D^{-1} \\ 0 \end{bmatrix}$  ( $n \times m$ ). So  $\mathbf{w}^* = V \begin{bmatrix} D^{-1} \\ 0 \end{bmatrix} U^T \mathbf{y}$ . Alternatively, using the formula  $\mathbf{w}^* = X^T (X X^T)^{-1} \mathbf{y}$ .

Part (e): Plug in SVD:  $X = U \Sigma V^T$ . Constraint:  $U \Sigma V^T \mathbf{w} = \mathbf{y}$ . Multiply by  $U^T$ :  $\Sigma V^T \mathbf{w} = U^T \mathbf{y}$ . Let  $\tilde{\mathbf{w}} = V^T \mathbf{w}$ ,  $\tilde{\mathbf{y}} = U^T \mathbf{y}$ . Then  $\Sigma \tilde{\mathbf{w}} = \tilde{\mathbf{y}}$ . Now  $\Sigma$  is  $m \times n$ , wide. Write  $\Sigma = [D \ 0]$  if full row rank, where  $D$  is  $m \times m$  diagonal. Then the equation becomes:  $[D \ 0] \begin{bmatrix} \tilde{\mathbf{w}}_1 \\ \tilde{\mathbf{w}}_2 \end{bmatrix} = D \tilde{\mathbf{w}}_1 = \tilde{\mathbf{y}}$ . So  $\tilde{\mathbf{w}}_1 = D^{-1} \tilde{\mathbf{y}}$ , and  $\tilde{\mathbf{w}}_2$  is free. To minimize  $\|\mathbf{w}\|^2 = \|V \tilde{\mathbf{w}}\|^2 = \|\tilde{\mathbf{w}}\|^2$  (since  $V$  orthogonal), we set  $\tilde{\mathbf{w}}_2 = 0$ . So  $\tilde{\mathbf{w}} = \begin{bmatrix} D^{-1} \tilde{\mathbf{y}} \\ 0 \end{bmatrix}$ . Then  $\mathbf{w} = V \tilde{\mathbf{w}} = V \begin{bmatrix} D^{-1} \tilde{\mathbf{y}} \\ 0 \end{bmatrix}$ . This is of the form  $\mathbf{w} = V \Sigma^\dagger U^T \mathbf{y}$ .

Part (f): You'll notice that the min-norm solution is in the form  $\mathbf{w}^* = B \mathbf{y}$ . What happens if we right-multiply  $X$  by our matrix  $B$ ? Here  $B = X^\dagger = V \Sigma^\dagger U^T$ . Then  $BX = X^\dagger X = V \Sigma^\dagger U^T U \Sigma V^T = V \Sigma^\dagger \Sigma V^T$ . For underdetermined case with full row rank,  $\Sigma^\dagger \Sigma = \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix}$ , so not identity unless rank =  $n$  (which it isn't). Actually,  $XB = XX^\dagger = U \Sigma V^T V \Sigma^\dagger U^T = U \Sigma \Sigma^\dagger U^T$ . Now if full row rank,  $\Sigma \Sigma^\dagger = I_m$  (since  $\Sigma$  is  $m \times n$ , and  $\Sigma^\dagger$  is  $n \times m$ , and their product is  $m \times m$ :  $\Sigma \Sigma^\dagger = DD^{-1} = I_m$ ). So  $XB = I_m$ . Thus  $B$  is a right-inverse of  $X$ :  $XB = I$ .

Now, I need to write up the answers clearly.

Let me structure the answers.

### 3. Least Squares and the Min-norm problem from the Perspective of SVD

Given  $X = U \Sigma V^T$  (full SVD), with  $U \in \mathbb{R}^{m \times m}$ ,  $V \in \mathbb{R}^{n \times n}$  orthogonal, and  $\Sigma \in \mathbb{R}^{m \times n}$  diagonal with singular values  $\sigma_1, \dots, \sigma_r > 0$  ( $r = \text{rank}(X)$ ) on the main diagonal. Define  $\Sigma^\dagger \in \mathbb{R}^{n \times m}$  as the matrix with  $1/\sigma_i$  on the diagonal for  $i=1,\dots,r$ , and zeros elsewhere.

**(a)** For the overdetermined case ( $m > n$ ), the least squares solution minimizes  $\|X \mathbf{w} - \mathbf{y}\|^2$ . The solution is given by the pseudo-inverse:

$$\mathbf{w}^* = X^\dagger \mathbf{y} = (X^T X)^{-1} X^T \mathbf{y} \quad (\text{if } X \text{ has full column rank}).$$

In general, using SVD,  $\mathbf{w}^* = V \Sigma^\dagger U^T \mathbf{y}$ .

**(b)** Substituting  $X = U \Sigma V^T$ , the objective becomes:

$$\|X \mathbf{w} - \mathbf{y}\|^2 = \|U \Sigma V^T \mathbf{w} - \mathbf{y}\|^2 = \|\Sigma(V^T \mathbf{w}) - U^T \mathbf{y}\|^2.$$

Let  $\tilde{\mathbf{w}} = V^T \mathbf{w}$  and  $\tilde{\mathbf{y}} = U^T \mathbf{y}$ . Then we minimize  $\|\Sigma \tilde{\mathbf{w}} - \tilde{\mathbf{y}}\|^2$ . Write  $\Sigma = \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix}$  where  $\Sigma_r = \text{diag}(\sigma_1, \dots, \sigma_r)$ , and partition  $\tilde{\mathbf{w}} = \begin{bmatrix} \tilde{\mathbf{w}}_1 \\ \tilde{\mathbf{w}}_2 \end{bmatrix}$ ,  $\tilde{\mathbf{y}} = \begin{bmatrix} \tilde{\mathbf{y}}_1 \\ \tilde{\mathbf{y}}_2 \end{bmatrix}$ . Then

$$\|\Sigma \tilde{\mathbf{w}} - \tilde{\mathbf{y}}\|^2 = \|\Sigma_r \tilde{\mathbf{w}}_1 - \tilde{\mathbf{y}}_1\|^2 + \|\tilde{\mathbf{y}}_2\|^2.$$

This is minimized by choosing  $\tilde{\mathbf{w}}_1 = \Sigma_r^{-1} \tilde{\mathbf{y}}_1$ , and  $\tilde{\mathbf{w}}_2$  arbitrary. To obtain the minimum norm solution among all least squares solutions, we set  $\tilde{\mathbf{w}}_2 = 0$ . Thus,

$$\tilde{\mathbf{w}}^* = \begin{bmatrix} \Sigma_r^{-1} \tilde{\mathbf{y}}_1 \\ 0 \end{bmatrix} = \Sigma^\dagger \tilde{\mathbf{y}}.$$

Transforming back:  $\mathbf{w}^* = V \tilde{\mathbf{w}}^* = V \Sigma^\dagger U^T \mathbf{y}$ .

**(c)** The least squares solution is of the form  $\mathbf{w}^* = A\mathbf{y}$  with  $A = V\Sigma^\dagger U^T$ . Left-multiplying  $X$  by  $A$  gives:

$$AX = (V\Sigma^\dagger U^T)(U\Sigma V^T) = V\Sigma^\dagger \Sigma V^T.$$

For the overdetermined case with full column rank,  $\Sigma^\dagger \Sigma = I_n$ , so  $AX = I_n$ . Hence,  $A$  is a left-inverse of  $X$ .

**(d)** For the underdetermined case ( $m < n$ ), we seek the minimum norm solution:

$$\min \|\mathbf{w}\|^2 \quad \text{subject to} \quad X\mathbf{w} = \mathbf{y}.$$

This is given by

$$\mathbf{w}^* = X^\dagger \mathbf{y} = X^T (XX^T)^{-1} \mathbf{y} \quad (\text{if } X \text{ has full row rank}).$$

In general, using SVD,  $\mathbf{w}^* = V\Sigma^\dagger U^T \mathbf{y}$ .

