

ML Study Sheet Completion Request

12/07/2025 16:29:03

ROLE: You are an expert instructor for EECS/CS machine learning courses...

Thinking: Developing Study Materials I've been focusing on constructing a...

hw01_question - PDF Here are the questions

Thinking: Reviewing the Scope I'm currently focused on the prompt's core...

Using your reasonings from above, what is your answer for problem 1a? Just the...

Thinking: Revisiting Problem 1a I'm revisiting the analysis of Problem 1a, ensurin...

What is your answer for 1b?

The answer for 1(b) is derived from solving for the number of iterations t needed...

Now give me just the answers for the rest of problem 1 please

Thinking: Confirming the Solutions I've completed my analysis of the user's...

Let's go to problem 2 now please

Problem 2: Stochastic Gradient Descent (when it is possible to interpolate) (1)...

Please proceed to reiterate all the ANSWERS of part 3. Your responses should...

Problem 3: Accelerating Gradient Descent with Momentum 3(a) $x_t[i]=\dots$

Your answers for f and g are suspect. Please give concise answers for both again...

Thinking: Revising Stability Analysis I'm revisiting the stability conditions and...

It seems you are confused. This is question 3: 3. Accelerating Gradient Descent...

This response provides the complete, detailed solutions for Problem 3,...

Now, question 4 please: 4. Optimizers Homework 1, © Faculty teaching EECS 18...

Thinking: Solving Adam Optimizer I'm currently focused on completing Problem...

How about this question now?: 5. Regularization and Instance Noise Say we have...

Thinking: Revisiting Regularization I've been mulling over the implications of noise...

6. General Case Tikhonov Regularization Consider the optimization problem: $\min_{x \in \mathbb{R}^n}$ $\frac{1}{2} \|Ax - b\|^2 + \frac{\lambda}{2} \|x\|^2$

Thinking: Tackling Tikhonov I'm focusing on the general case of Tikhonov...

What about this one: 7. An Alternate MAP Interpretation of Ridge Regression...

Thinking: Calculating Problem 7 I've begun the derivation for Problem 7, focusing...



Here's the prompt I gave GPT to refine.



ROLE: You are an expert instructor for EECS/CS machine learning courses (including CS182/EECS182).

OBJECTIVE: Your task is to complete all non-coding portions of the following study sheet.

Your Requirements When writing your answers:

Give fully detailed explanations suitable for an A-level exam answer.

Derivations must be step-by-step

Use math notation

Define every symbol

Explain intuition behind each step

Do NOT write or modify any code.

Only fill in conceptual, mathematical, and explanatory parts.

Use clear section headers, bullet points, and boxed key results.

Always relate mathematical results back to ML intuition, e.g. optimization, generalization, gradient flow, eigenvalue behavior, etc.

If the sheet includes proofs, write them rigorously but also add a plain-English explanation.

If the sheet asks for interpretations, provide diagrams (ASCII is fine), analogies, and examples.

Your Output Format

For each question:

Problem X



(1) Conceptual Explanation

(2) Step-by-step Derivation

(3) Final Answer Box

(4) Intuition / Interpretation



Thinking:

Developing Study Materials

I've been focusing on constructing a comprehensive study sheet. My latest considerations involve the non-coding aspects, primarily theoretical understanding and conceptual frameworks. I'm exploring various organizational methods to enhance clarity and retention. I'm also considering potential pitfalls and areas where students might struggle.

Assessing Instructions' Readiness

I've been meticulously reviewing the instructions for the study sheet. They're quite detailed! It's clear that it requires A-level quality explanations, step-by-step derivations, and LaTeX formatting. I'm focusing on defining all symbols and explaining the intuitive connections between the math and machine learning concepts. My next step is to confirm whether the study material itself has been provided or if I should expect it shortly.

Verifying Sheet Availability

I'm now certain the study sheet, or at least its initial problems, are missing. The prompt structure and the "following" in "complete the following study sheet" clarified this. My current plan is to explicitly ask the user for the study sheet itself so I can start tackling the non-coding aspects as instructed. I am ready to begin.

Addressing Missing Study Content

I am now certain that the study sheet is missing, based on the prompt's structure. I've re-evaluated the phrasing "complete the following study sheet," confirming that the

content should be provided. I need to ask for the study sheet itself so I can start working on the non-coding components. I am ready to begin.

Confirming Input Availability

I'm now thinking the user might have forgotten to include the study sheet's actual content. To ensure I'm providing the most relevant assistance as an EECS/CS182 expert, I'm going to politely ask for the sheet's text or questions. To build user confidence, I'll also include a sample answer to a typical CS182 question, such as backpropagation. I will focus on adhering to the instructions: LaTeX formatting, step-by-step derivations, conceptual explanations, and zero code.

Refining Response Strategy

I'm now revising my response strategy. While I initially planned to provide a sample answer, I've decided to be more direct. My current thinking is to confirm the instructions are clear and politely request the study sheet content. I'm focusing on the "Your Output Format" instructions to ensure my response is aligned with the desired output. I'm prepared to adapt further if the missing content is provided in a subsequent interaction.

Clarifying User Intent

I'm now certain the user intended to give me problems. I'll provide a concise message confirming my readiness and requesting the questions. While the prompt suggested the study sheet was incomplete, the phrasing confirms it should be provided. I'm focusing on ensuring I'm aligned with the intended workflow. My primary goal now is to establish a clear starting point for addressing the provided content.

