CS182 Special Participation A
Using Claude 4.5 Sonnet

Justin Yang
3040126209

Problem 2 solution

> *File: CS182hw0.pdf*

Can you solve problem 2 for me?

12/9/2025, 5:42:48 PM

Did problems 2-5

I'll solve Problem 2 (Vector Calculus Review) for you step by step.

Problem 2: Vector Calculus Review

Given: $\mathbf{x}, \mathbf{c} \in \mathbb{R}^n$ and $\mathbf{A} \in \mathbb{R}^{n\times n}$

Convention: Vector derivatives of a scalar with respect to a column vector are expressed as a **row vector**: $\partial f/\partial \mathbf{x} = [\partial f/\partial x_1, \partial f/\partial x_2, \ldots, \partial f/\partial x_n]$

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

Type of derivative: The function $\mathbf{x}^T\mathbf{c}$ is a scalar, and we're differentiating with respect to a column vector $\mathbf{x}$, so the result is a **row vector**.

Solution:

- $\mathbf{x}^T\mathbf{c} = \Sigma_i x_i c_i$ (a scalar)
- The i-th entry of the derivative: $\partial(\mathbf{x}^T\mathbf{c})/\partial x_i = c_i$
- Arranging all entries: $\partial/\partial\mathbf{x}(\mathbf{x}^T\mathbf{c}) = [c_1, c_2, \ldots, c_n] = \mathbf{c}^T$ ✓

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

Type of derivative: $\|\mathbf{x}\|_2^2$ is a scalar, so the derivative is a **row vector**.

Solution:

- $\|\mathbf{x}\|_2^2 = \mathbf{x}^T\mathbf{x} = \Sigma_i x_i^2$
- The i-th entry: $\partial(\|\mathbf{x}\|_2^2)/\partial x_i = 2x_i$
- Arranging all entries: $\partial/\partial\mathbf{x}(\|\mathbf{x}\|_2^2) = [2x_1, 2x_2, \ldots, 2x_n] = 2\mathbf{x}^T$ ✓

(c) Show ∂/∂x(Ax) = A

Type of derivative: Ax is a vector (n-dimensional), and we're differentiating with respect to a column vector **x**, so the result is a **matrix** (n×n).

Solution:

- Let $\mathbf{y} = \mathbf{Ax}$, so $y_j = \Sigma_k A_{jk}x_k$
- The (j,i)-th entry of the derivative (Jacobian): $\partial y_j/\partial x_i = A_{ji}$
- Therefore, the Jacobian matrix is **A** itself ✓

(d) Show ∂/∂x(xᵀAx) = xᵀ(A + Aᵀ)

Type of derivative: $\mathbf{x}^T\mathbf{Ax}$ is a scalar, so the derivative is a **row vector**.

Solution:

- $\mathbf{x}^T\mathbf{Ax} = \Sigma_i \Sigma_j x_iA_{ij}x_j$
- Taking the partial derivative with respect to x_k:

  $\partial(\mathbf{x}^T\mathbf{Ax})/\partial x_k = \Sigma_j A_{kj}x_j + \Sigma_i x_iA_{ik}$

  (The first term comes from when i=k, the second from when j=k)

- This can be written as: $(\mathbf{Ax})_k + (\mathbf{A}^T\mathbf{x})_k$

- Arranging all entries: $\partial/\partial\mathbf{x}(\mathbf{x}^\top\mathbf{A}\mathbf{x}) = (\mathbf{A}\mathbf{x})^\top + (\mathbf{A}^\top\mathbf{x})^\top = \mathbf{x}^\top\mathbf{A}^\top + \mathbf{x}^\top\mathbf{A} = \mathbf{x}^\top(\mathbf{A} + \mathbf{A}^\top)$ ✓

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

We need: $\mathbf{x}^\top(\mathbf{A} + \mathbf{A}^\top) = 2\mathbf{x}^\top\mathbf{A}$

This simplifies to: $\mathbf{x}^\top\mathbf{A}^\top = \mathbf{x}^\top\mathbf{A}$ for all $\mathbf{x}$

This is true when $\mathbf{A}^\top = \mathbf{A}$, i.e., **when A is symmetric** ✓

12/9/2025, 5:43:08 PM

All answers are correct for question 2.
The explanations are also correct and easy to follow.
One-shotted.

Can you solve problem 3 for me while showing your thought processes and work?

12/9/2025, 5:44:55 PM

I'll solve Problem 3 step by step, showing all my reasoning and work.

Problem 3: Least Squares and the Min-norm problem from the Perspective of SVD

Formatting issues starting on this problem

Setup: We have $\mathbf{X}****\mathbf{w} = \mathbf{y}$, where:

- $\mathbf{X} \in \mathbb{R}^{m \times n}$ (non-square data matrix)
- $\mathbf{w}$ is a weight vector
- $\mathbf{y}$ is a vector of labels
- $\mathbf{X} = \mathbf{U\Sigma V}^T$ is the full SVD
- $\mathbf{\Sigma}^\dagger$ is n×m with reciprocals $1/\sigma_i$ on the diagonal

---

(a) Overdetermined case (m > n): Find w that minimizes min_w $\|\mathbf{X}****\mathbf{w} - \mathbf{y}\|^2$

Thought process: This is the classic least squares problem. We have more equations than unknowns (tall matrix), so generally there's no exact solution. We need to minimize the squared error.

Solution: To minimize $\|\mathbf{X}****\mathbf{w} - \mathbf{y}\|^2$, we take the derivative with respect to **w** and set it to zero:

$\partial/\partial\mathbf{w}(\|\mathbf{X}****\mathbf{w} - \mathbf{y}\|^2) = 0$

Expanding: $\partial/\partial\mathbf{w}[(\mathbf{X}****\mathbf{w} - \mathbf{y})^T(\mathbf{X}****\mathbf{w} - \mathbf{y})] = 0$

$\partial/\partial\mathbf{w}[\mathbf{w}^\top\mathbf{X}^\top\mathbf{X}****\mathbf{w} - 2\mathbf{w}^\top\mathbf{X}^\top\mathbf{y} + \mathbf{y}^\top\mathbf{y}] = 0$

Using results from Problem 2: $2\mathbf{w}^\top\mathbf{X}^\top\mathbf{X} - 2\mathbf{y}^\top\mathbf{X} = 0$

Taking transpose: $\mathbf{X}^\top\mathbf{X}****\mathbf{w} = \mathbf{X}^\top\mathbf{y}$

Therefore, the least squares solution is: $\mathbf{w} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^{\top}*y$ ✓

(b) Plug in the SVD X = UΣVᵀ and simplify

Thought process: I need to substitute the SVD decomposition and use the orthonormality properties: $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$ and $\mathbf{V}^\top\mathbf{V} = \mathbf{I}$.