**(e)** Substitute SVD:  $X = U\Sigma V^T$ . The constraint  $X\mathbf{w} = \mathbf{y}$  becomes  $U\Sigma V^T \mathbf{w} = \mathbf{y}$ , or  $\Sigma(V^T \mathbf{w}) = U^T \mathbf{y}$ . Let  $\tilde{\mathbf{w}} = V^T \mathbf{w}$ ,  $\tilde{\mathbf{y}} = U^T \mathbf{y}$ . Then we have  $\Sigma \tilde{\mathbf{w}} = \tilde{\mathbf{y}}$ . Write  $\Sigma = [\Sigma_r \ 0]$  if full row rank, where  $\Sigma_r$  is  $m \times m$  diagonal (assuming rank = m). Partition  $\tilde{\mathbf{w}} = \begin{bmatrix} \tilde{\mathbf{w}}_1 \\ \tilde{\mathbf{w}}_2 \end{bmatrix}$  with  $\tilde{\mathbf{w}}_1 \in \mathbb{R}^m$ . Then  $\Sigma \tilde{\mathbf{w}} = \Sigma_r \tilde{\mathbf{w}}_1 = \tilde{\mathbf{y}}$ , so  $\tilde{\mathbf{w}}_1 = \Sigma_r^{-1} \tilde{\mathbf{y}}$ . To minimize  $\|\tilde{\mathbf{w}}\|^2$ , set  $\tilde{\mathbf{w}}_2 = 0$ . Thus,

$$\tilde{\mathbf{w}}^* = \begin{bmatrix} \Sigma_r^{-1} \tilde{\mathbf{y}} \\ 0 \end{bmatrix} = \Sigma^\dagger \tilde{\mathbf{y}}.$$

Then  $\mathbf{w}^* = V \tilde{\mathbf{w}}^* = V \Sigma^\dagger U^T \mathbf{y}$ .

**(f)** The min-norm solution is of the form  $\mathbf{w}^* = B\mathbf{y}$  with  $B = V\Sigma^\dagger U^T$ . Right-multiplying  $X$  by  $B$  gives:

$$XB = (U\Sigma V^T)(V\Sigma^\dagger U^T) = U\Sigma \Sigma^\dagger U^T.$$

For the underdetermined case with full row rank,  $\Sigma \Sigma^\dagger = I_m$ , so  $XB = I_m$ . Hence,  $B$  is a right-inverse of  $X$ .

I'll now produce the final answer in the required format.

### Question 3: Least Squares and the Min-Norm Problem from the Perspective of SVD

Let  $X \in \mathbb{R}^{m \times n}$  with SVD  $X = U\Sigma V^T$ , where  $U \in \mathbb{R}^{m \times m}$  and  $V \in \mathbb{R}^{n \times n}$  are orthogonal, and  $\Sigma \in \mathbb{R}^{m \times n}$  is diagonal with singular values  $\sigma_1 \geq \dots \geq \sigma_r > 0$  ( $r = \text{rank}(X)$ ) on the main diagonal. Define  $\Sigma^\dagger \in \mathbb{R}^{n \times m}$  as the matrix with  $1/\sigma_i$  on the diagonal for  $i = 1, \dots, r$  and zeros elsewhere.

#### (a) Overdetermined case ( $m > n$ ): least squares solution

We solve  $\min_{\mathbf{w}} \|X\mathbf{w} - \mathbf{y}\|^2$ . The optimal weight vector is given by the pseudo-inverse:

$$\boxed{\mathbf{w}^* = X^\dagger \mathbf{y} = (X^T X)^{-1} X^T \mathbf{y} \quad (\text{if } X \text{ has full column rank})}$$

In general, using the SVD,  $\mathbf{w}^* = V\Sigma^\dagger U^T \mathbf{y}$ .

#### (b) Substitute SVD and simplify

$$\begin{aligned} \|X\mathbf{w} - \mathbf{y}\|^2 &= \|U\Sigma V^T \mathbf{w} - \mathbf{y}\|^2 \\ &= \|\Sigma(V^T \mathbf{w}) - U^T \mathbf{y}\|^2 \quad (\text{since } U \text{ is orthogonal}). \end{aligned}$$

Let  $\tilde{\mathbf{w}} = V^T \mathbf{w}$  and  $\tilde{\mathbf{y}} = U^T \mathbf{y}$ . Partition

$$\Sigma = \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix}, \quad \tilde{\mathbf{w}} = \begin{bmatrix} \tilde{\mathbf{w}}_1 \\ \tilde{\mathbf{w}}_2 \end{bmatrix}, \quad \tilde{\mathbf{y}} = \begin{bmatrix} \tilde{\mathbf{y}}_1 \\ \tilde{\mathbf{y}}_2 \end{bmatrix},$$

where  $\Sigma_r = \text{diag}(\sigma_1, \dots, \sigma_r)$  is  $r \times r$ . Then

$$\|\Sigma \tilde{\mathbf{w}} - \tilde{\mathbf{y}}\|^2 = \|\Sigma_r \tilde{\mathbf{w}}_1 - \tilde{\mathbf{y}}_1\|^2 + \|\tilde{\mathbf{y}}_2\|^2.$$

Minimization gives  $\tilde{\mathbf{w}}_1 = \Sigma_r^{-1} \tilde{\mathbf{y}}_1$ , and to obtain the minimum-norm solution we set  $\tilde{\mathbf{w}}_2 = \mathbf{0}$ . Hence

$$\tilde{\mathbf{w}}^* = \begin{bmatrix} \Sigma_r^{-1} \tilde{\mathbf{y}}_1 \\ \mathbf{0} \end{bmatrix} = \Sigma^\dagger \tilde{\mathbf{y}}, \quad \mathbf{w}^* = V \tilde{\mathbf{w}}^* = V \Sigma^\dagger U^T \mathbf{y}.$$

### (c) Left-inverse interpretation

Writing  $\mathbf{w}^* = A\mathbf{y}$  with  $A = V\Sigma^\dagger U^T$ , we compute

$$AX = (V\Sigma^\dagger U^T)(U\Sigma V^T) = V\Sigma^\dagger \Sigma V^T.$$

If  $X$  has full column rank ( $r = n$ ), then  $\Sigma^\dagger \Sigma = I_n$ , so  $AX = I_n$ . Thus  $A$  is a **left-inverse** of  $X$ .

### (d) Underdetermined case ( $m < n$ ): minimum-norm solution

We solve  $\min \|\mathbf{w}\|^2$  subject to  $X\mathbf{w} = \mathbf{y}$ . The solution is

$$\boxed{\mathbf{w}^* = X^\dagger \mathbf{y} = X^T (XX^T)^{-1} \mathbf{y} \quad (\text{if } X \text{ has full row rank})}$$

In general, using SVD,  $\mathbf{w}^* = V\Sigma^\dagger U^T \mathbf{y}$ .

### (e) Substitute SVD and simplify for the underdetermined case

From  $X\mathbf{w} = \mathbf{y}$  we obtain  $\Sigma(V^T \mathbf{w}) = U^T \mathbf{y}$ . Let  $\tilde{\mathbf{w}} = V^T \mathbf{w}$  and  $\tilde{\mathbf{y}} = U^T \mathbf{y}$ . With full row rank ( $r = m$ ), write

$$\Sigma = \begin{bmatrix} \Sigma_m & 0 \end{bmatrix}, \quad \tilde{\mathbf{w}} = \begin{bmatrix} \tilde{\mathbf{w}}_1 \\ \tilde{\mathbf{w}}_2 \end{bmatrix},$$

where  $\Sigma_m = \text{diag}(\sigma_1, \dots, \sigma_m)$  is  $m \times m$ . The constraint becomes  $\Sigma_m \tilde{\mathbf{w}}_1 = \tilde{\mathbf{y}}$ . Thus  $\tilde{\mathbf{w}}_1 = \Sigma_m^{-1} \tilde{\mathbf{y}}$ . To minimize  $\|\tilde{\mathbf{w}}\|^2$  we set  $\tilde{\mathbf{w}}_2 = \mathbf{0}$ , so

$$\tilde{\mathbf{w}}^* = \begin{bmatrix} \Sigma_m^{-1} \tilde{\mathbf{y}} \\ \mathbf{0} \end{bmatrix} = \Sigma^\dagger \tilde{\mathbf{y}}, \quad \mathbf{w}^* = V\tilde{\mathbf{w}}^* = V\Sigma^\dagger U^T \mathbf{y}.$$

Overall, Deepseek performs really well in these initial problems due to their straightforward nature.

### (f) Right-inverse interpretation

Writing  $\mathbf{w}^* = B\mathbf{y}$  with  $B = V\Sigma^\dagger U^T$ , we compute

$$XB = (U\Sigma V^T)(V\Sigma^\dagger U^T) = U\Sigma\Sigma^\dagger U^T.$$

If  $X$  has full row rank ( $r = m$ ), then  $\Sigma\Sigma^\dagger = I_m$ , so  $XB = I_m$ . Thus  $B$  is a **right-inverse** of  $X$ .

Similarly solve Question 4.

 Thought for 131 seconds ▾

We are to solve Question 4, which has multiple parts (a) through (h). I will go through each part carefully.