Preparing Final Response

I've streamlined my response further. I'm now certain the best approach is to confirm my understanding of the role, constraints, and request the necessary input. To demonstrate my capabilities, I'll briefly outline how I'll approach a derivation, aiming to showcase my comprehension of the task. My focus remains on efficiency and clear communication to establish a solid foundation for the subsequent problem-solving process. My primary goal now is to efficiently move into addressing the learning materials.

Finalizing Request for Input

I've decided to be direct. I'm certain the user intended to provide problems. My current strategy is a concise message confirming my readiness and a clear request for the

study sheet content. To quickly demonstrate my understanding, I'll briefly outline how I'll approach a derivation, focusing on adhering to instructions and highlighting my grasp of the task. The goal is now to efficiently begin addressing the study materials. I believe this is best.

Preparing for Input

I've decided on the best approach. I will now issue a concise message confirming my readiness and a clear request for the study sheet's problems. To demonstrate my capabilities immediately, I will briefly outline how I plan to approach a derivation, focusing on adhering to instructions and highlighting my grasp of the core concepts. The objective is to efficiently get the materials to address the learning problems as intended.

Confirming Readiness and Requesting

I'm now confident in the approach. I'll include a concise message confirming my understanding of the role, constraints, and request the problems. To build confidence and demonstrate my abilities, I will include a formatted derivation of the Sigmoid gradient, following the instructions precisely. My primary goal remains efficient communication to move forward with learning content.

Refining Response and Demonstrating

I'm now revising my response. Based on the constraints and my earlier thoughts, I'll directly ask for the study sheet. To demonstrate my abilities and adherence to the format, I'll include a formatted demonstration of the gradient of linear regression, showcasing all necessary aspects. My core focus is efficient communication and demonstrating my capabilities by efficiently working with an example. This approach is best. I am prepared.

Clarifying Role and Requirements

Now, I'm thinking the best approach is to confirm the role as an instructor and detail my understanding of the format requirements from the given instructions. It seems vital to restate the constraints to ensure the user understands that these will be followed. I'll reiterate my readiness and request the problems or topics to begin. This approach is the most efficient and clear to move forward. My focus now is ensuring they understand.

Creating a Demonstration Example

I'm now designing an illustrative example to highlight my capabilities. The focus is on clarity and adherence to the format specifications. I'm choosing the Backpropagation local gradient for Sigmoid activation due to its conceptual nature, well-suited to the non-coding emphasis. I'm confident this will effectively showcase my ability to provide insightful, step-by-step explanations within the prescribed framework. My goal is to use this as a proactive demonstration, inviting the user to provide their specific study content, while simultaneously exhibiting my readiness to dive in. I am excited to begin.

Constructing the Response

I'm now integrating a demonstration example into my response. I've chosen the backpropagation derivation for the sigmoid activation due to its conceptual nature and suitability for a non-coding emphasis. I've written the explanation using chain rule, and I'll include the intuition behind it. To meet the visual requirement, I will include an image tag referencing the sigmoid function. My goal is to demonstrate my ability to meet the requirements of the task.

I am ready to act as your expert instructor for CS182/EECS182. I have fully internalized the requirements: strict LaTeX formatting for math, step-by-step rigorous derivations, clear conceptual explanations, and deep intuitive connections to machine learning theory—with **no code**.

It appears you have not yet pasted the specific problems from your study sheet.

To demonstrate that I am aligned with your expectations, below is a **calibration example** of how I will answer your questions once you provide them.

Example Problem: Derivative of the Logistic Sigmoid Function

Question: Derive the derivative of the sigmoid activation function $\sigma(z) = \frac{1}{1+e^{-z}}$ with respect to its input z , and express the result in terms of $\sigma(z)$.

(1) Conceptual Explanation The sigmoid function is a smooth, differentiable non-linear activation function often used in binary classification and early neural networks. To perform backpropagation (gradient descent), we need its derivative to apply the Chain Rule. We are looking for the instantaneous rate of change of the activation output a with respect to the pre-activation input z .

(2) Step-by-step Derivation Let the sigmoid function be defined as:

$$\sigma(z) = (1 + e^{-z})^{-1}$$

We apply the derivative operator with respect to z , using the **Power Rule** and the **Chain Rule**:

1. **Apply Power Rule:** Treat $(1 + e^{-z})$ as the inner function u .

$$\frac{d}{dz} \sigma(z) = -1 \cdot (1 + e^{-z})^{-2} \cdot \frac{d}{dz} (1 + e^{-z})$$

2. **Differentiate the Inner Term:** The derivative of 1 is 0, and the derivative of e^{-z} is $-e^{-z}$.

$$\frac{d}{dz} (1 + e^{-z}) = 0 + (-e^{-z}) = -e^{-z}$$

3. **Substitute back:**

$$\begin{aligned} \frac{d}{dz} \sigma(z) &= - (1 + e^{-z})^{-2} \cdot (-e^{-z}) \\ &= \frac{e^{-z}}{(1 + e^{-z})^2} \end{aligned}$$

4. **Algebraic Manipulation to express in terms of $\sigma(z)$:** We can split the fraction:

$$\frac{e^{-z}}{(1 + e^{-z})^2} = \frac{1}{1 + e^{-z}} \cdot \frac{e^{-z}}{1 + e^{-z}}$$