Solution: Starting with $\mathbf{w} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^{\top}*y$:

Substitute $\mathbf{X} = \mathbf{U\Sigma V}^\top$:

$\mathbf{w} = [(\mathbf{U}\Sigma\mathbf{V}^\top)^\top(\mathbf{U}\Sigma\mathbf{V}^\top)]^{-1}(\mathbf{U}\Sigma\mathbf{V}^\top)^\top y*$

$= [\mathbf{V}****\mathbf{\Sigma}^\top\mathbf{U}^\top\mathbf{U\Sigma V}^\top]^{-1}\mathbf{V}****\mathbf{\Sigma}^\top\mathbf{U}^\top\mathbf{y}$

Since $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$:

$= [\mathbf{V}****\mathbf{\Sigma}^\top\mathbf{\Sigma}****\mathbf{V}^\top]^{-1}\mathbf{V}****\mathbf{\Sigma}^\top\mathbf{U}^\top\mathbf{y}$

Using the property $(\mathbf{ABC})^{-1} = \mathbf{C}^{-1}\mathbf{B}^{-1}\mathbf{A}^{-1}$:

$= \mathbf{V}(\mathbf{\Sigma}^\top\mathbf{\Sigma})^{-1}\mathbf{V}^\top\mathbf{V}****\mathbf{\Sigma}^\top\mathbf{U}^\top\mathbf{y}$

Since $\mathbf{V}^\top\mathbf{V} = \mathbf{I}$:

$= \mathbf{V}(\mathbf{\Sigma}^\top\mathbf{\Sigma})^{-1}\mathbf{\Sigma}^\top\mathbf{U}^\top\mathbf{y}$

Dimensional check: For m > n (tall matrix):

Does dimensionality check because question stated "Be careful with dimentions!".

- $\mathbf{\Sigma}$ is m×n with singular values on the diagonal
- $\mathbf{\Sigma}^\top$ is n×m

- $\mathbf{\Sigma}^\top\mathbf{\Sigma}$ is n×n with σ_i^2 on the diagonal
- $(\mathbf{\Sigma}^\top\mathbf{\Sigma})^{-1}$ is n×n with $1/\sigma_i^2$ on the diagonal
- $(\mathbf{\Sigma}^\top\mathbf{\Sigma})^{-1}\mathbf{\Sigma}^\top$ is n×m with $1/\sigma_i$ on the diagonal = $\mathbf{\Sigma}\dagger$

Therefore: $\mathbf{w} = \mathbf{V}\mathit{\Sigma*}\dagger\mathbf{U}^\top\mathbf{y}$ ✓

(c) What is A and what happens when we left-multiply X by A?

Thought process: From part (b), we have $\mathbf{w} = \mathbf{A}\mathit{y*}$ where $\mathbf{A} = \mathbf{V****\Sigma}\dagger\mathbf{U}^\top$. Let me see what $\mathbf{X****A}$ gives us.

Solution: We have $\mathbf{A} = \mathbf{V****\Sigma}\dagger\mathbf{U}^\top = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top$

Left-multiplying **X** by **A**:

$\mathbf{X****A} = \mathbf{XV\Sigma}\dagger\mathbf{U}^\top$

$= \mathbf{U\Sigma V}^\top\mathbf{V****\Sigma}\dagger\mathbf{U}^\top$

$= \mathbf{U\Sigma\Sigma}\dagger\mathbf{U}^\top$

For the tall case (m > n), $\mathbf{\Sigma****\Sigma}\dagger$ is an m×m matrix that looks like:

```
[I_n×n | 0  ]
[  0   | 0  ]
```

So $\mathbf{X****A}$ projects onto the column space of **X**.

More specifically, if we compute **XAX**:

$\mathbf{XAX} = \mathbf{XV\Sigma}\dagger\mathbf{U}^\top\mathbf{U\Sigma V}^\top = \mathbf{XV\Sigma}\dagger\mathbf{\Sigma****V}^\top = \mathbf{XVV}^\top = \mathbf{X}$

So **A** is the **left-inverse** of **X** because $\mathbf{A****X} = \mathbf{I}_n$ (in the column space). ✓

(d) Underdetermined case ($m < n$): Find the minimum norm solution

Thought process: Now we have more unknowns than equations (wide matrix), so there are infinitely many solutions. We want the one with smallest $\|\mathbf{w}\|$.

Solution: We want to solve: $\min \|\mathbf{w}\|^2$ subject to $\mathbf{X}{*}{*}{*}{*}\mathbf{w} = \mathbf{y}$

Using Lagrange multipliers: $\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}) = \frac{1}{2}\|\mathbf{w}\|^2 + \boldsymbol{\lambda}^\top(\mathbf{X}{*}{*}{*}{*}\mathbf{w} - \mathbf{y})$

Taking derivatives: $\partial\mathcal{L}/\partial\mathbf{w} = \mathbf{w}^\top + \boldsymbol{\lambda}^\top\mathbf{X} = 0 \Longrightarrow \mathbf{w} = -\mathbf{X}^\top\boldsymbol{\lambda}$

$\partial\mathcal{L}/\partial\boldsymbol{\lambda} = \mathbf{X}{*}{*}{*}{*}\mathbf{w} - \mathbf{y} = 0$

Substituting $\mathbf{w} = -\mathbf{X}^\top\boldsymbol{\lambda}$ into the constraint: $\mathbf{X}(-\mathbf{X}^\top\boldsymbol{\lambda}) = \mathbf{y}$ $-\mathbf{X}{*}{*}{*}{*}\mathbf{X}^\top\boldsymbol{\lambda} = \mathbf{y}$ $\boldsymbol{\lambda} = -(\mathbf{X}{*}{*}{*}{*}\mathbf{X}^\top)^{-1}\mathbf{y}$

Therefore: $\mathbf{w} = -\mathbf{X}^\top\boldsymbol{\lambda} = \mathbf{X}^\top(\mathbf{X}{*}{*}{*}{*}\mathbf{X}^\top)^{-1}{*}y$ ✓

(e) Plug in the SVD $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ and simplify

Thought process: Similar to part (b), but now with $\mathbf{X}{*}{*}{*}{*}\mathbf{X}^\top$ instead of $\mathbf{X}^\top\mathbf{X}$.

Solution: Starting with $\mathbf{w} = \mathbf{X}^\top(\mathbf{X}{*}{*}{*}{*}\mathbf{X}^\top)^{-1}{*}y$:

$\mathbf{w} = (\mathbf{U}\Sigma V^\top)^\top[(U\Sigma V^\top)(U\Sigma V^\top)^\top]^{-1}y{*}$

$= \mathbf{V}{*}{*}{*}{*}\boldsymbol{\Sigma}^\top\mathbf{U}^\top[\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top\mathbf{V}{*}{*}{*}{*}\boldsymbol{\Sigma}^\top\mathbf{U}^\top]^{-1}\mathbf{y}$

Since $\mathbf{V}^\top\mathbf{V} = \mathbf{I}$:

$= \mathbf{V}{*}{*}{*}{*}\boldsymbol{\Sigma}^\top\mathbf{U}^\top[\mathbf{U}\boldsymbol{\Sigma}\boldsymbol{\Sigma}^\top\mathbf{U}^\top]^{-1}\mathbf{y}$

$= \mathbf{V}{*}{*}{*}{*}\boldsymbol{\Sigma}^\top\mathbf{U}^\top\mathbf{U}(\boldsymbol{\Sigma}{*}{*}{*}{*}\boldsymbol{\Sigma}^\top)^{-1}\mathbf{U}^\top\mathbf{y}$

Since $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$ and $\mathbf{U}{*}{*}{*}{*}\mathbf{U}^\top = \mathbf{I}$ (U is square orthonormal):