#### Question 4: The 5 Interpretations of Ridge Regression

##### (a) Perspective 1: Optimization Problem

We want to solve:

$$\underset{\mathbf{w}}{\operatorname{argmin}} \|\mathbf{y} - X\mathbf{w}\|_2^2 + \lambda \|\mathbf{w}\|_2^2,$$

where  $X \in \mathbb{R}^{n \times d}$ ,  $\mathbf{y} \in \mathbb{R}^n$ , and  $\lambda > 0$ .

Let the objective function be:

$$J(\mathbf{w}) = \|\mathbf{y} - X\mathbf{w}\|^2 + \lambda \|\mathbf{w}\|^2.$$

We can expand:

$$\|\mathbf{y} - X\mathbf{w}\|^2 = (\mathbf{y} - X\mathbf{w})^T(\mathbf{y} - X\mathbf{w}) = \mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T X\mathbf{w} + \mathbf{w}^T X^T X\mathbf{w}.$$

Also,  $\lambda \|\mathbf{w}\|^2 = \lambda \mathbf{w}^T \mathbf{w}$ .

Thus,

$$J(\mathbf{w}) = \mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T X\mathbf{w} + \mathbf{w}^T(X^T X + \lambda I)\mathbf{w}.$$

Take derivative with respect to  $\mathbf{w}$  (using the convention that the derivative of a scalar with respect to a column vector is a row vector, but we'll later set to zero as a column).

Alternatively, we can use the standard result that the gradient (as a column vector) is:

$$\nabla_{\mathbf{w}} J = -2X^T\mathbf{y} + 2(X^T X + \lambda I)\mathbf{w}.$$

Setting the gradient to zero:

$$-2X^T\mathbf{y} + 2(X^T X + \lambda I)\mathbf{w} = 0 \quad \Rightarrow \quad (X^T X + \lambda I)\mathbf{w} = X^T\mathbf{y}.$$

Assuming  $X^T X + \lambda I$  is invertible (which it is for  $\lambda > 0$  since it is positive definite), we get:

$$\boxed{\mathbf{w} = (X^T X + \lambda I)^{-1} X^T \mathbf{y}}.$$

### (b) Perspective 2: "Hack" of shifting the Singular Values

Given the SVD of  $X = U\Sigma V^T$ , where  $U$  is  $n \times n$  orthogonal,  $V$  is  $d \times d$  orthogonal, and  $\Sigma$  is  $n \times d$  diagonal with singular values  $\sigma_i$  (only non-zero for  $i = 1, \dots, r$ , with  $r = \text{rank}(X)$ ). The ridge solution becomes:

$$\mathbf{w} = (X^T X + \lambda I)^{-1} X^T \mathbf{y}.$$

Now,

$$X^T X = V\Sigma^T \Sigma V^T = V\Lambda V^T,$$

where  $\Lambda = \Sigma^T \Sigma$  is a  $d \times d$  diagonal matrix with entries  $\sigma_i^2$  (and possibly zeros). Then,

$$X^T X + \lambda I = V(\Lambda + \lambda I)V^T,$$

so

$$(X^T X + \lambda I)^{-1} = V(\Lambda + \lambda I)^{-1}V^T.$$

Also,

$$X^T \mathbf{y} = V\Sigma^T U^T \mathbf{y}.$$

Thus,

$$\mathbf{w} = V(\Lambda + \lambda I)^{-1}V^T \cdot V\Sigma^T U^T \mathbf{y} = V(\Lambda + \lambda I)^{-1}\Sigma^T U^T \mathbf{y}.$$

Note that  $(\Lambda + \lambda I)^{-1}\Sigma^T$  is an  $d \times n$  matrix. Specifically, consider the diagonal entries: For each singular value  $\sigma_i$ , the corresponding entry in  $(\Lambda + \lambda I)^{-1}\Sigma^T$  is  $\frac{\sigma_i}{\sigma_i^2 + \lambda}$  for  $i = 1, \dots, r$ , and zero for  $i > r$  (since  $\sigma_i = 0$ ). Thus, the effective singular values of the operator  $(X^T X + \lambda I)^{-1} X^T$  are  $\frac{\sigma_i}{\sigma_i^2 + \lambda}$ .

- When  $\sigma_i \ll \lambda$ , then  $\frac{\sigma_i}{\sigma_i^2 + \lambda} \approx \frac{\sigma_i}{\lambda} \rightarrow 0$ .
- When  $\sigma_i \gg \lambda$ , then  $\frac{\sigma_i}{\sigma_i^2 + \lambda} \approx \frac{\sigma_i}{\sigma_i^2} = \frac{1}{\sigma_i}$ , which is the same as in the ordinary least squares pseudo-inverse.

Thus, ridge regression shrinks the influence of small singular values (directions with low variance in the data) more heavily, while keeping the large singular value directions nearly unchanged.

### (c) Perspective 3: Maximum A Posteriori (MAP) estimation

We are given: Prior on  $\mathbf{W} \sim \mathcal{N}(\mathbf{0}, I)$  (so covariance is identity; note: the problem statement says  $\mathcal{N}(\mathbf{0}, I)$ , but we need to be careful about scaling). And the generative model:  $Y = \mathbf{x}^T \mathbf{W} + \sqrt{\lambda} N$ , where  $N \sim \mathcal{N}(0, 1)$  i.i.d. across training samples. At the vector level:  $\mathbf{Y} = X\mathbf{W} + \sqrt{\lambda}\mathbf{N}$ , where  $\mathbf{N} \sim \mathcal{N}(\mathbf{0}, I_n)$ .

We observe  $\mathbf{Y} = \mathbf{y}$ . We want the MAP estimate of  $\mathbf{W}$ . MAP maximizes the posterior  $p(\mathbf{W}|\mathbf{Y} = \mathbf{y}) \propto p(\mathbf{Y}|\mathbf{W})p(\mathbf{W})$ .

Given  $\mathbf{W}, \mathbf{Y}$  is Gaussian:  $\mathbf{Y} \sim \mathcal{N}(X\mathbf{W}, \lambda I_n)$  because  $\sqrt{\lambda}\mathbf{N}$  has covariance  $\lambda I_n$ .

Prior:  $\mathbf{W} \sim \mathcal{N}(\mathbf{0}, I_d)$ .

Thus,

$$\log p(\mathbf{W}) = -\frac{1}{2}\mathbf{W}^T \mathbf{W} + \text{const},$$

and

$$\log p(\mathbf{Y} = \mathbf{y}|\mathbf{W}) = -\frac{1}{2\lambda}\|\mathbf{y} - X\mathbf{W}\|^2 + \text{const.}$$

So the log-posterior (up to constants) is:

$$-\frac{1}{2\lambda}\|\mathbf{y} - X\mathbf{W}\|^2 - \frac{1}{2}\mathbf{W}^T \mathbf{W}.$$

Maximizing the log-posterior is equivalent to minimizing:

$$\frac{1}{\lambda}\|\mathbf{y} - X\mathbf{W}\|^2 + \mathbf{W}^T \mathbf{W}.$$

Multiplying by  $\lambda$  (which is positive, so doesn't change optimum):

$$\|\mathbf{y} - X\mathbf{W}\|^2 + \lambda \mathbf{W}^T \mathbf{W}.$$

This is exactly the ridge regression objective (1). Therefore, the MAP estimate is given by the ridge solution.

### (d) Perspective 4: Fake Data

We define augmented data:

$$\hat{\mathbf{y}} = \begin{bmatrix} \mathbf{y} \\ \mathbf{0}_d \end{bmatrix}, \quad \hat{X} = \begin{bmatrix} X \\ \sqrt{\lambda}I_d \end{bmatrix}.$$

The OLS problem on this augmented data is:

$$\min_{\mathbf{w}} \|\hat{\mathbf{y}} - \hat{X}\mathbf{w}\|^2 = \left\| \begin{bmatrix} \mathbf{y} \\ \mathbf{0}_d \end{bmatrix} - \begin{bmatrix} X \\ \sqrt{\lambda}I_d \end{bmatrix} \mathbf{w} \right\|^2 = \|\mathbf{y} - X\mathbf{w}\|^2 + \|\mathbf{0}_d - \sqrt{\lambda}\mathbf{w}\|^2.$$

But  $\|\mathbf{0}_d - \sqrt{\lambda}\mathbf{w}\|^2 = \lambda\|\mathbf{w}\|^2$ . Hence the objective is exactly  $\|\mathbf{y} - X\mathbf{w}\|^2 + \lambda\|\mathbf{w}\|^2$ , which is the ridge regression objective. So they have the same minimizer.