We know the first term is $\sigma(z)$. Let's manipulate the second term by adding and subtracting 1 in the numerator:

$$\begin{aligned} \frac{e^{-z}}{1 + e^{-z}} &= \frac{(1 + e^{-z}) - 1}{1 + e^{-z}} = \frac{1 + e^{-z}}{1 + e^{-z}} - \frac{1}{1 + e^{-z}} \\ &= 1 - \sigma(z) \end{aligned}$$

5. **Combine terms:**

$$\frac{d}{dz} \sigma(z) = \sigma(z) \cdot (1 - \sigma(z))$$

(3) Final Answer Box

$$\frac{d}{dz} \sigma(z) = \sigma(z) (1 - \sigma(z))$$

(4) Intuition / Interpretation

- **Computational Efficiency:** The derivative can be computed using only the value of the function itself ($\sigma(z)$). This is computationally efficient during backpropagation because we already computed $\sigma(z)$ during the forward pass; we don't need to re-evaluate exponentials.
 - **Vanishing Gradient Problem:** Notice the shape of the derivative.
 - When $z = 0$, $\sigma(z) = 0.5$, so the gradient is $0.5(1 - 0.5) = 0.25$ (the maximum).
 - When $|z|$ is large (saturation), $\sigma(z) \approx 0$ or 1 . In both cases, the term $\sigma(z)(1 - \sigma(z))$ approaches **zero**.
 - This mathematical property causes the **vanishing gradient problem** in deep networks: if a neuron saturates, local gradients vanish, and weights stop updating. This motivates the use of non-saturating activations like ReLU.
-

Please paste the text of your study sheet below, and I will proceed with the rest of the problems.

I gave a pdf of my questions here.



[hw01_question - PDF](#)

Here are the questions



Thinking:

Reviewing the Scope

I'm currently focused on the prompt's core intention. I need to understand the role I'm playing, which is that of a knowledgeable instructor in machine learning. My objective is also very clear, which is to solve the non-coding questions for the user. I'm taking this in stride and assessing the best approach to deliver clear, helpful responses.

Assessing Problem Complexity

I've moved on to examining the specific problems outlined in the document. I'm noting the mathematical depth of Problem 2, with stochastic gradient descent and Lyapunov functions, as a key area. I'm also now carefully considering the structure of the provided output format, ensuring a clear and organized presentation of solutions for each problem.

Prioritizing Problem Approaches

I've just finished the problem set and am determining an effective solution path. Considering the user's desire to "complete" the sheet and the length of the problems, I've decided to prioritize problems 1-7. The goal is a detailed walkthrough of the key problems, with an emphasis on the most mathematically challenging, such as 2, 5, 6, and 7. I'll include problem 4, as it's algorithmically focused.

Implementing Rigorously & Clearly

I'm now integrating the core requirements into my process. I'm confirming that I'll include step derivations, LaTeX notation, symbol definitions, and intuitive explanations where helpful. The response will be completely code-free, and I will strictly adhere to



the output format with appropriate citations as necessary. My confidence in tackling the problem set is high, and I've developed a detailed approach.

Designing Full Solutions

I'm now diving into the specifics of each problem to develop the complete answer set. I'm prioritizing vertical space efficiency within the required format. I'm actively working to structure my explanations to align perfectly with the "Problem X" header format, ensuring seamless integration with the user's initial setup. I am now in the final analysis and implementation phase, ensuring all calculations, LaTeX formatting, and step derivations are accurate and clearly presented for each problem. I will also make sure to use notation from the problem when possible, as I'm creating the final versions of the solution set.

Correcting the Algorithm

I'm presently focused on correcting the algorithm descriptions, specifically lines 7 and 8 within the table, to ensure they accurately represent SGD with Momentum and Adam. I've pinpointed an OCR error that caused a crucial misalignment, and I'm currently using my knowledge of standard implementations to resolve the inconsistencies. I've re-examined the standard SGD with Momentum and Adam equations from external sources and the provided text for consistency.

Confirming Algorithm Details

I'm presently focused on correcting the algorithm descriptions within the table for Problem 4, specifically the missing elements for Adam. I have correctly identified the missing values for A and B. I'm confirming that the provided text is accurate, and I'll now complete the table to ensure the responses are mathematically precise and consistent with standard implementations. I'm now ready to proceed with the remaining parts of this problem.

Here are the solutions for the conceptual and mathematical portions of Homework 1.

Problem 1: Why Learning Rates Cannot be Too Big

(1) Conceptual Explanation This problem explores the stability conditions for Gradient Descent (GD). In optimization, if the learning rate (step size) η is too large, the updates can overshoot the minimum, causing the parameters to oscillate or diverge (explode) rather than converge. We analyze this using a simple linear recurrence relation derived from the squared loss function. We then extend this to the vector case, where different

directions (eigenvectors) have different curvatures (eigenvalues), and the stability is bottlenecked by the sharpest direction (largest eigenvalue).