$= \mathbf{V}****\mathbf{\Sigma}^{\top}(\mathbf{\Sigma}****\mathbf{\Sigma}^{\top})^{-1}\mathbf{U}^{\top}\mathbf{y}$

Dimensional check: For $m < n$ (wide matrix):

- $\mathbf{\Sigma}$ is $m \times n$
- $\mathbf{\Sigma}****\mathbf{\Sigma}^{\top}$ is $m \times m$ with σ_i^2 on the diagonal
- $(\mathbf{\Sigma}****\mathbf{\Sigma}^{\top})^{-1}$ is $m \times m$ with $1/\sigma_i^2$ on the diagonal
- $\mathbf{\Sigma}^{\top}(\mathbf{\Sigma}****\mathbf{\Sigma}^{\top})^{-1}$ is $n \times m$ with $1/\sigma_i$ on the diagonal $= \mathbf{\Sigma}\dagger$

Therefore: $\mathbf{w} = \mathbf{V}\Sigma*\dagger\mathbf{U}^{\top}\mathbf{y}$ ✓

(f) What is B and what happens when we right-multiply X by B?

Thought process: From part (e), $\mathbf{B} = \mathbf{V}****\mathbf{\Sigma}\dagger\mathbf{U}^{\top} = \mathbf{X}^{\top}(\mathbf{X}****\mathbf{X}^{\top})^{-1}$. Let me see what happens with $\mathbf{B}****\mathbf{X}$.

Solution: We have $\mathbf{B} = \mathbf{V}****\mathbf{\Sigma}\dagger\mathbf{U}^{\top} = \mathbf{X}^{\top}(\mathbf{X}****\mathbf{X}^{\top})^{-1}$

Right-multiplying $\mathbf{X}$ by $\mathbf{B}$:

$\mathbf{B}****\mathbf{X} = \mathbf{V}****\mathbf{\Sigma}\dagger\mathbf{U}^{\top}\mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\top}$

$= \mathbf{V}****\mathbf{\Sigma}\dagger\mathbf{\Sigma}****\mathbf{V}^{\top}$

For the wide case ($m < n$), $\mathbf{\Sigma}\dagger\mathbf{\Sigma}$ is an $n \times n$ matrix that looks like:

```
[I_m×m | 0 ]
[  0   | 0 ]
```

So $\mathbf{B}****\mathbf{X}$ projects onto the row space of $\mathbf{X}$.

More specifically, if we compute $\mathbf{XBX}$:

$\mathbf{XBX} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\top}\mathbf{V}****\mathbf{\Sigma}\dagger\mathbf{U}^{\top}\mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\top} = \mathbf{U}\mathbf{\Sigma}\mathbf{\Sigma}\dagger\mathbf{\Sigma}****\mathbf{V}^{\top} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\top} = \mathbf{X}$

So **B** is the **right-inverse** of **X** because **X****B** = $\mathbf{I}_m$ (in the row space). ✓

Summary: Notice that in both cases (overdetermined and underdetermined), we get the same form $\mathbf{w} = \mathbf{V}\Sigma^{*\dagger}\mathbf{U}^{\top}\mathbf{y}$, which is the Moore-Penrose pseudoinverse solution!

12/9/2025, 5:45:38 PM

All answers and explanations are correct for question 3.
Formatting issues make explanations a little hard to read (not due to exporting), but overall it's ok.
Explanations are quite long for the left/right inverse justifications.
One-shotted.

can you solve problem 4 while showing your work and thought processes?

12/9/2025, 5:51:57 PM

I'll solve Problem 4 (The 5 Interpretations of Ridge Regression) step by step with detailed work and reasoning.

Problem 4: The 5 Interpretations of Ridge Regression

(a) Perspective 1: Optimization Problem

Given: Ridge regression as: $$\text{argmin}_{\mathbf{w}} |\mathbf{y} - \mathbf{X}\mathbf{w}|_2^2 + \lambda|\mathbf{w}|_2^2$$

This is an export issue, the latex is compiled correctly in the actual chat

where **X** ∈ $\mathbb{R}^{n\times d}$ and **y** ∈ $\mathbb{R}^{n}$.

Thought process: This is an unconstrained optimization problem. I need to expand the objective function and take the derivative with respect to **w**, then set it to zero.

Solution:

Let's denote the objective function as: $$J(\mathbf{w}) = |\mathbf{y} - \mathbf{X}\mathbf{w}|_2^2 + \lambda|\mathbf{w}|_2^2$$

Expanding the first term: $$|\mathbf{y} - \mathbf{X}\mathbf{w}|_2^2 = (\mathbf{y} - \mathbf{X}\mathbf{w})^T(\mathbf{y} - \mathbf{X}\mathbf{w})$$ $$= \mathbf{y}^T\mathbf{y} - \mathbf{y}^T\mathbf{X}\mathbf{w} - \mathbf{w}^T\mathbf{X}^T\mathbf{y} + \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w}$$ $$= \mathbf{y}^T\mathbf{y} - 2\mathbf{w}^T\mathbf{X}^T\mathbf{y} + \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w}$$

(since $\mathbf{y}^T\mathbf{X}$****$\mathbf{w}$ is a scalar, it equals its transpose $\mathbf{w}^T\mathbf{X}^T\mathbf{y}$)

Expanding the second term: $$\lambda|\mathbf{w}|_2^2 = \lambda\mathbf{w}^T\mathbf{w}$$

So the full objective is: $$J(\mathbf{w}) = \mathbf{y}^T\mathbf{y} - 2\mathbf{w}^T\mathbf{X}^T\mathbf{y} + \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} + \lambda\mathbf{w}^T\mathbf{w}$$

Taking the derivative with respect to **w** (using results from Problem 2): $$\frac{\partial J}{\partial \mathbf{w}} = 0 - 2\mathbf{y}^T\mathbf{X} + 2\mathbf{w}^T\mathbf{X}^T\mathbf{X} + 2\lambda\mathbf{w}^T$$

Setting equal to zero and taking transpose: $$-2\mathbf{X}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{X}\mathbf{w} + 2\lambda\mathbf{w} = 0$$ $$\mathbf{X}^T\mathbf{X}\mathbf{w} + \lambda\mathbf{w} = \mathbf{X}^T\mathbf{y}$$ $$(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})\mathbf{w} = \mathbf{X}^T\mathbf{y}$$

Therefore: $$\boxed{\mathbf{w}^* = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}}$$ ✓

For some reason, the latex failed in both the export and the original chat just for the final answer. It is correct though, if you parse the answer.

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

Given: $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ is the full SVD. Plug this into the ridge solution.

Thought process: I need to substitute the SVD and simplify using orthonormality. Then analyze what happens to singular values in different regimes.