### (e) Perspective 5: Fake Features

We define augmented design matrix:

$$\tilde{X} = [X \quad \sqrt{\lambda}I_n] \in \mathbb{R}^{n \times (d+n)}.$$

We consider the underdetermined system  $\tilde{X}\boldsymbol{\eta} = \mathbf{y}$ , and seek the min-norm solution:

$$\min_{\boldsymbol{\eta}} \|\boldsymbol{\eta}\|^2 \quad \text{s.t.} \quad \tilde{X}\boldsymbol{\eta} = \mathbf{y}.$$

Let  $\boldsymbol{\eta} = \begin{bmatrix} \mathbf{w} \\ \boldsymbol{\nu} \end{bmatrix}$ , where  $\mathbf{w} \in \mathbb{R}^d$  and  $\boldsymbol{\nu} \in \mathbb{R}^n$ . Then the constraint becomes:

$$X\mathbf{w} + \sqrt{\lambda}\boldsymbol{\nu} = \mathbf{y}.$$

The objective is  $\|\boldsymbol{\eta}\|^2 = \|\mathbf{w}\|^2 + \|\boldsymbol{\nu}\|^2$ . We want to minimize over  $\mathbf{w}$  and  $\boldsymbol{\nu}$  subject to the constraint. We can solve for  $\boldsymbol{\nu}$  from the constraint:

$$\boldsymbol{\nu} = \frac{1}{\sqrt{\lambda}}(\mathbf{y} - X\mathbf{w}),$$

provided  $\lambda > 0$ . Then the objective becomes:

$$\|\mathbf{w}\|^2 + \left\| \frac{1}{\sqrt{\lambda}}(\mathbf{y} - X\mathbf{w}) \right\|^2 = \|\mathbf{w}\|^2 + \frac{1}{\lambda} \|\mathbf{y} - X\mathbf{w}\|^2.$$

Multiplying by  $\lambda$  (which does not change the minimizer in  $\mathbf{w}$ ), we get:

$$\lambda\|\mathbf{w}\|^2 + \|\mathbf{y} - X\mathbf{w}\|^2,$$

which is the ridge regression objective (with the order of terms swapped but same). Hence the minimizer  $\mathbf{w}^*$  from this problem is the ridge regression solution. And indeed, the first  $d$  coordinates of  $\boldsymbol{\eta}^*$  give  $\mathbf{w}^*$ .

### (f) Use pseudo-inverse to show equivalence

We have  $\tilde{X} = [X \quad \sqrt{\lambda}I_n]$ . The min-norm solution to  $\tilde{X}\boldsymbol{\eta} = \mathbf{y}$  is  $\boldsymbol{\eta}^* = \tilde{X}^\dagger \mathbf{y}$ . For a wide matrix (underdetermined), the Moore-Penrose pseudo-inverse is given by  $\tilde{X}^\dagger =$

$\tilde{X}^T(\tilde{X}\tilde{X}^T)^{-1}$ . First compute:

$$\tilde{X}\tilde{X}^T = [X \quad \sqrt{\lambda}I_n] \begin{bmatrix} X^T \\ \sqrt{\lambda}I_n \end{bmatrix} = XX^T + \lambda I_n.$$

Thus,

$$\tilde{X}^\dagger = \tilde{X}^T(XX^T + \lambda I_n)^{-1} = \begin{bmatrix} X^T \\ \sqrt{\lambda}I_n \end{bmatrix} (XX^T + \lambda I_n)^{-1}.$$

Then,

$$\boldsymbol{\eta}^* = \tilde{X}^\dagger \mathbf{y} = \begin{bmatrix} X^T(XX^T + \lambda I_n)^{-1} \mathbf{y} \\ \sqrt{\lambda}(XX^T + \lambda I_n)^{-1} \mathbf{y} \end{bmatrix}.$$

The first  $d$  coordinates give:

$$\hat{\mathbf{w}} = X^T(XX^T + \lambda I_n)^{-1} \mathbf{y}.$$

This is the so-called "kernel ridge regression" form (or dual form). Now we need to show this is equivalent to the standard ridge regression formula:

$$\hat{\mathbf{w}} = (X^T X + \lambda I_d)^{-1} X^T \mathbf{y}.$$

We use the identity (Woodbury matrix identity, or more simply, note that both expressions are solutions to the same problem, so they must be equal if the inverses exist). Alternatively, we can verify by manipulating:

Starting from  $\hat{\mathbf{w}} = X^T(XX^T + \lambda I_n)^{-1} \mathbf{y}$ , multiply both sides by  $(X^T X + \lambda I_d)$ :

$$(X^T X + \lambda I_d) X^T (XX^T + \lambda I_n)^{-1} \mathbf{y} = X^T (XX^T + \lambda I_n) (XX^T + \lambda I_n)^{-1} \mathbf{y} = X^T \mathbf{y},$$

where we used the fact that  $(X^T X + \lambda I_d) X^T = X^T (XX^T + \lambda I_n)$ . Indeed,

$$(X^T X + \lambda I_d) X^T = X^T X X^T + \lambda X^T = X^T (XX^T + \lambda I_n).$$

Thus,  $\hat{\mathbf{w}}$  satisfies  $(X^T X + \lambda I_d) \hat{\mathbf{w}} = X^T \mathbf{y}$ , so it must be equal to  $(X^T X + \lambda I_d)^{-1} X^T \mathbf{y}$  (provided the inverse exists). This shows the equivalence.

### (g) Limit as $\lambda \rightarrow \infty$

As  $\lambda \rightarrow \infty$ , the ridge solution  $\hat{\mathbf{w}}_r = (X^T X + \lambda I)^{-1} X^T \mathbf{y}$  behaves like:

$$\hat{\mathbf{w}}_r \approx \frac{1}{\lambda} X^T \mathbf{y} \rightarrow \mathbf{0}.$$

More precisely,  $(X^T X + \lambda I)^{-1} \approx \frac{1}{\lambda} I$  for large  $\lambda$ , so  $\hat{\mathbf{w}}_r \approx \frac{1}{\lambda} X^T \mathbf{y} \rightarrow \mathbf{0}$ . Thus, the weights shrink to zero. That's why ridge regression is called "shrinkage".

### (h) Limit as $\lambda \rightarrow 0$

Consider two cases:

- **Tall matrix (overdetermined,  $n > d$ ):** As  $\lambda \rightarrow 0$ , ridge regression approaches ordinary least squares. If  $X$  has full column rank, then  $(X^T X)^{-1}$  exists and  $\hat{\mathbf{w}} \rightarrow (X^T X)^{-1} X^T \mathbf{y}$ . If  $X$  does not have full column rank, then the limit may not be unique, but ridge regression with  $\lambda \rightarrow 0$  converges to the minimum-norm least squares solution (since it adds a tiny regularization, it picks the solution with smallest norm).
- **Wide matrix (underdetermined,  $n < d$ ):** As  $\lambda \rightarrow 0$ , the ridge solution approaches the minimum-norm solution of  $X\mathbf{w} = \mathbf{y}$  (if consistent) or the minimum-norm least squares solution. More precisely, the limit is  $X^\dagger \mathbf{y}$ , which is the minimum-norm solution among all least squares solutions. For  $\lambda > 0$ , the solution is unique; as  $\lambda \rightarrow 0$ , it converges to the pseudo-inverse solution.

In both cases, as  $\lambda \rightarrow 0$ , ridge regression recovers the pseudo-inverse solution (minimum-norm least squares solution).

I have provided detailed solutions for all parts of Question 4. Now, if needed, I can also solve Questions 5, 6, and 7, but the instruction was to solve Question 4 similarly to how I did Questions 2 and 3. So I'll stop here unless further questions are asked.

## Question 4: The 5 Interpretations of Ridge Regression

### (a) Perspective 1: Optimization Problem

We minimize

$$J(\mathbf{w}) = \|\mathbf{y} - X\mathbf{w}\|^2 + \lambda \|\mathbf{w}\|^2.$$

Expanding and taking the gradient with respect to  $\mathbf{w}$ :

$$\nabla_{\mathbf{w}} J = -2X^T \mathbf{y} + 2(X^T X + \lambda I)\mathbf{w}.$$

Setting the gradient to zero:

$$(X^T X + \lambda I)\mathbf{w} = X^T \mathbf{y}.$$

Thus, the solution is

$$\boxed{\mathbf{w} = (X^T X + \lambda I)^{-1} X^T \mathbf{y}}.$$

## (b) Perspective 2: "Hack" of Shifting Singular Values

Let  $X = U\Sigma V^T$  be the SVD. Then

$$\mathbf{w} = V(\Lambda + \lambda I)^{-1}\Sigma^T U^T \mathbf{y},$$

where  $\Lambda = \Sigma^T \Sigma$ . The effective singular values are  $\frac{\sigma_i}{\sigma_i^2 + \lambda}$ .