(2) Step-by-step Derivation

(a) Stability Condition

The recurrence is given by:

$$w_{t+1} = (1 - 2\eta\sigma^2) w_t + 2\eta\sigma y$$

Let $r = 1 - 2\eta\sigma^2$. This is a linear dynamical system of the form $w_{t+1} = rw_t + C$. The convergence depends on the geometric ratio r . For the sequence to converge (be stable), the magnitude of the multiplier must be strictly less than 1 (inside the unit circle).

$$\begin{aligned} |1 - 2\eta\sigma^2| &< 1 \\ -1 &< 1 - 2\eta\sigma^2 < 1 \end{aligned}$$

Subtract 1 from all sides:

$$-2 < -2\eta\sigma^2 < 0$$

Divide by $-2\sigma^2$ (remembering to flip inequalities because we divide by a negative):

$$\begin{aligned} \frac{-2}{-2\sigma^2} &> \eta > \frac{0}{-2\sigma^2} \\ \frac{1}{\sigma^2} &> \eta > 0 \end{aligned}$$

(b) Convergence Speed The convergence rate is determined by the absolute value of the ratio $|r| = |1 - 2\eta\sigma^2|$. The distance to the optimal solution $w^* = y/\sigma$ decays as $|w_t - w^*| = |r|^t |w_0 - w^*|$. To reach a factor of $(1 - \epsilon)$ of the initial error, we typically look at the contraction. However, the question asks "if we wanted to get within a factor $(1 - \epsilon)$ of w^* ". Assuming this means the error ratio $\frac{|w_t - w^*|}{|w_0 - w^*|} \leq \epsilon'$ (interpreting "within a factor" loosely as achieving a specific reduction), the number of iterations t is governed by:

$$|1 - 2\eta\sigma^2|^t \approx \text{target}$$

Generally, the convergence is fastest when $1 - 2\eta\sigma^2 = 0$ (instant convergence), which happens at $\eta = 1/(2\sigma^2)$.

(c) Vector Case Stability We have two decoupled equations because the matrix is diagonal:

$$w[1]_{t+1} = (1 - 2\eta\sigma_{l2}) w[1]_t + \dots$$

$$w[2]_{t+1} = (1 - 2\eta\sigma_{s2}) w[2]_t + \dots$$

For the *entire* system to converge, **both** dimensions must be stable simultaneously.

Condition 1: $0 < \eta < \frac{1}{\sigma_{l2}}$ Condition 2: $0 < \eta < \frac{1}{\sigma_{s2}}$

Since $\sigma_l \gg \sigma_s$, we have $\frac{1}{\sigma_{l2}} \ll \frac{1}{\sigma_{s2}}$. The tighter constraint comes from the larger singular value σ_l . Therefore, stability requires $0 < \eta < \frac{1}{\sigma_{l2}}$. The limiting factor is σ_l (the sharpest direction).

(d) Convergence Speed Comparison Assuming a stable η (small enough for σ_l), the convergence rates are: Rate 1: $|1 - 2\eta\sigma_{l2}|$ Rate 2: $|1 - 2\eta\sigma_{s2}|$ Since η must be small (close to 0), $2\eta\sigma^2$ is small. The term $(1 - 2\eta\sigma_{l2})$ will be closer to 0 (faster convergence) than $(1 - 2\eta\sigma_{s2})$, which will be very close to 1 (slow convergence). **Faster dimension:** Dimension 1 (σ_l). **Slower dimension:** Dimension 2 (σ_s).

(e) Fastest Overall Convergence The overall speed is dominated by the slowest dimension. We need to minimize the maximum of the two rates:

$\max(|1 - 2\eta\sigma_{l2}|, |1 - 2\eta\sigma_{s2}|)$. The optimal η balances the oscillating convergence of the sharp direction and the monotonic convergence of the flat direction:

$$1 - 2\eta\sigma_{s2} = -(1 - 2\eta\sigma_{l2})$$

$$1 - 2\eta\sigma_{s2} = -1 + 2\eta\sigma_{l2}$$

$$2 = 2\eta(\sigma_{l2} + \sigma_{s2}) \implies \eta_{opt} = \frac{1}{\sigma_{l2} + \sigma_{s2}}$$

(f) Multiple Parallel Problems If there were intermediate σ_i such that $\sigma_s < \sigma_i < \sigma_l$, they would not affect the choice of η . The stability is bounded by the max curvature ($\sigma_{max} = \sigma_l$) and the convergence speed is bottlenecked by the min curvature ($\sigma_{min} = \sigma_s$) and max curvature. The condition numbers depend only on the extremes.

(g) SVD Relevance For a general problem $Xw \approx y$, the Singular Value Decomposition (SVD) $X = U\Sigma V^T$ diagonalizes the problem. Rotating into the eigenbasis $V^T w$, the dynamics decouple exactly into independent scalar problems corresponding to the singular values σ_i of X . Thus, the analysis of σ_l vs σ_s corresponds exactly to the condition number $\kappa(X) = \sigma_{max}/\sigma_{min}$ determining GD performance on general matrices.

(3) Final Answer Box (a) **Stability:** $0 < \eta < \frac{1}{\sigma^2}$ (c) **Vector Stability:** $0 < \eta < \frac{1}{\sigma_{l2}}$. Limiting factor is σ_l . (e) **Optimal Rate:** $\eta = \frac{1}{\sigma_{l2} + \sigma_{s2}}$

(4) Intuition / Interpretation

- **Valley Analogy:** Think of the loss function as a valley. σ_l corresponds to the steep sides, and σ_s corresponds to the flat valley floor.
 - **Step Size Dilemma:** If you take large steps to move fast along the flat floor (σ_s), you will crash into the steep walls (σ_l) and diverge. You are forced to take tiny steps to accommodate the steep walls, which makes progress along the valley floor agonizingly slow. This is why "ill-conditioned" problems (large ratio σ_l/σ_s) are hard for standard Gradient Descent.
-

Problem 3: Accelerating Gradient Descent with Momentum

(1) Conceptual Explanation Momentum accumulates a "velocity" vector of past gradients, allowing the optimizer to smooth out oscillations in high-curvature directions and build up speed in flat directions. We analyze the stability of momentum by treating the update rule as a linear dynamical system and examining the eigenvalues of the iteration matrix.