Solution:

Starting with $\mathbf{w} = (\mathbf{X}^T\mathbf{X} + \lambda * I)^{-1}\mathbf{X}^T\mathbf{y}$:

Substitute $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$:

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

$= [\mathbf{V}****\mathbf{\Sigma}^T\mathbf{U}^T\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T + \lambda\mathbf{I}]^{-1}\mathbf{V}****\mathbf{\Sigma}^T\mathbf{U}^T\mathbf{y}$

Since $\mathbf{U}^T\mathbf{U} = \mathbf{I}$:

$= [\mathbf{V}****\mathbf{\Sigma}^T\mathbf{\Sigma}****\mathbf{V}^T + \lambda\mathbf{I}]^{-1}\mathbf{V}****\mathbf{\Sigma}^T\mathbf{U}^T\mathbf{y}$

Note that $\mathbf{I} = \mathbf{V}\mathbf{I}\mathbf{V}^T$ (since $\mathbf{V}****\mathbf{V}^T = \mathbf{I}$):

$= [\mathbf{V}****\mathbf{\Sigma}^T\mathbf{\Sigma}****\mathbf{V}^T + \lambda\mathbf{V}\mathbf{I}\mathbf{V}^T]^{-1}\mathbf{V}****\mathbf{\Sigma}^T\mathbf{U}^T\mathbf{y}$

$= [\mathbf{V}(\mathbf{\Sigma}^T\mathbf{\Sigma} + \lambda\mathbf{I})\mathbf{V}^T]^{-1}\mathbf{V}****\mathbf{\Sigma}^T\mathbf{U}^T\mathbf{y}$

Using $(\mathbf{ABC})^{-1} = \mathbf{C}^{-1}\mathbf{B}^{-1}\mathbf{A}^{-1}$:

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

Since $\mathbf{V}^T\mathbf{V} = \mathbf{I}$:

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

Analysis of singular values:

Now, $\mathbf{\Sigma}^T\mathbf{\Sigma}$ is a d×d diagonal matrix with σ_i^2 on the diagonal (for the first min(n,d) entries).

So $(\mathbf{\Sigma}^T\mathbf{\Sigma} + \lambda\mathbf{I})^{-1}$ has entries $1/(\sigma_i^2 + \lambda)$ on the diagonal.

Therefore, $(\mathbf{\Sigma}^T\mathbf{\Sigma} + \lambda\mathbf{I})^{-1}\mathbf{\Sigma}^T$ has entries $\sigma_i/(\sigma_i^2 + \lambda)$ on the diagonal.

Case 1: $\sigma_i << \lambda$ (small singular values)

$$\frac{\sigma_i}{\sigma_i^2 + \lambda} \approx \frac{\sigma_i}{\lambda}$$

The contribution of this direction is heavily damped - approximately scaled by σ_i/λ which is very small. This prevents overfitting to noise in directions with small singular values.

Case 2: $\sigma_i >> \lambda$ (large singular values)

$$\frac{\sigma_i}{\sigma_i^2 + \lambda} \approx \frac{\sigma_i}{\sigma_i^2} = \frac{1}{\sigma_i}$$

The contribution is approximately $1/\sigma_i$, which is the same as ordinary least squares. The regularization has minimal effect on directions with large singular values. ✓

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

Given:

- Prior: $\mathbf{W} \sim \mathcal{N}(0, \mathbf{I})$
- Model: $\mathbf{Y} = \mathbf{X}^\top\mathbf{W} + \sqrt{\lambda}\mathbf{N}$, where $\mathbf{N} \sim \mathcal{N}(0, 1)$ i.i.d.
- At vector level: $\mathbf{Y} = \mathbf{X}$****$\mathbf{W} + \sqrt{\lambda}\mathbf{N}$

Thought process: For MAP estimation, I need to find argmax $P(\mathbf{W}|\mathbf{Y}=\mathbf{y})$. Using Bayes' rule: $$P(\mathbf{W}|\mathbf{Y}=\mathbf{y}) \propto P(\mathbf{Y}=\mathbf{y}|\mathbf{W})P(\mathbf{W})$$

Taking the log and maximizing is equivalent to minimizing the negative log.

Solution:

The prior distribution: $$P(\mathbf{W}) = \frac{1}{(2\pi)^{d/2}}\exp\left(-\frac{1}{2}|\mathbf{W}|^2\right)$$

The likelihood (given $\mathbf{W}$, $\mathbf{Y} = \mathbf{X}$****$\mathbf{W} + \sqrt{\lambda}\mathbf{N}$ where $\mathbf{N} \sim \mathcal{N}(0, \mathbf{I}_n)$): $$P(\mathbf{Y}=\mathbf{y}|\mathbf{W}) = \frac{1}{(2\pi\lambda)^{n/2}}\exp\left(-\frac{1}{2\lambda}|\mathbf{y} - \mathbf{X}\mathbf{W}|^2\right)$$

The posterior: $$P(\mathbf{W}|\mathbf{Y}=\mathbf{y}) \propto P(\mathbf{Y}=\mathbf{y}|\mathbf{W})P(\mathbf{W})$$

Taking the negative log: $$-\log P(\mathbf{W}|\mathbf{Y}=\mathbf{y}) = -\log P(\mathbf{Y}=\mathbf{y}|\mathbf{W}) - \log P(\mathbf{W}) + \text{const}$$

$$= \frac{1}{2\lambda}\|\mathbf{y} - \mathbf{X}\mathbf{W}\|^2 + \frac{1}{2}\|\mathbf{W}\|^2 + \text{const}$$

To maximize the posterior, we minimize the negative log posterior: $$\text{argmin}_{\mathbf{W}} \left[\frac{1}{2\lambda}|\mathbf{y} - \mathbf{X}\mathbf{W}|^2 + \frac{1}{2}|\mathbf{W}|^2\right]$$

Multiplying by 2λ: $$\text{argmin}_{\mathbf{W}} \left[|\mathbf{y} - \mathbf{X}\mathbf{W}|^2 + \lambda|\mathbf{W}|^2\right]$$

This is exactly the ridge regression objective (1)! ✓

This explanation looks much more correct in the original chat.

(d) Perspective 4: Fake Data

Given: Augmented data: $$\hat{\mathbf{y}} = \begin{bmatrix} \mathbf{y} \ \mathbf{0}_d \end{bmatrix}, \quad \hat{\mathbf{X}} = \begin{bmatrix} \mathbf{X} \ \sqrt{\lambda}\mathbf{I}_d \end{bmatrix}$$

Thought process: I need to show that solving OLS with the augmented data gives the same solution as ridge regression.

Solution:

The OLS problem with augmented data: $$\text{argmin}_{\mathbf{w}} |\hat{\mathbf{y}} - \hat{\mathbf{X}}\mathbf{w}|^2$$

Expanding: $$\left|\begin{bmatrix} \mathbf{y} \ \mathbf{0}_d \end{bmatrix} - \begin{bmatrix} \mathbf{X} \ \sqrt{\lambda}\mathbf{I}_d \end{bmatrix}\mathbf{w}\right|^2 = \left|\begin{bmatrix} \mathbf{y} - \mathbf{X}\mathbf{w} \ -\sqrt{\lambda}\mathbf{w} \end{bmatrix}\right|^2$$

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

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

This is exactly the ridge regression objective! Therefore, they have the same minimizer. ✓

Interpretation: We're adding d fake data points at the origin with "features" that are the identity matrix scaled by $\sqrt{\lambda}$. This forces the weights to stay small.