- When  $\sigma_i \ll \lambda$ :  $\frac{\sigma_i}{\sigma_i^2 + \lambda} \approx \frac{\sigma_i}{\lambda} \rightarrow 0$ .
- When  $\sigma_i \gg \lambda$ :  $\frac{\sigma_i}{\sigma_i^2 + \lambda} \approx \frac{1}{\sigma_i}$ , as in ordinary least squares.

While Deepseek arrives at the correct solutions, it has begun to minimize the amount of the rigorous proofs it provides. However, this may show more understanding as it does not require a robotic thinking process to solve these problems and finds shortcuts in the explanations.

## (c) Perspective 3: Maximum A Posteriori (MAP) Estimation

Prior:  $\mathbf{W} \sim \mathcal{N}(\mathbf{0}, I)$ . Likelihood:  $\mathbf{Y} \mid \mathbf{W} \sim \mathcal{N}(X\mathbf{W}, \lambda I)$ .

The log-posterior is proportional to

$$-\frac{1}{2\lambda} \|\mathbf{y} - X\mathbf{W}\|^2 - \frac{1}{2} \|\mathbf{W}\|^2.$$

Maximizing this is equivalent to minimizing  $\|\mathbf{y} - X\mathbf{W}\|^2 + \lambda \|\mathbf{W}\|^2$ , which is the ridge regression objective.

## (d) Perspective 4: Fake Data

Define augmented data:

$$\hat{\mathbf{y}} = \begin{bmatrix} \mathbf{y} \\ \mathbf{0}_d \end{bmatrix}, \quad \hat{X} = \begin{bmatrix} X \\ \sqrt{\lambda} I_d \end{bmatrix}.$$

Then

$$\|\hat{\mathbf{y}} - \hat{X}\mathbf{w}\|^2 = \|\mathbf{y} - X\mathbf{w}\|^2 + \lambda \|\mathbf{w}\|^2,$$

so the minimizer is the same as ridge regression.

### (e) Perspective 5: Fake Features

Define  $\tilde{X} = [X \ \sqrt{\lambda}I_n]$ . Let  $\boldsymbol{\eta} = [\mathbf{w}; \boldsymbol{\nu}]$ . The constraint  $\tilde{X}\boldsymbol{\eta} = \mathbf{y}$  gives

$$\boldsymbol{\nu} = \frac{1}{\sqrt{\lambda}}(\mathbf{y} - X\mathbf{w}).$$

The min-norm problem minimizes  $\|\mathbf{w}\|^2 + \|\boldsymbol{\nu}\|^2$ , which becomes

$$\|\mathbf{w}\|^2 + \frac{1}{\lambda}\|\mathbf{y} - X\mathbf{w}\|^2.$$

Multiplying by  $\lambda$  gives the ridge objective, so the first  $d$  coordinates of  $\boldsymbol{\eta}^*$  give the ridge solution.

### (f) Pseudo-inverse and Equivalence

For  $\tilde{X} = [X \ \sqrt{\lambda}I_n]$ , the pseudo-inverse is

$$\tilde{X}^\dagger = \tilde{X}^T(\tilde{X}\tilde{X}^T)^{-1} = \begin{bmatrix} X^T \\ \sqrt{\lambda}I_n \end{bmatrix} (XX^T + \lambda I_n)^{-1}.$$

Thus,

$$\boldsymbol{\eta}^* = \begin{bmatrix} X^T(XX^T + \lambda I_n)^{-1}\mathbf{y} \\ \sqrt{\lambda}(XX^T + \lambda I_n)^{-1}\mathbf{y} \end{bmatrix}.$$

The first block gives  $\hat{\mathbf{w}} = X^T(XX^T + \lambda I_n)^{-1}\mathbf{y}$ . Using the identity

$$(X^T X + \lambda I_d)X^T = X^T(XX^T + \lambda I_n),$$

we multiply both sides of the expression for  $\hat{\mathbf{w}}$  by  $(X^T X + \lambda I_d)$  to get

$$(X^T X + \lambda I_d)\hat{\mathbf{w}} = X^T\mathbf{y},$$

so  $\hat{\mathbf{w}} = (X^T X + \lambda I_d)^{-1}X^T\mathbf{y}$ .

### (g) Limit as $\lambda \rightarrow \infty$

As  $\lambda \rightarrow \infty$ ,

$$\mathbf{w} \approx \frac{1}{\lambda} X^T \mathbf{y} \rightarrow \mathbf{0}.$$

Thus, ridge regression shrinks weights to zero.

### (h) Limit as $\lambda \rightarrow 0$

- **Tall matrix (overdetermined):**  $\mathbf{w} \rightarrow (X^T X)^{-1} X^T \mathbf{y}$  if  $X$  has full column rank; otherwise, it converges to the minimum-norm least squares solution.
- **Wide matrix (underdetermined):**  $\mathbf{w} \rightarrow X^\dagger \mathbf{y}$ , the minimum-norm solution to  $X\mathbf{w} = \mathbf{y}$  (if consistent) or the minimum-norm least squares solution.

In both cases, as  $\lambda \rightarrow 0$ , ridge regression recovers the pseudo-inverse solution.

Similarly solve Question 5.

### ⌚ Thought for 109 seconds ~

We are solving Question 5: ReLU Elbow Update under SGD.

Part (a) deals with a single ReLU:  $\phi(x) = \max(0, wx+b)$ . The "elbow" is the point where  $wx+b = 0$ , i.e.,  $x = -b/w$  (provided  $w \neq 0$ ). We assume  $w \neq 0$  for the elbow to exist.

Loss:  $\ell(x,y,\phi) = 1/2 (\phi(x)-y)^2$ .

(i) Elbow location e:

Since  $\phi(x) = wx+b$  for  $wx+b > 0$ , and 0 otherwise, the elbow is at  $wx+b = 0 \Rightarrow x = -b/w$ .

(ii) Derivative of loss w.r.t.  $\phi(x)$ :

$$d\ell/d\phi = \phi(x) - y.$$

(iii) Partial derivative of loss w.r.t. w:

Note that  $\phi(x)$  depends on w only when  $wx+b > 0$ . Using chain rule:  $\partial\ell/\partial w = (\phi(x)-y) * \partial\phi/\partial w$ . But  $\partial\phi/\partial w = x$  if  $wx+b > 0$ , and 0 if  $wx+b < 0$ . At  $wx+b=0$ , we take gradient 0 as given.

So:

$$\partial\ell/\partial w = (\phi(x)-y) * x * I\{wx+b>0\}.$$

(iv) Partial derivative of loss w.r.t. b:

Similarly,  $\partial\phi/\partial b = 1$  if  $wx+b > 0$ , else 0. So:

$$\partial\ell/\partial b = (\phi(x)-y) * I\{wx+b>0\}.$$

Now part (b): Given  $\phi(x)-y = 1$  (prediction too high by 1). We consider cases and describe what happens to slope and elbow after a gradient descent update on w and b. The update:  $w' = w - \lambda \partial\ell/\partial w$ ,  $b' = b - \lambda \partial\ell/\partial b$ .

We need to analyze each case.

Case (i):  $\phi(x)=0$ . But if  $\phi(x)=0$ , then  $\phi(x)-y = -y = 1 \Rightarrow y = -1$ . But  $\phi(x)=0$  means either  $wx+b < 0$  or  $wx+b=0$ . Since  $\phi(x)-y=1$ , we have  $0-y=1$  so  $y=-1$ . For gradient, we need to check if  $wx+b>0$ ? No, because  $\phi(x)=0$ . So by definition, if  $\phi(x)=0$ , we assume  $wx+b \leq 0$ . With our convention gradient at 0 is 0. So  $\partial\ell/\partial w = 0$ ,  $\partial\ell/\partial b = 0$ . Therefore, no update:  $w'=w$ ,  $b'=b$ . So slope and elbow unchanged.

Case (ii):  $w>0$ ,  $x>0$ , and  $\phi(x)>0$ . Then  $wx+b > 0$ . Given  $\phi(x)-y=1$ , so prediction is 1 above target. Then  $\partial\ell/\partial w = (\phi(x)-y)*x = 1*x = x > 0$ ,  $\partial\ell/\partial b = (\phi(x)-y)*1 = 1$ . So update:  $w' = w - \lambda x$ ,  $b' = b - \lambda$ . Since  $\lambda>0$ ,  $w$  decreases,  $b$  decreases. The slope  $w$  becomes smaller. The elbow  $e = -b/w$ . Let's compute change in  $e$ . Initially  $e = -b/w$ . After update,  $e' = -b'/w' = -(b-\lambda)/(w-\lambda x)$ . We need to check if  $e' > e$  or  $< e$ . Since  $w>0$ ,  $x>0$ , and both  $w$  and  $b$  decrease, we can analyze numerically. For example, take  $w=2$ ,  $b=1$ ,  $x=1$ ,  $\lambda=0.1$ . Then initially  $e = -1/2 = -0.5$ . After update:  $w'=2-0.1=1.9$ ,  $b'=1-0.1=0.9$ , so  $e' = -0.9/1.9 \approx -0.4737$ . So  $e' > e$  (less negative), meaning the elbow moves to the right. In general, for  $w>0$ ,  $x>0$ , the elbow moves to the right? Let's derive the change:  $e' - e = (-b'/w') - (-b/w) = -b'/w' + b/w$ . We can compute the derivative of  $e$  w.r.t. parameters, but the problem says it's fine to check numerically. So answer: slope decreases, elbow moves to the right.