(2) Step-by-step Derivation

(a) Reparameterization We substitute $x_t = V^T(w_t - w^*)$ and $a_t = V^T z_t$ into the momentum update. Original update:

$$\begin{aligned} w_{t+1} &= w_t - \eta z_{t+1} \\ z_{t+1} &= (1 - \beta) z_t + \beta (2X^T X (w_t - w^*)) \end{aligned}$$

(Note: $\nabla L(w) = 2X^T X w - 2X^T y = 2X^T X (w - w^*)$).

Left-multiply by V^T :

$$\begin{aligned} V^T w_{t+1} &= V^T w_t - \eta V^T z_{t+1} \implies x_{t+1} + V^T w^* = x_t + V^T w^* - \eta a_{t+1} \implies x_{t+1} = x_t - \eta a_{t+1} \\ a_{t+1} &= (1 - \beta) a_t + \beta (2V^T X^T X V V^T (w_t - w^*)) = (1 - \beta) a_t + 2\beta \Sigma^2 x_t \end{aligned}$$

For component i with singular value σ_i :

$$\begin{aligned} x_{t+1}[i] &= x_t[i] - \eta a_{t+1}[i] \\ a_{t+1}[i] &= (1 - \beta) a_t[i] + 2\beta \sigma_{i2} x_t[i] \end{aligned}$$

Substituting a_{t+1} into the x equation:

$$x_{t+1}[i] = x_t[i] - \eta((1-\beta)a_t[i] + 2\beta\sigma_{i2}x_t[i])$$

$$x_{t+1}[i] = (1 - 2\eta\beta\sigma_{i2})x_t[i] - \eta(1-\beta)a_t[i]$$

**(b) System Matrix R_i ** Writing this in matrix form for the state vector $[a_{t+1}[i] \ x_{t+1}[i]]$:

$$[a_{t+1}x_{t+1}] = [1 - \beta - \eta(1-\beta) \ 2\beta\sigma_{i2} \ 1 - 2\eta\beta\sigma_{i2}] [a_t x_t]$$

Thus, $R_i = [1 - \beta - \eta(1-\beta) \ 2\beta\sigma_{i2} \ 1 - 2\eta\beta\sigma_{i2}]$.

**(c) Eigenvalues of R_i ** The eigenvalues λ are roots of the characteristic equation $\det(R_i - \lambda I) = 0$. Trace $T = (1-\beta) + (1 - 2\eta\beta\sigma_{i2}) = 2 - \beta - 2\eta\beta\sigma_{i2}$. Determinant $D = (1-\beta)(1 - 2\eta\beta\sigma_{i2}) - (-\eta(1-\beta)(2\beta\sigma_{i2})) = 1 - \beta - 2\eta\beta\sigma_{i2} + 2\eta\beta^2\sigma_{i2} + 2\eta\beta\sigma_{i2} - 2\eta\beta^2\sigma_{i2}$. Simplifying the determinant: $D = (1-\beta)(1 - 2\eta\beta\sigma_{i2}) + 2\eta\beta(1-\beta)\sigma_{i2} = (1-\beta)[1 - 2\eta\beta\sigma_{i2} + 2\eta\beta\sigma_{i2}] = 1 - \beta$. This is a remarkable result:
The determinant is always $1 - \beta$.

The eigenvalues are:

$$\lambda = \frac{T \pm \sqrt{T^2 - 4D}}{2}$$

Discriminant $\Delta = T^2 - 4(1-\beta)$.

- **Real:** $\Delta > 0$.
- **Repeated:** $\Delta = 0$.
- **Complex:** $\Delta < 0$.

(d) Repeated Eigenvalues Condition: $T^2 - 4(1-\beta) = 0 \implies T = \pm 2\sqrt{1-\beta}$.

Substituting $T = 2 - \beta - 2\eta\beta\sigma_{i2}$: This defines the critical damping boundary. The highest learning rate resulting in repeated roots corresponds to the transition point.

(g) Application ($\beta = 0.1, \sigma_{max2} = 5, \sigma_{min2} = 0.05$) We want to choose η to optimize convergence. Without momentum ($\beta = 1$), $\eta_{opt} = \frac{1}{5+0.05} \approx 0.198$. Convergence rate $\rho \approx \frac{5-0.05}{5+0.05} \approx 0.98$. With momentum, we want the eigenvalues for both extremes to be small. Ideally, we set parameters such that the worst-case eigenvalues are complex (underdamped) but with small magnitude $\sqrt{1-\beta}$. Optimal step size with momentum often scales as $\eta \approx \frac{1}{\sigma_{max2}}$. However, the standard optimal parameters for Polyak momentum are: $\beta = \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^2$ and $\eta = \frac{4}{(\sqrt{\sigma_{max2}} + \sqrt{\sigma_{min2}})^2}$. Here, we are given fixed $\beta = 0.1$. We need to find the best η . This usually occurs when the discriminant is zero for the smallest eigenvalue or the system is critically damped.