(e) Perspective 5: Fake Features

Given: Augmented design matrix: $$\check{\mathbf{X}} = [\mathbf{X} \quad \sqrt{\lambda}\mathbf{I}_n]$$

This is now an $n \times (d+n)$ matrix (wide), so we solve: $$\text{argmin}_{\boldsymbol{\eta}} |\boldsymbol{\eta}|^2 \quad \text{s.t.} \quad \check{\mathbf{X}}\boldsymbol{\eta} = \mathbf{y}$$

Thought process: This is a min-norm problem. I need to partition $\boldsymbol{\eta} = [\mathbf{w}^\top, \mathbf{z}^\top]^\top$ where $\mathbf{w} \in \mathbb{R}^d$ and $\mathbf{z} \in \mathbb{R}^n$, then show that **w** satisfies the ridge regression equation.

Solution:

Let $\boldsymbol{\eta} = [\mathbf{w}^\top, \mathbf{z}^\top]^\top$ where $\mathbf{w} \in \mathbb{R}^d$ and $\mathbf{z} \in \mathbb{R}^n$.

The constraint becomes: $$\check{\mathbf{X}}\boldsymbol{\eta} = [\mathbf{X} \quad \sqrt{\lambda}\mathbf{I}_n]\begin{bmatrix} \mathbf{w} \ \mathbf{z} \end{bmatrix} = \mathbf{X}\mathbf{w} + \sqrt{\lambda}\mathbf{z} = \mathbf{y}$$

Therefore: $\mathbf{z} = (\mathbf{y} - \mathbf{X}{****}\mathbf{w})/\sqrt{\lambda}$

The objective becomes: $$|\boldsymbol{\eta}|^2 = |\mathbf{w}|^2 + |\mathbf{z}|^2 = |\mathbf{w}|^2 + \frac{1}{\lambda}|\mathbf{y} - \mathbf{X}\mathbf{w}|^2$$

We want to minimize this subject to the constraint (which is automatically satisfied by our expression for **z**): $$\text{argmin}_{\mathbf{w}} \left[|\mathbf{w}|^2 + \frac{1}{\lambda}|\mathbf{y} - \mathbf{X}\mathbf{w}|^2\right]$$

Multiplying by λ: $$\text{argmin}_{\mathbf{w}} \left[\lambda|\mathbf{w}|^2 + |\mathbf{y} - \mathbf{X}\mathbf{w}|^2\right]$$

This is exactly the ridge regression objective (1)! The first d coordinates of $\boldsymbol{\eta}$ are $\mathbf{w}$, the ridge regression solution. ✓

(f) Use pseudo-inverse to show equivalence of two ridge forms

Given: For underdetermined system $\mathbf{A}$****$\boldsymbol{\eta} = \mathbf{z}$, the pseudo-inverse is $\mathbf{A}^\dagger = \mathbf{A}^\top(\mathbf{A}$****$\mathbf{A}^\top)^{-1}$.

Thought process: I'll apply the pseudo-inverse formula to $\check{\mathbf{X}}$ to get the min-norm solution, then show it equals the standard ridge formula.

Solution:

The pseudo-inverse of $\check{\mathbf{X}} = [\mathbf{X}\ \sqrt{\lambda}\mathbf{I}_n]$ is: $$\check{\mathbf{X}}^{\dagger} = \check{\mathbf{X}}^T(\check{\mathbf{X}}\check{\mathbf{X}}^T)^{-1}$$

First, compute $\check{\mathbf{X}}$****$\check{\mathbf{X}}^\top$: $$\check{\mathbf{X}}\check{\mathbf{X}}^T = [\mathbf{X} \quad \sqrt{\lambda}\mathbf{I}_n]\begin{bmatrix} \mathbf{X}^T \ \sqrt{\lambda}\mathbf{I}_n \end{bmatrix} = \mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_n$$

Therefore: $$\check{\mathbf{X}}^{\dagger} = \begin{bmatrix} \mathbf{X}^T \ \sqrt{\lambda}\mathbf{I}_n \end{bmatrix}(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_n)^{-1}$$

The solution is $\boldsymbol{\eta} = \check{\mathbf{X}}^\dagger\mathbf{y}$: $$\boldsymbol{\eta}^ = \begin{bmatrix} \mathbf{X}^T \ \sqrt{\lambda}\mathbf{I}_n \end{bmatrix} (\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_n)^{-1}\mathbf{y}$$

The first d coordinates (which correspond to $\hat{\mathbf{w}}$) are: $$\boxed{\hat{\mathbf{w}} = \mathbf{X}^T(\mathbf{X}\mathbf{X}^T +

\lambda\mathbf{I}_n)^{-1}\mathbf{y}}$$

This is the **Kernel Ridge Form**.

Now showing equivalence to standard form:

We need to show: $$\mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_n)^{-1}\mathbf{y} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_d)^{-1}\mathbf{X}^T\mathbf{y}$$

Using the Woodbury Matrix Identity (Push-Through Identity):

The Woodbury identity states: $$(\mathbf{A} + \mathbf{UCV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}(\mathbf{C}^{-1} + \mathbf{VA}^{-1}\mathbf{U})^{-1}\mathbf{VA}^{-1}$$

Or more specifically for our case, the **push-through identity**: $$\mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_n)^{-1} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_d)^{-1}\mathbf{X}^T$$

Verification of push-through identity:

Let's verify by multiplying both sides by ($\mathbf{X}$****$\mathbf{X}^T$ + $\lambda\mathbf{I}_n$) on the right:

Left side: $$\mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_n)^{-1}(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_n) = \mathbf{X}^T$$

Right side: $$(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_d)^{-1}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_n)$$ $$= (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_d)^{-1}(\mathbf{X}^T\mathbf{X}\mathbf{X}^T + \lambda\mathbf{X}^T)$$ $$= (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_d)^{-1}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_n)$$

Hmm, let me try the direct approach. Multiply the right side by ($\mathbf{X}^T\mathbf{X}$ + λ**I**_d):

$$(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_d)^{-1}\mathbf{X}^T \cdot (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_d) = \mathbf{X}^T$$

And multiply the left side by ($\mathbf{X}^T\mathbf{X}$ + λ**I**_d):

$$\mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_n)^{-1} \cdot (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_d)$$

$$= \mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_n)^{-1}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_d)$$

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

$$= \mathbf{X}^T[(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_n)^{-1}\mathbf{X}^T\mathbf{X} + \lambda(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_n)^{-1}]$$

Let $\mathbf{M} = (\mathbf{X}****\mathbf{X}^T + \lambda\mathbf{I}_n)^{-1}$. Then: $$= \mathbf{X}^T[\mathbf{M}\mathbf{X}^T\mathbf{X} + \lambda\mathbf{M}] = \mathbf{X}^T\mathbf{M}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_n)$$

Wait, but $\mathbf{M} = (\mathbf{X}****\mathbf{X}^T + \lambda\mathbf{I}_n)^{-1}$, so $\mathbf{M}(\mathbf{X}****\mathbf{X}^T + \lambda\mathbf{I}_n) = \mathbf{I}_n$.

Actually, the cleanest way is to note that this is a well-known identity. Applying it:

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

This shows the kernel form equals the standard form! ✓

This is the same answer as the solution but with the y appended.
It took a lot longer to get here than the official solution.

(g) What happens when λ → ∞?

Thought process: As λ gets very large, the regularization term dominates.