Case (iii):  $w>0$ ,  $x<0$ , and  $\phi(x)>0$ . Then  $wx+b>0$ . Since  $w>0$ ,  $x<0$ , note that  $wx$  is negative. For  $\phi(x)>0$ , we need  $b$  sufficiently large. Given  $\phi(x)-y=1$ . Compute gradients:  $\partial\ell/\partial w = (\phi(x)-y)*x = 1*x = x < 0$ ,  $\partial\ell/\partial b = 1$ . So  $w' = w - \lambda x = w - \lambda$  (negative)  $= w + \lambda|x|$ , so  $w$  increases.  $b' = b - \lambda$ , so  $b$  decreases. Slope increases. Elbow  $e = -b/w$ . Initially  $e = -b/w$ . After update,  $w' > w$ ,  $b' < b$ . So numerator of  $e'$  becomes  $-b' > -b$  (since  $b' < b$ , so  $-b' > -b$ ), denominator  $w' > w$ . Both numerator and denominator increase. The change in  $e$  depends. For example, take  $w=2$ ,  $b=3$ ,  $x=-1$ ,  $\lambda=0.1$ . Then initially  $e = -3/2 = -1.5$ . Update:  $w' = 2 - 0.1*(-1)=2+0.1=2.1$ ,  $b'=3-0.1=2.9$ ,  $e' = -2.9/2.1 \approx -1.381$ . So  $e' > e$  (less negative), elbow moves to the right. Another example:  $w=0.5$ ,  $b=1$ ,  $x=-0.5$ ,  $\lambda=0.1$ . Then  $e = -1/0.5 = -2$ . Update:  $w'=0.5 - 0.1*(-0.5)=0.5+0.05=0.55$ ,  $b'=1-0.1=0.9$ ,  $e' = -0.9/0.55 \approx -1.636$ , still  $> -2$ . So elbow moves right. Generally, it seems elbow moves right. But we can analyze:  $e = -b/w$ ,  $de/dw = b/w^2$ ,  $de/db = -1/w$ . The total change  $\Delta e \approx (\partial e/\partial w)\Delta w + (\partial e/\partial b)\Delta b$ . Here  $\Delta w = -\lambda \partial\ell/\partial w = -\lambda x$ ,  $\Delta b = -\lambda \partial\ell/\partial b = -\lambda$ . So  $\Delta e \approx (b/w^2)*(-\lambda x) + (-1/w)*(-\lambda) = -\lambda b x / w^2 + \lambda/w = \lambda/w (1 - b x / w)$ . Since  $\phi(x)=wx+b > 0$ , we have  $b > -wx$ . But  $x<0$ , so  $-wx>0$ . So  $b$  is positive enough. The sign of  $\Delta e$  depends on  $1 - b x / w$ . Since  $x<0$ ,  $b x / w$  is negative, so  $1 - \text{negative} > 1$ , so  $\Delta e > 0$ . So  $e$  increases, meaning elbow moves right.

Case (iv):  $w<0$ ,  $x>0$ , and  $\phi(x)>0$ . Then  $wx+b>0$ . Since  $w<0$ ,  $x>0$ ,  $wx$  is negative. For  $\phi(x)>0$ ,  $b$  must be positive and large enough. Given  $\phi(x)-y=1$ . Gradients:  $\partial\ell/\partial w = (\phi(x)-y)*x = x > 0$ ,  $\partial\ell/\partial b = 1$ . So update:  $w' = w - \lambda x$ . Since  $x>0$ ,  $w' = w - \lambda x$  becomes more negative (since subtracting positive from negative). So magnitude of  $w$  increases (more negative).  $b' = b - \lambda$ , decreases. Slope  $w$  is negative, so slope becomes more negative (steeper downward slope). Elbow  $e = -b/w$ . Since  $w<0$ ,  $-b/w = b/|w|$ . Initially  $e = b/|w|$ . After update:  $w'$  more negative, so  $|w'| > |w|$ ;  $b' < b$ . So numerator decreases, denominator increases, so  $e' < e$ . Elbow

moves left. Numerical:  $w=-2$ ,  $b=3$ ,  $x=1$ ,  $\lambda=0.1$ . Then  $e = -3/(-2)=1.5$ . Update:  $w'=-2-0.1=-2.1$ ,  $b'=3-0.1=2.9$ ,  $e' = -2.9/(-2.1)= 2.9/2.1 \approx 1.381$ , so  $e' < e$ , moves left. So answer: slope becomes more negative, elbow moves left.

We also need to draw and label, but since this is text, we'll describe: For each case, sketch  $\phi(x)$  as a piecewise linear function with elbow at  $e$ . Show the point  $(x,y)$  such that  $\phi(x)-y=1$  (so  $\phi(x)=y+1$ ). After update, the slope changes and elbow shifts as described.

Now part (c): Full network:  $\hat{f}(x) = W^{(2)} \Phi(W^{(1)} x + b)$ . Here  $W^{(1)}$  is  $d \times 1$ , so it's a column vector (call it  $w^{(1)}$ ),  $b$  is  $d$ -dimensional vector,  $W^{(2)}$  is  $1 \times d$  row vector (call it  $w^{(2)T}$ ). Actually careful:  $W^{(1)} \in \mathbb{R}^{d \times 1}$ , so it's essentially a vector; let's denote it as a column vector  $w_1$  (each entry  $w_{1,i}$ ). Similarly  $b$  is vector  $b_i$ . Then the input to the  $i$ -th ReLU is  $z_i = w_{1,i} * x + b_i$ . The elbow for the  $i$ -th ReLU is where  $z_i = 0$ , i.e.,  $x$  such that  $w_{1,i} x + b_i = 0 \Rightarrow x = -b_i / w_{1,i}$ , provided  $w_{1,i} \neq 0$ . So  $e_i = -b_i / w_{1,i}$ .

Part (d): After one SGD update with learning rate  $\lambda$ , we need the new elbow location  $e'_i$ . The update depends on the loss gradient w.r.t. parameters. The loss is  $\ell = 1/2 (f(x)-y)^2$ . We need gradients w.r.t.  $w_{1,i}$ ,  $b_i$ , and  $w_{2,i}$ . But the elbow  $e_i$  depends only on  $w_{1,i}$  and  $b_i$ . So we need to compute how  $w_{1,i}$  and  $b_i$  change.

Let's derive gradients for the network. Let  $z = W^{(1)} x + b$ , a vector. Then  $a = \Phi(z)$  (elementwise ReLU). Then  $f = W^{(2)} a = \sum_{i=1}^d w_{2,i} a_i$ .

Loss  $\ell = 1/2 (f - y)^2$ .

Gradient w.r.t.  $W^{(2)}$ :  $\partial \ell / \partial w_{2,i} = (f-y) a_i$ .

Gradient w.r.t.  $b_i$ :  $\partial \ell / \partial b_i = (f-y) w_{2,i} * I\{z_i > 0\}$  (since  $\partial a_i / \partial b_i = 1$  if  $z_i > 0$  else 0).

Gradient w.r.t.  $w_{1,i}$ :  $\partial \ell / \partial w_{1,i} = (f-y) w_{2,i} * x * I\{z_i > 0\}$ .

Now, after an SGD step on parameters (assuming we update all parameters using the same point  $(x,y)$ ), we have:

$$w'_{1,i} = w_{1,i} - \lambda * \partial \ell / \partial w_{1,i} = w_{1,i} - \lambda (f-y) w_{2,i} x * I\{z_i > 0\}$$

$$b'_i = b_i - \lambda * \partial \ell / \partial b_i = b_i - \lambda (f-y) w_{2,i} I\{z_i > 0\}$$

Then the new elbow for neuron  $i$  is  $e'_i = -b'_i / w'_{1,i}$ , provided  $w'_{1,i} \neq 0$ .

If the neuron was not active ( $z_i \leq 0$ ), then  $I\{z_i > 0\}=0$ , so no update to  $w_{1,i}$  and  $b_i$  from this data point, so  $e_i$  unchanged.