(3) Final Answer Box (b) $R_i = [1 - \beta - \eta(1 - \beta)2\beta\sigma_{i2}1 - 2\eta\beta\sigma_{i2}]$ (c) Eigenvalues are $\lambda = \frac{\text{Tr}(R_i) \pm \sqrt{\text{Tr}(R_i)^2 - 4(1-\beta)}}{2}$.

- Complex if $(2 - \beta - 2\eta\beta\sigma_{i2})^2 < 4(1 - \beta)$.
- Real otherwise.

(4) Intuition / Interpretation

- **Determinant as Damping:** Since the determinant is the product of eigenvalues $\lambda_1\lambda_2 = 1 - \beta$, the magnitude of the eigenvalues is roughly $\sqrt{1 - \beta}$ when they are complex conjugates. This means convergence is dictated by β directly.
 - **Acceleration:** By allowing complex eigenvalues, momentum allows the trajectory to spiral towards the optimum (underdamped harmonic oscillator). It doesn't need to monotonically descend the valley; it can swing past the bottom and swing back, effectively "averaging" out the error faster than standard GD can crawl down the slope.
-

Problem 4: Optimizers

(1) Conceptual Explanation This problem covers the implementation details of the Adam optimizer and the theoretical equivalence between L2 regularization (penalty on loss) and Weight Decay (penalty on update).

(2) Step-by-step Derivation

(a) Adam Pseudocode Completion Adam maintains exponential moving averages of the gradients (m_t) and the squared gradients (v_t). Based on standard Adam formulation: (A) First Moment: Update biased first moment estimate.

$$m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

(B) Second Moment: Update biased raw second moment estimate.

$$v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2) g_{t2}$$

(where g_{t2} is element-wise).

(b) Weight Decay vs L2 Regularization L2 Regularization Update: Objective: $J(\theta) = f(\theta) + \frac{\lambda}{2} \|\theta\|^2$. Gradient: $\nabla J(\theta) = \nabla f(\theta) + \lambda\theta$. SGD Update:

$$\theta_{t+1} = \theta_t - \eta (\nabla f(\theta_t) + \lambda \theta_t)$$

$$\theta_{t+1} = \theta_t - \eta \nabla f(\theta_t) - \eta \lambda \theta_t$$

$$\theta_{t+1} = (1 - \eta \lambda) \theta_t - \eta \nabla f(\theta_t)$$

Weight Decay Update: Given in prompt:

$$\theta_{t+1} = (1 - \gamma) \theta_t - \eta \nabla f(\theta_t)$$

Equivalence: Comparing the coefficients of θ_t :

$$1 - \eta \lambda = 1 - \gamma$$

$$\gamma = \eta \lambda$$

(3) Final Answer Box (a) (A): $\beta_1 m_{t-1} + (1 - \beta_1) g_t$ (B): $\beta_2 v_{t-1} + (1 - \beta_2) (g_t \odot g_t)$ (b)
Equivalence holds when $\gamma = \eta \lambda$.

(4) Intuition / Interpretation

- **Coupled parameters:** In standard SGD, L2 regularization and weight decay are mathematically identical. However, in adaptive methods like Adam (not shown in the derivation but worth noting for context), they differ because Adam scales the gradient part $\eta \nabla f$ by $1/\sqrt{v_t}$, but weight decay is typically applied directly to the weights.
- **Mechanism:** Both methods pull the weights toward zero. L2 Reg does it by adding a "slope" to the loss landscape pointing to the origin. Weight Decay does it by shrinking the weight vector by a constant factor $(1 - \gamma)$ at every step.

Problem 5: Regularization and Instance Noise

(1) Conceptual Explanation We show that training on data corrupted by additive noise is equivalent to training on clean data with L2 regularization (Ridge Regression). This explains why noise injection (like Dropout or jittering inputs) acts as a regularizer to prevent overfitting.

(2) Step-by-step Derivation

(a) Equivalence to Regularization We want to minimize $L(w) = E \left[\| \tilde{X}w - y \|^2 \right]$ where $\tilde{X} = X + N$. Let's look at the i -th term (row): $E \left[\| (x_i + n_i)^T w - y_i \|^2 \right]$. Expand the square:

$$\begin{aligned}
&= E \left[((x_{iT} w - y_i) + n_{iT} w)^2 \right] \\
&= E \left[(x_{iT} w - y_i)^2 + 2(x_{iT} w - y_i)(n_{iT} w) + (n_{iT} w)^2 \right]
\end{aligned}$$

Linearity of Expectation:

1. **First term:** Constant wrt noise. $(x_{iT} w - y_i)^2$.
2. **Second term:** $E[n_i] = 0$, so $E[2(\dots)n_{iT}w] = 2(\dots)E[n_i]^T w = 0$.
3. **Third term:** $E[(w^T n_i)(n_{iT} w)] = E[w^T (n_i n_{iT}) w] = w^T E[n_i n_{iT}] w$. Since $n_i \sim N(0, \sigma^2 I)$, the covariance $E[n_i n_{iT}] = \sigma^2 I$. So, term is $w^T (\sigma^2 I) w = \sigma^2 \|w\|^2$.