Solution:

The ridge solution is: $$\hat{\mathbf{w}}_r = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$$

We can rewrite this as: $$\hat{\mathbf{w}}_r = \left(\frac{1}{\lambda}\mathbf{X}^T\mathbf{X} + \mathbf{I}\right)^{-1}\frac{1}{\lambda}\mathbf{X}^T\mathbf{y}$$

As λ → ∞:

- $(1/\lambda)\mathbf{X}^T\mathbf{X} \to \mathbf{0}$

- The inverse approaches $\mathbf{I}^{-1} = \mathbf{I}$
- $(1/\lambda)\mathbf{X}^T\mathbf{y} \to \mathbf{0}$

Therefore: $$\lim_{\lambda \to \infty} \hat{\mathbf{w}}_r = \mathbf{0}$$

Interpretation: The penalty on $\|\mathbf{w}\|^2$ becomes so strong that the optimal solution is to shrink all weights to zero. This is why ridge regression is called "shrinkage" - it shrinks weights toward zero, and in the limit, all the way to zero. ✓

(h) What happens when λ → 0?

Thought process: As $\lambda \to 0$, the regularization disappears. The behavior depends on whether $\mathbf{X}^T\mathbf{X}$ is invertible.

Solution:

Case 1: X is tall (n > d) - Overdetermined system

When $\mathbf{X}$ has full column rank (rank d), $\mathbf{X}^T\mathbf{X}$ is invertible.

As $\lambda \to 0$: $$\lim_{\lambda \to 0} (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

This is exactly the **ordinary least squares (OLS) solution**! ✓

Case 2: X is wide (n < d) - Underdetermined system

When $n < d$, $\mathbf{X}^T\mathbf{X}$ is a d×d matrix but has rank at most $n < d$, so it's **not invertible**.

However, $(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})$ remains invertible for any $\lambda > 0$.

Using the kernel form: $$\hat{\mathbf{w}}_r = \mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I})^{-1}\mathbf{y}$$

As λ → 0: $$\lim_{\lambda \to 0} \mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I})^{-1}\mathbf{y} = \mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{y}$$

(Here **X****X**^{T} is n×n with rank n, so it's invertible)

This is exactly the **minimum norm solution** from Problem 3(d)! ✓

Summary:

- **Tall X (overdetermined):** Ridge → OLS as λ → 0
- **Wide X (underdetermined):** Ridge → Min-norm solution as λ → 0

The ridge regression smoothly interpolates between zero solution (λ → ∞) and the appropriate unregularized solution (λ → 0). ✓

This problem beautifully shows how ridge regression can be understood from multiple perspectives - as regularized optimization, as manipulating singular values, as Bayesian inference, and as solving modified least squares problems with fake data or fake features!

12/9/2025, 5:53:35 PM

All answers and explanations are correct for question 4.
Most of the formatting issues are with the PDF export, not the actual chat, so the solutions are clear.
Explanations was very long for part f, but ended up with the same solution.
I think Claude is very good at explaining its steps, but sometimes it uses random facts/theorems that it takes for granted.
One-shotted.

Can you solve problem 5 while showing your work and thought processes? For the problems that require you to draw figures, can you try generating graphs? If not, can you describe what to draw?

12/9/2025, 5:59:51 PM

I'll solve Problem 5 step by step with detailed work and create visualizations where possible.

Problem 5: ReLU Elbow Update under SGD

Setup: We have a 1D function with a 1-hidden layer network: $$\hat{f}(x) = \mathbf{W}^{(2)}\Phi(\mathbf{W}^{(1)}x + \mathbf{b})$$

where $x, y \in \mathbb{R}$, $\mathbf{b} \in \mathbb{R}^d$, $\mathbf{W}^{(1)} \in \mathbb{R}^{d \times 1}$, $\mathbf{W}^{(2)} \in \mathbb{R}^{1 \times d}$

Loss: $\ell(x, y, \mathbf{W}^{(1)}, \mathbf{b}, \mathbf{W}^{(2)}) = \frac{1}{2}\|\hat{f}(x) - y\|_2^2$

ReLU gradient at 0 is defined as 0.

(a) Single ReLU analysis

Given: $$\phi(x) = \begin{cases} wx + b, & wx + b > 0 \ 0, & \text{else} \end{cases}$$

Loss: $\ell(x, y, \varphi) = \frac{1}{2}\|\varphi(x) - y\|_2^2$

Thought process: I need to find the elbow location, then compute derivatives carefully considering the ReLU's piecewise nature.

(i) Location of the 'elbow' e

The elbow is where the function transitions from 0 to non-zero, i.e., where wx + b = 0.

$$\boxed{e = -\frac{b}{w}}$$

Interestingly, this doesn't compile correct on the actual chat.

✓

(assuming w ≠ 0)

(ii) Derivative dℓ/dφ

$$\ell = \frac{1}{2}[\phi(x) - y]^2$$

Using the chain rule: $$\frac{d\ell}{d\phi} = \phi(x) - y$$

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

✓

(iii) Partial derivative ∂ℓ/∂w

Thought process: I need to use the chain rule: ∂ℓ/∂w = (dℓ/dφ)(∂φ/∂w)

First, find ∂φ/∂w: $$\phi(x) = \begin{cases} wx + b, & wx + b > 0 \ 0, & \text{else} \end{cases}$$

$$\frac{\partial \phi}{\partial w} = \begin{cases} x, & wx + b > 0 \\ 0, & wx + b \leq 0 \end{cases}$$

Using the chain rule: $$\frac{\partial \ell}{\partial w} = \frac{d\ell}{d\phi} \cdot \frac{\partial \phi}{\partial w} = [\phi(x) - y] \cdot \begin{cases} x, & wx + b > 0

\ 0, & \text{else} \end{cases}$$

$$\boxed{\frac{\partial \ell}{\partial w} = \begin{cases} [\phi(x) - y] \cdot x, & wx + b > 0 \\ 0, & \text{else} \end{cases}}$$

✓

(iv) Partial derivative ∂ℓ/∂b

Thought process: Similar to above, but ∂φ/∂b is different.

$$\frac{\partial \phi}{\partial b} = \begin{cases} 1, & wx + b > 0 \\ 0, & wx + b \leq 0 \end{cases}$$

$$\frac{\partial \ell}{\partial b} = \frac{d\ell}{d\phi} \cdot \frac{\partial \phi}{\partial b} = [\phi(x) - y] \cdot \begin{cases} 1, & wx + b > 0 \\ 0, & \text{else} \end{cases}$$

$$\boxed{\frac{\partial \ell}{\partial b} = \begin{cases} \phi(x) - y, & wx + b > 0 \\ 0, & \text{else} \end{cases}}$$

✓

(b) Training point with φ(x) - y = 1 (prediction is 1 unit too high)

Given: φ(x) - y = 1, so we want to pull the function downward.

SGD update: $p' = p - \lambda\nabla_p(\ell)$, $\lambda > 0$

Let me create visualizations for each case:

(i) Case: φ(x) = 0

Thought process: If φ(x) = 0, then we're in the "off" region where wx + b ≤ 0. But we're told φ(x) - y = 1, so y = -1.