If the neuron was active, then the update occurs. Note that the change depends on  $(f-y)$ ,  $w_{2,i}$ ,  $x$ , and  $\lambda$ .

So the new elbow location is:

$$\begin{aligned} e'_i &= - (b_i - \lambda (f-y) w_{2,i} I\{z_i > 0\}) / (w_{1,i} - \lambda (f-y) w_{2,i} x * I\{z_i > 0\}) \\ &= -b'_i / w'_{1,i} \end{aligned}$$

with the understanding that if  $w1_i'$  becomes zero, the elbow is at infinity or undefined.

That is the answer.

We should also note that if  $w1_i$  was zero, then originally there is no elbow (the ReLU is either always 0 or always linear depending on  $b_i$ ). But if  $w1_i=0$ , then the gradient for  $w1_i$  is  $\partial\ell/\partial w1_i = (f-y) w2_i * x * \mathbb{I}\{b_i>0\}$  (since  $z_i = b_i$ ). So it might become nonzero after update. But the problem likely assumes  $w1_i \neq 0$ .

So final answers:

(a)

- (i)  $e = -b/w$ .
- (ii)  $d\ell/d\phi = \phi(x)-y$ .
- (iii)  $\partial\ell/\partial w = (\phi(x)-y) * x * \mathbb{I}\{wx+b>0\}$ .
- (iv)  $\partial\ell/\partial b = (\phi(x)-y) * \mathbb{I}\{wx+b>0\}$ .

(b)

- (i) No change.
- (ii) Slope decreases, elbow moves right.
- (iii) Slope increases, elbow moves right.
- (iv) Slope becomes more negative, elbow moves left.

(c)  $e_i = -b_i / w1_i$ .

(d)  $e_i' = - (b_i - \lambda (f-y) w2_i \mathbb{I}\{z_i>0\}) / (w1_i - \lambda (f-y) w2_i x \mathbb{I}\{z_i>0\})$ .

Now, we need to put it in a box. Since it's long, we'll box final answers per part.

Let's present clearly.

## Question 5: ReLU Elbow Update under SGD

### (a) Single ReLU Analysis

For  $\phi(x) = \max(0, wx + b)$  and loss  $\ell(x, y, \phi) = \frac{1}{2}(\phi(x) - y)^2$ :

#### (i) Elbow location:

$$e = -\frac{b}{w} \quad (w \neq 0)$$

#### (ii) Derivative of loss w.r.t. $\phi(x)$ :

$$\frac{d\ell}{d\phi} = \phi(x) - y$$

(iii) Partial derivative w.r.t.  $w$ :

$$\frac{\partial \ell}{\partial w} = (\phi(x) - y) \cdot x \cdot \mathbb{I}_{\{wx+b>0\}}$$

(iv) Partial derivative w.r.t.  $b$ :

$$\frac{\partial \ell}{\partial b} = (\phi(x) - y) \cdot \mathbb{I}_{\{wx+b>0\}}$$

(b) Gradient Descent Updates Given  $\phi(x) - y = 1$

Update rule:  $p' = p - \lambda \frac{\partial \ell}{\partial p}$  for  $p = w, b$ .

(i)  $\phi(x) = 0$ :

No gradient ( $\mathbb{I}_{\{wx+b>0\}} = 0$ ), so slope and elbow remain unchanged.

(ii)  $w > 0, x > 0, \phi(x) > 0$ :

- $\frac{\partial \ell}{\partial w} = x > 0, \frac{\partial \ell}{\partial b} = 1$
- $w$  decreases,  $b$  decreases
- Slope decreases, elbow moves **right** (e.g.,  $w = 2, b = 1, x = 1, \lambda = 0.1$ :  $e$  from  $-0.5$  to  $\approx -0.474$ )

(iii)  $w > 0, x < 0, \phi(x) > 0$ :

- $\frac{\partial \ell}{\partial w} = x < 0, \frac{\partial \ell}{\partial b} = 1$
- $w$  increases,  $b$  decreases
- Slope increases, elbow moves **right** (e.g.,  $w = 2, b = 3, x = -1, \lambda = 0.1$ :  $e$  from  $-1.5$  to  $\approx -1.381$ )

(iv)  $w < 0, x > 0, \phi(x) > 0$ :

- $\frac{\partial \ell}{\partial w} = x > 0, \frac{\partial \ell}{\partial b} = 1$
- $w$  becomes more negative,  $b$  decreases
- Slope becomes more negative, elbow moves **left** (e.g.,  $w = -2, b = 3, x = 1, \lambda = 0.1$ :  $e$  from  $1.5$  to  $\approx 1.381$ )

Deepseek finally begins to make errors in the solutions. The model relies on a specific numerical example to determine the direction of the elbow movement.

This causes deepseek to ignore other possibilites per each part.

### (c) Elbow Location in the Full Network

For network  $\hat{f}(x) = \mathbf{W}^{(2)}\Phi(\mathbf{W}^{(1)}x + \mathbf{b})$ , let  $\mathbf{z} = \mathbf{W}^{(1)}x + \mathbf{b}$ . The elbow for the  $i$ -th ReLU is:

$$e_i = -\frac{b_i}{W_i^{(1)}} \quad (W_i^{(1)} \neq 0)$$

### (d) New Elbow After One SGD Update

Let  $z_i = W_i^{(1)}x + b_i$  and  $a_i = \Phi(z_i)$ . The loss gradient yields updates:

$$\begin{aligned} W_i^{(1)\prime} &= W_i^{(1)} - \lambda(\hat{f} - y)W_i^{(2)}x\mathbb{I}_{\{z_i>0\}} \\ b'_i &= b_i - \lambda(\hat{f} - y)W_i^{(2)}\mathbb{I}_{\{z_i>0\}} \end{aligned}$$

The new elbow location is:

$$e'_i = -\frac{b'_i}{W_i^{(1)\prime}}$$

I flag deepseek for its solution for part D since this is a lazy answer and does not fully solve for the new elbow location.

If  $z_i \leq 0$ , no update occurs and  $e'_i = e_i$ .

Expand on the solution for 5.d. This solution is not correct

⊗ Thought for 51 seconds ▾

We are asked to derive the new elbow location after one SGD update for the  $i$ -th ReLU in the full network.

We have:

$\hat{f}(x) = \mathbf{W}^{(2)}\Phi(\mathbf{W}^{(1)}x + \mathbf{b}) = \mathbf{W}^{(2)}a$ , where  $a = \Phi(z)$  and  $z = \mathbf{W}^{(1)}x + \mathbf{b}$ . Here,  $\mathbf{W}^{(1)}$  is  $d \times 1$ ,  $\mathbf{b}$  is  $d \times 1$ ,  $\mathbf{W}^{(2)}$  is  $1 \times d$ .

We consider the loss:  $L = 1/2 (\hat{f}(x) - y)^2$ .

We want to update parameters  $\mathbf{W}^{(1)}_i$ ,  $b_i$  for the  $i$ -th neuron. The gradient of  $L$  w.r.t. these parameters:

First, note:  $\hat{f}(x) = \sum_j \mathbf{W}^{(2)}_j a_j$ , where  $a_j = \Phi(z_j) = \max(0, z_j)$ .

Then:

$$\frac{\partial L}{\partial W^{(1)}_i} = \frac{\partial L}{\partial \hat{f}} \cdot \frac{\partial \hat{f}}{\partial a_i} \cdot \frac{\partial a_i}{\partial z_i} \cdot \frac{\partial z_i}{\partial W^{(1)}_i} = (\hat{f} - y) \cdot W^{(2)}_i \cdot \mathbb{I}(z_i > 0) \cdot x.$$

Similarly:

$$\frac{\partial L}{\partial b_i} = (\hat{f} - y) \cdot W^{(2)}_i \cdot \mathbb{I}(z_i > 0).$$

The update rule (SGD with learning rate  $\lambda$ ):

$$W^{(1)}_i' = W^{(1)}_i - \lambda \frac{\partial L}{\partial W^{(1)}_i},$$
$$b_i' = b_i - \lambda \frac{\partial L}{\partial b_i}.$$

Now, the elbow location for the  $i$ -th ReLU is defined as the point where the input to the ReLU becomes zero:  $z_i = W^{(1)}_i x + b_i = 0$ , so  $x = -b_i / W^{(1)}_i$ , provided  $W^{(1)}_i \neq 0$ . That is  $e_i$ .