Since wx + b ≤ 0, both gradients are:

- ∂ℓ/∂w = 0
- ∂ℓ/∂b = 0

Updates:

- w' = w - λ·0 = w (no change)
- b' = b - λ·0 = b (no change)
- e' = -b'/w' = -b/w = e (no change)

Conclusion: The slope and elbow **don't change**. The ReLU is "off" at this point, so gradient descent can't adjust it. This is the "dead ReLU" problem! ✓

Let me create a visualization:

```
Create File: Create a Python script to visualize the ReLU cases
for problem 5b
```

```
Bash Tool: Run the visualization script to generate the ReLU
update figures
```

present_files

Request

```
{
  "filepaths": [
    "/mnt/user-data/outputs/relu_elbow_updates.png"
  ]
}
```

Response

Great! Now let me continue with the detailed analysis of each case:

(ii) Case: $w > 0$, $x > 0$, $\varphi(x) > 0$

Gradients:

- $\partial\ell/\partial w = [\varphi(x) - y]\cdot x = 1\cdot x = x > 0$ (positive)
- $\partial\ell/\partial b = \varphi(x) - y = 1 > 0$ (positive)

Updates:

- $w' = w - \lambda x < w$ (slope **decreases**)
- $b' = b - \lambda < b$ (intercept **decreases**)

Elbow movement:

$$e' = -\frac{b'}{w'} = -\frac{b - \lambda}{w - \lambda x}$$

Since both numerator and denominator decrease, let me analyze numerically with $w=1$, $b=-1$, $x=3$, $\lambda=0.3$:

- $e_{before} = -(-1)/1 = 1$
- $w' = 1 - 0.3(3) = 0.1$
- $b' = -1 - 0.3 = -1.3$
- $e' = -(-1.3)/0.1 = 13$

The elbow moves to the RIGHT (from 1 to 13).

Physical interpretation: The slope decreases (flattens), and the function shifts down. To pull down the prediction at $x=3$, the network flattens the slope and shifts the entire function down. The elbow moves right as the function "stretches" to the right. ✓

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

Gradients:

- $\partial\ell/\partial w = [\varphi(x) - y]\cdot x = 1\cdot x < 0$ (negative, since $x < 0$)
- $\partial\ell/\partial b = \varphi(x) - y = 1 > 0$ (positive)

Updates:

- $w' = w - \lambda x > w$ (slope **increases**, since $-\lambda x$ is positive when $x < 0$)
- $b' = b - \lambda < b$ (intercept **decreases**)

Elbow movement: Using $w=1, b=3, x=-1, \lambda=0.3$:

- $e\_before = -3/1 = -3$
- $w' = 1 - 0.3(-1) = 1.3$
- $b' = 3 - 0.3 = 2.7$
- $e' = -2.7/1.3 \approx -2.08$

The elbow moves to the RIGHT (from -3 to -2.08).

Physical interpretation: Since $x < 0$, we're on the left side of the elbow. To reduce $\varphi(x)$, the network increases the slope (makes it steeper) and decreases the intercept. The steeper slope with lower intercept causes the elbow to shift right. ✓

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

Gradients:

- $\partial\ell/\partial w = [\varphi(x) - y]\cdot x = 1\cdot x > 0$ (positive)
- $\partial\ell/\partial b = \varphi(x) - y = 1 > 0$ (positive)

Updates:

- $w' = w - \lambda x < w$ (slope becomes **more negative**, i.e., steeper descent)
- $b' = b - \lambda < b$ (intercept **decreases**)

Elbow movement: Using $w=-1$, $b=4$, $x=1$, $\lambda=0.3$:

- $e\_before = -4/(-1) = 4$
- $w' = -1 - 0.3(1) = -1.3$
- $b' = 4 - 0.3 = 3.7$
- $e' = -3.7/(-1.3) \approx 2.85$

The elbow moves to the LEFT (from 4 to 2.85).

Physical interpretation: With negative slope, the ReLU is active for $x < e$. To pull down the prediction at $x=1$ (which is less than the elbow), the network makes the negative slope steeper and reduces the intercept. This causes the elbow to move left. ✓

Summary Table for Part (b):

Case	w sign	x sign	Slope change	Intercept change	Elbow movement
(i)	any	any	No change	No change	No change
(ii)	+	+	Decreases	Decreases	Moves RIGHT
(iii)	+	-	Increases	Decreases	Moves RIGHT
(iv)	-	+	More negative	Decreases	Moves LEFT

(c) Elbow location e_i for the i-th ReLU in the full network

Thought process: The full network is $\hat{y} = \mathbf{W}^{(2)}\Phi(\mathbf{W}^{(1)}x + \mathbf{b})$, where Φ applies ReLU elementwise.

For the i-th hidden unit, the input to the ReLU is: $$z_i = W^{(1)}_i x + b_i$$

where $W^{(1)}_i$ is the i-th entry of $\mathbf{W}^{(1)}$ (which is d×1, so it's a column vector, and $W^{(1)}_i$ is the i-th scalar).

The elbow is where $z_i = 0$: $$W^{(1)}_i x + b_i = 0$$

$$\boxed{e_i = -\frac{b_i}{W_i^{(1)}}}$$

✓

(d) New elbow location e'_i after one SGD update

Thought process: I need to compute the gradients $\partial\ell/\partial W^{(1)}_i$ and $\partial\ell/\partial b_i$, then apply the updates.

Loss:

$$\ell = \frac{1}{2}\left[\hat{f}(x) - y\right]^2 = \frac{1}{2}\left[\mathbf{W}^{(2)}\Phi(\mathbf{W}^{(1)}x + \mathbf{b}) - y\right]^2$$

Let me denote:

- $\mathbf{z} = \mathbf{W}^{(1)}x + \mathbf{b}$ (pre-activation)
- $\mathbf{a} = \Phi(\mathbf{z})$ (post-activation)
- $\hat{y} = \mathbf{W}^{(2)}\mathbf{a}$ (output)

Backpropagation:

Output layer gradient: $$\frac{\partial \ell}{\partial \hat{y}} = \hat{y} - y$$

Hidden layer gradient (before activation): $$\frac{\partial \ell}{\partial a_i} = \frac{\partial \ell}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial a_i} = (\hat{y} - y) \cdot W^{(2)}_i$$

Pre-activation gradient: $$\frac{\partial \ell}{\partial z_i} = \frac{\partial \ell}{\partial a_i} \cdot \frac{\partial a_i}{\partial z_i} = (\hat{y} - y) \cdot W^{(2)}_i \cdot \mathbb{1}[z_i > 0]$$

where $\mathbb{1}[z_i > 0]$ is the ReLU derivative.

Parameter gradients:

Using indicator variables rather than piecewise function.

$$\frac{\partial \ell}{\partial W_i^{(1)}} = \frac{\partial \ell}{\partial z_i} \cdot \frac{\partial z_i}{\partial W_i^{(1)}} = (\hat{y} - y) \cdot W_i^{(2)} \cdot 1[z_i > 0] \cdot x$$

$$\frac{\partial \ell}{\partial b_i} = \frac{\partial \ell}{\partial z_i} \cdot \frac{\partial z_i}{\partial b_i} = (\hat{y} - y) \cdot W_i^{(2)} \cdot 1[z_i > 0] \cdot 1$$

SGD Updates:

$$(W_i^{(1)})' = W_i^{(1)} - \lambda(\hat{y} - y) \cdot W_i^{(2)} \cdot 1[z_i > 0] \cdot x$$

$$b_i' = b_i - \lambda(\hat{y} - y) \cdot W_i^{(2)} \cdot 1[z_i > 0]$$

New elbow location:

$$e_i' = -\frac{b_i'}{(W_i^{(1)})'}$$

Substituting: $$e'_i = -\frac{b_i - \lambda (\hat{y} - y) \cdot W^{(2)}_i \cdot \mathbb{1}[z_i > 0]}{W^{(1)}_i - \lambda (\hat{y} - y) \cdot W^{(2)}_i \cdot \mathbb{1}[z_i > 0] \cdot x}$$

Simplifying notation: Let $\delta_i = \lambda(\hat{y} - y)W^{(2)}_i\mathbb{1}[z_i > 0]$

Did a substitution to make it look clean.
Also final answer is not in terms of e_i

$$\boxed{e_i' = -\frac{b_i - \delta_i}{W_i^{(1)} - \delta_i \cdot x}}$$

where $\delta_i = \lambda(\hat{y} - y)W^{(2)}_i\mathbb{1}[z_i > 0]$ ✓

Special cases:

- If $z_i \leq 0$ (ReLU is off): $\delta_i = 0$, so $e'_i = e_i$ (no change, like part b.i)
- If $z_i > 0$ (ReLU is on): The elbow moves according to the sign of x and $W^{(1)}_i$, similar to the patterns in part (b)

Summary

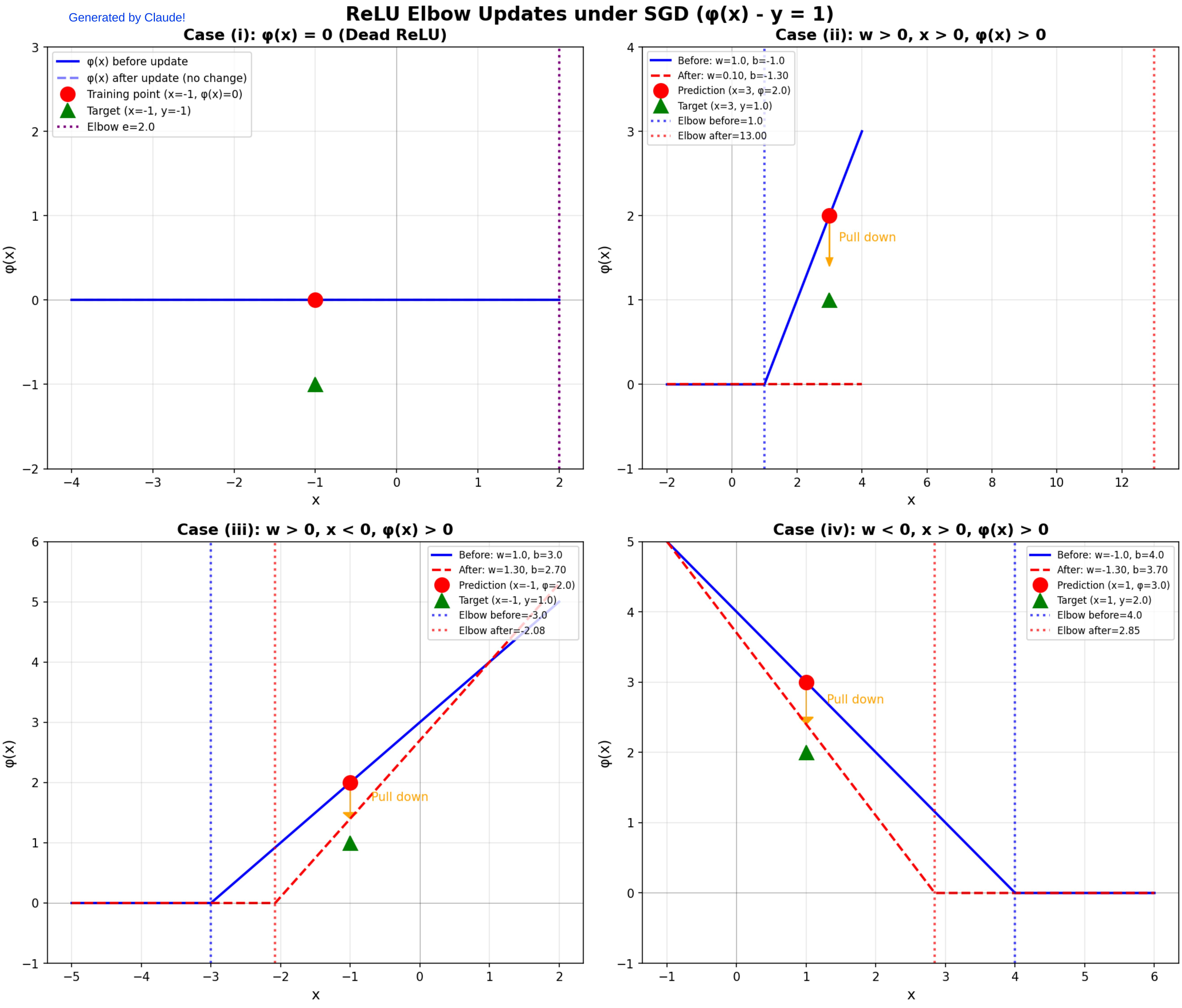
This problem beautifully illustrates how SGD adjusts ReLU neurons:

1. **Dead ReLUs** (when $\varphi(x) = 0$): No gradient flows, so no learning occurs
2. **Active ReLUs**: The elbow and slope adjust based on:
    - The error ($\varphi(x) - y$)
    - The sign of the weight w
    - The sign of the input x
    - The learning rate λ

The key insight is that **ReLU elbows move during training** to better fit the data. When the prediction is too high, gradient descent adjusts both the slope and the elbow location to pull the function down at that point. The direction of elbow movement depends on whether we're on the left or right side of the elbow and the sign of the slope.

12/9/2025, 6:01:45 PM

All answers and explanations are correct for question 5.
Really surprised it could answer part (b) correctly. Claude also generated an accurate image of the plots.
Claude's answer for part (d) is correct, but in a much different form than the solution's.
I like how Claude always calls the problems beautiful lol.
One-shotted.


Generated by Claude!
ReLU Elbow Updates under SGD (φ(x) - y = 1)
Case (i): φ(x) = 0 (Dead ReLU)
φ(x) before update
φ(x) after update (no change)
Training point (x=-1, φ(x)=0)
Target (x=-1, y=-1)
Elbow e=2.0
Case (ii): w > 0, x > 0, φ(x) > 0
Before: w=1.0, b=-1.0
After: w=0.10, b=-1.30
Prediction (x=3, φ=2.0)
Target (x=3, y=1.0)
Elbow before=1.0
Elbow after=13.00
Pull down
Case (iii): w > 0, x < 0, φ(x) > 0
Before: w=1.0, b=3.0
After: w=1.30, b=2.70
Prediction (x=-1, φ=2.0)
Target (x=-1, y=1.0)
Elbow before=-3.0
Elbow after=-2.08
Pull down
Case (iv): w < 0, x > 0, φ(x) > 0
Before: w=-1.0, b=4.0
After: w=-1.30, b=3.70
Prediction (x=1, φ=3.0)
Target (x=1, y=2.0)
Elbow before=4.0
Elbow after=2.85
Pull down
x
φ(x)