After one update, the new parameters are  $W^{(1)}_i'$  and  $b_i'$ , so the new elbow location  $e_i'$  is given by solving  $W^{(1)}_i' x + b_i' = 0$ :

$$e_i' = -b_i' / W^{(1)}_i', \text{ provided } W^{(1)}_i' \neq 0.$$

We need to express  $e_i'$  in terms of the old parameters and the update. Let's compute explicitly:

Let  $\delta = \hat{f} - y$ .

Let  $I_i = \mathbb{I}(z_i > 0)$ .

Then:

$$W^{(1)}_i' = W^{(1)}_i - \lambda \delta W^{(2)}_i I_i x,$$

$$b_i' = b_i - \lambda \delta W^{(2)}_i I_i.$$

Note: The indicator  $I_i$  depends on whether  $z_i > 0$ . Importantly, after the update, the elbow may shift, and the indicator may change if the update causes the activation pattern to change. However, the problem likely assumes that the learning rate is small enough that the activation pattern does not change in a single update, or we are to compute the new elbow location based solely on the updated parameters regardless of whether the activation pattern changes. Usually, in such analysis, we consider the update as a small step and ignore the change in the indicator.

Now, compute  $e_i'$ :

$$e_i' = -b_i' / W^{(1)}_i' = - (b_i - \lambda \delta W^{(2)}_i I_i) / (W^{(1)}_i - \lambda \delta W^{(2)}_i I_i x).$$

We can rewrite this in terms of the old elbow  $e_i = -b_i / W^{(1)}_i$ .

We have:

$$b_i = -e_i W^{(1)}_i.$$

Plug into numerator and denominator:

$$\text{Numerator: } b_i - \lambda \delta W^{(2)}_i I_i = -e_i W^{(1)}_i - \lambda \delta W^{(2)}_i I_i.$$

$$\text{Denominator: } W^{(1)}_i - \lambda \delta W^{(2)}_i I_i x.$$

So:

$$e_i' = -(-e_i W^{(1)}_i - \lambda \delta W^{(2)}_i l_i) / (W^{(1)}_i - \lambda \delta W^{(2)}_i l_i x) = (e_i W^{(1)}_i + \lambda \delta W^{(2)}_i l_i) / (W^{(1)}_i - \lambda \delta W^{(2)}_i l_i x).$$

Divide numerator and denominator by  $W^{(1)}_i$  (assuming non-zero):

$$e_i' = (e_i + (\lambda \delta W^{(2)}_i l_i) / W^{(1)}_i) / (1 - (\lambda \delta W^{(2)}_i l_i x) / W^{(1)}_i).$$

Let  $\alpha = \lambda \delta W^{(2)}_i l_i / W^{(1)}_i$ . Then:

$$e_i' = (e_i + \alpha) / (1 - \alpha x).$$

This is the new elbow location in terms of old elbow, the learning rate, the error, the weight, and the input  $x$ .

Alternatively, we can write it as:

$$e_i' = e_i + \frac{\alpha (e_i + x)}{1 - \alpha x} \text{? Let's see:}$$

$$e_i' - e_i = (e_i + \alpha) / (1 - \alpha x) - e_i = [e_i + \alpha - e_i (1 - \alpha x)] / (1 - \alpha x) = [e_i + \alpha - e_i + e_i \alpha x] / (1 - \alpha x) = \alpha (1 + e_i x) / (1 - \alpha x).$$

But maybe it's simpler to leave as is.

We should note that this derivation assumes that the indicator  $l_i$  remains the same after the update. In practice, if the update is small, this is a reasonable approximation.

Thus, the answer for part (d) is:

$$e_i' = \frac{e_i W^{(1)}_i + \lambda \delta W^{(2)}_i l_i}{W^{(1)}_i - \lambda \delta W^{(2)}_i l_i x} = \frac{e_i + \frac{\lambda \delta W^{(2)}_i l_i}{W^{(1)}_i}}{1 - \frac{\lambda \delta W^{(2)}_i l_i x}{W^{(1)}_i}}.$$

If we want to express  $\delta$  explicitly:  $\delta = \hat{f} - y$ .

I'll provide a boxed answer.

Note: The earlier answer I gave was too brief and not fully correct because it didn't account for the indicator and the fact that the update depends on  $W^{(2)}_i$  and  $x$ . So I'll correct it now.

## Question 5 (d) – Full Derivation of New Elbow Location After SGD Update

Given the network:

$$\hat{f}(x) = \mathbf{W}^{(2)} \Phi \left( \mathbf{W}^{(1)} x + \mathbf{b} \right),$$

where:

- $\mathbf{W}^{(1)} \in \mathbb{R}^{d \times 1}$  (column vector),
- $\mathbf{b} \in \mathbb{R}^d$ ,

- $\mathbf{W}^{(2)} \in \mathbb{R}^{1 \times d}$  (row vector),
- $\Phi$  is the element-wise ReLU:  $\Phi(\mathbf{z})_i = \max(0, z_i)$ .

Let  $\mathbf{z} = \mathbf{W}^{(1)}x + \mathbf{b}$ , and  $a_i = \Phi(z_i)$ . Then  $\hat{f}(x) = \sum_{j=1}^d W_j^{(2)} a_j$ .

The loss is:

$$\ell = \frac{1}{2} \left( \hat{f}(x) - y \right)^2.$$

### Step 1 – Gradients w.r.t. parameters of the $i$ -th neuron

We compute the gradients for  $W_i^{(1)}$  and  $b_i$ :

$$\begin{aligned} \frac{\partial \ell}{\partial W_i^{(1)}} &= \underbrace{\frac{\partial \ell}{\partial \hat{f}}}_{\hat{f} - y} \cdot \underbrace{\frac{\partial \hat{f}}{\partial a_i}}_{W_i^{(2)}} \cdot \underbrace{\frac{\partial a_i}{\partial z_i}}_{\mathbb{I}(z_i > 0)} \cdot \underbrace{\frac{\partial z_i}{\partial W_i^{(1)}}}_x \\ &= (\hat{f} - y) W_i^{(2)} \mathbb{I}(z_i > 0) x, \end{aligned}$$

$$\frac{\partial \ell}{\partial b_i} = (\hat{f} - y) W_i^{(2)} \mathbb{I}(z_i > 0).$$

### Step 2 – SGD update (learning rate $\lambda$ )

Let  $\delta = \hat{f} - y$  and  $I_i = \mathbb{I}(z_i > 0)$ . Then:

$$\begin{aligned} W_i^{(1)\prime} &= W_i^{(1)} - \lambda \delta W_i^{(2)} I_i x, \\ b_i' &= b_i - \lambda \delta W_i^{(2)} I_i. \end{aligned}$$

### Step 3 – New elbow location

The elbow for the  $i$ -th ReLU is the point where its input becomes zero:

$$z_i = W_i^{(1)}x + b_i = 0 \quad \Rightarrow \quad e_i = -\frac{b_i}{W_i^{(1)}} \quad (W_i^{(1)} \neq 0).$$

After the update, the new elbow  $e'_i$  satisfies:

$$W_i^{(1)'} e'_i + b'_i = 0 \quad \Rightarrow \quad e'_i = -\frac{b'_i}{W_i^{(1)'}}.$$

Substitute the updated parameters:

$$e'_i = -\frac{b_i - \lambda \delta W_i^{(2)} I_i}{W_i^{(1)} - \lambda \delta W_i^{(2)} I_i x}.$$

Using the original elbow  $e_i = -b_i/W_i^{(1)}$  (so  $b_i = -e_i W_i^{(1)}$ ):

$$\begin{aligned} e'_i &= -\frac{-e_i W_i^{(1)} - \lambda \delta W_i^{(2)} I_i}{W_i^{(1)} - \lambda \delta W_i^{(2)} I_i x} \\ &= \frac{e_i W_i^{(1)} + \lambda \delta W_i^{(2)} I_i}{W_i^{(1)} - \lambda \delta W_i^{(2)} I_i x}. \end{aligned}$$

Assuming  $W_i^{(1)} \neq 0$ , divide numerator and denominator by  $W_i^{(1)}$ :

$$e'_i = \boxed{\frac{e_i + \frac{\lambda \delta W_i^{(2)} I_i}{W_i^{(1)}}}{1 - \frac{\lambda \delta W_i^{(2)} I_i x}{W_i^{(1)}}}}$$

where  $\delta = \hat{f} - y$  and  $I_i = \mathbb{I}(W_i^{(1)}x + b_i > 0)$ .

### Remarks:

1. The expression assumes the activation pattern  $I_i$  remains unchanged after the small update (common for infinitesimal learning rates).
2. If  $W_i^{(1)} = 0$ , the elbow is undefined (the ReLU is either always-on or always-off).
3. The new elbow depends on the error  $\delta$ , the output-layer weight  $W_i^{(2)}$ , the input  $x$ , and the learning rate  $\lambda$ .