Summing over m samples and averaging (factor $1/m$ implied by prompt notation in Eq 29):

$$\begin{aligned}
L(w) &= \frac{1}{m} \sum_{i=1}^m (x_{iT} w - y_i)^2 + \frac{1}{m} \sum_{i=1}^m \sigma^2 \|w\|^2 \\
&= \frac{1}{m} \|Xw - y\|^2 + \sigma^2 \|w\|^2
\end{aligned}$$

Compare to Eq 29: $\frac{1}{m} \|Xw - y\|^2 + \lambda \|w\|^2$. We identify $\lambda = \sigma^2$.

(b) Gradient Descent Expectation Loss on instance t : $L_t(w) = \frac{1}{2}(\tilde{x}_t w - y)^2 = \frac{1}{2}((x + n_t)w - y)^2$. Gradient: $\nabla_w L_t = ((x + n_t)w - y)(x + n_t)$. Update: $w_{t+1} = w_t - \eta((x + n_t)w_t - y)(x + n_t)$. Assume w_t is independent of n_t (current noise). Take expectation E wrt n_t :

$$E[w_{t+1} | w_t] = w_t - \eta E[((xw_t - y) + n_tw_t)(x + n_t)]$$

Expand the product inside expectation: $((xw_t - y) + n_tw_t)(x + n_t) = (xw_t - y)x + (xw_t - y)n_t + (n_tw_t)x + (n_tw_t)n_t$. Expectations:

1. $(xw_t - y)x$: Constant.
2. $(xw_t - y)n_t$: Zero (mean 0).
3. $(n_tw_t)x = (w_t n_t)x$: Zero.
4. $(n_tw_t)n_t = w_t n_{t2}$: $E[n_{t2}] = \sigma^2$. So $\sigma^2 w_t$.

Result:

$$\begin{aligned}
E[w_{t+1}] &= E[w_t] - \eta ((x^2 E[w_t] - xy) + \sigma^2 E[w_t]) \\
E[w_{t+1}] &= (1 - \eta(x^2 + \sigma^2)) E[w_t] + \eta xy
\end{aligned}$$

(c) Convergence Condition This is the same stability form as Problem 1. | $1 - \eta(x^2 + \sigma^2) < 1$. $0 < \eta < \frac{2}{x^2 + \sigma^2}$.

(d) Converged Value At convergence, $E[w_{t+1}] = E[w_t] = w_\infty$. $w_\infty = (1 - \eta(x^2 + \sigma^2))w_\infty + \eta xy$

$$w_\infty = \frac{xy}{x^2 + \sigma^2}$$

Comparison to optimal (no noise): Optimal OLS $w^* = (X^T X)^{-1} X^T y = \frac{xy}{x^2} = \frac{y}{x}$. The noise shrinks the solution toward 0 (denominator is larger). This is exactly the Ridge Regression solution $w_{\text{ridge}} = \frac{xy}{x^2 + \lambda}$ with $\lambda = \sigma^2$.

(3) Final Answer Box (a) $\lambda = \sigma^2$. (b) $E[w_{t+1}] = (1 - \eta(x^2 + \sigma^2))E[w_t] + \eta xy$. (c) $0 < \eta < \frac{2}{x^2 + \sigma^2}$. (d) Converges to $\frac{xy}{x^2 + \sigma^2}$. This is smaller (shrunk) compared to unregularized $\frac{y}{x}$.

(4) Intuition / Interpretation

- **Noise = Penalty:** Adding noise to the input x makes the effective input "larger" in the variance sense ($x^2 + \sigma^2$), which dominates the denominator of the solution.
 - **Robustness:** The model learns not to trust the specific value of x too much because it's unreliable (noisy). It becomes conservative, preferring smaller weights, which prevents overfitting to specific training points.
-

Problem 6: General Case Tikhonov Regularization

(1) Conceptual Explanation Tikhonov Regularization is a generalized form of Ridge Regression where we penalize not just the magnitude of the weights ($\|x\|^2$), but their deviation from a prior c ($\|x - c\|^2$) and potentially weight different residuals differently (W_1).

(2) Step-by-step Derivation

(a) Manual Matrix Derivation Objective: $J(x) = (W_1(Ax - b))^T (W_1(Ax - b)) + (W_2(x - c))^T (W_2(x - c))$. Expand: $J(x) = (Ax - b)^T W_{1T} W_1 (Ax - b) + (x - c)^T W_{2T} W_2 (x - c)$. Gradient $\nabla_x J(x) = 0$: Using $\nabla_x (Ax - b)^T M (Ax - b) = 2A^T M (Ax - b)$:

$$2A^T W_{1T} W_1 (Ax - b) + 2W_{2T} W_2 (x - c) = 0$$

Drop factor 2 and distribute:

$$A^T W_{1T} W_1 A x - A^T W_{1T} W_1 b + W_{2T} W_2 x - W_{2T} W_2 c = 0$$

Group x terms:

$$(A^T W_{1T} W_1 A + W_{2T} W_2) x = A^T W_{1T} W_1 b + W_{2T} W_2 c$$

Solution:

$$x = (A^T W_{1T} W_1 A + W_{2T} W_2)^{-1} (A^T W_{1T} W_1 b + W_{2T} W_2 c)$$

(b) OLS Form Construction We want $\min \|Cx - d\|^2$. Observe the structure $(A^T \dots A + W^T \dots W)x$. This looks like a stacked matrix product. Let's stack the linear constraints:

1. $W_1 A x \approx W_1 b$
2. $W_2 x \approx W_2 c$

Define stacked matrix C and vector d :

$$C = [W_1 A W_2], d = [W_1 b W_2 c]$$

Then $Cx - d = [W_1 (Ax - b) W_2 (x - c)]$. Norm squared: $\|Cx - d\|^2 = \|W_1 (Ax - b)\|^2 + \|W_2 (x - c)\|^2$, which matches the objective. OLS Solution: $x = (C^T C)^{-1} C^T d$.
 $C^T C = [(W_1 A)^T W_2 T] [W_1 A W_2] = A^T W_{1T} W_1 A + W_{2T} W_2$. $C^T d = [(W_1 A)^T W_2 T] [W_1 b W_2 c] = A^T W_{1T} W_1 b + W_{2T} W_2 c$. This confirms the answer in (a).

(c) Reduction to Ridge Regression Ridge Regression: $\min \|Ax - b\|^2 + \lambda \|x\|^2$. We need:

1. $W_1 = I$ (Identity, standard least squares).
2. $c = 0$ (Prior is zero mean).
3. $W_{2T} W_2 = \lambda I$. This implies $W_2 = \sqrt{\lambda} I$.

(3) Final Answer Box (a) $x^* = (A^T W_{1T} W_1 A + W_{2T} W_2)^{-1} (A^T W_{1T} W_1 b + W_{2T} W_2 c)$ (b)
 $C = [W_1 A W_2], d = [W_1 b W_2 c]$ (c) $W_1 = I, c = 0, W_2 = \sqrt{\lambda} I$.

(4) Intuition / Interpretation

- **Virtual Data Points:** Regularization can always be interpreted as adding "virtual" data points to your training set. Tikhonov adds virtual data points saying "x should be

close to c with confidence W_2 ".

- **Bayesian Prior:** W_2 represents the inverse covariance (precision) of the Gaussian prior on the weights. W_1 represents the inverse covariance of the noise in the measurements.
-

Problem 7: Alternate MAP Interpretation of Ridge Regression

(1) Conceptual Explanation This problem connects the "Primal" (weight space) and "Dual" (sample space/kernel) forms of Ridge Regression using Gaussian probability properties. We prove they are identical by treating the output Y and weights W as jointly Gaussian random variables and computing the conditional expectation (MAP estimate).

(2) Step-by-step Derivation

We want to calculate $E[W | Y = y]$ using formula (32):

$$E[W | Y = y] = \Sigma_{WY} \Sigma_{YY^{-1}} y$$

Step 1: Define the model $W \sim N(0, I_d)$ (Prior) $N \sim N(0, I_n)$ (Noise) $Y = XW + \sqrt{\lambda}N$

**Step 2: Compute Covariance Σ_{YY} ** $\Sigma_{YY} = E[YY^T] = E\left[\left(XW + \sqrt{\lambda}N\right)\left(XW + \sqrt{\lambda}N\right)^T\right]$ Since W and N are independent, cross terms are 0. $= E[XWW^TX^T] + E[\lambda NN^T] = XE[WW^T]X^T + \lambda E[NN^T]$ Since $E[WW^T] = I$ and $E[NN^T] = I$:

$$\Sigma_{YY} = XX^T + \lambda I$$

**Step 3: Compute Cross-Covariance Σ_{WY} ** $\Sigma_{WY} = E[WY^T] = E\left[W\left(XW + \sqrt{\lambda}N\right)^T\right] = E[WW^TX^T] + E\left[WN^T\sqrt{\lambda}\right]$ The second term is 0 (independence). $= E[WW^T]X^T = IX^T = X^T$.

Step 4: Combine into Conditional Expectation Substitute into the formula:

$$E[W | Y = y] = X^T (XX^T + \lambda I)^{-1} y$$

This matches Eq (31) (The Kernel/Dual form).



Step 5: Connection to Primal Form To show this equals the primal form $(X^T X + \lambda I)^{-1} X^T y$, we use the **Matrix Inversion Lemma** (Woodbury Identity) or a simple algebraic identity: Identity: $X^T (XX^T + \lambda I)^{-1} = (X^T X + \lambda I)^{-1} X^T$. *Proof of Identity:* Start with: $(X^T X + \lambda I) X^T = X^T X X^T + \lambda X^T = X^T (XX^T + \lambda I)$. Multiply left by $(X^T X + \lambda I)^{-1}$ and right by $(XX^T + \lambda I)^{-1}$: $X^T (XX^T + \lambda I)^{-1} = (X^T X + \lambda I)^{-1} X^T$.

Thus, the calculated MAP estimate matches both forms.

(3) Final Answer Box $\hat{w}_{MAP} = X^T (XX^T + \lambda I)^{-1} y$ Using matrix identity, this equals $(X^T X + \lambda I)^{-1} X^T y$.

(4) Intuition / Interpretation

- **Dual View:** The term $(XX^T + \lambda I)^{-1} y$ effectively computes "dual weights" α for each data point. The weight vector is then a linear combination of the data points: $w = X^T \alpha = \sum \alpha_i x_i$. This is the foundation of Kernel methods (SVMs, Gaussian Processes).
- **Computation:**
 - Primal form involves inverting a $d \times d$ matrix ($X^T X$). Good when $n \gg d$ (lots of data, few features).
 - Dual form involves inverting an $n \times n$ matrix (XX^T). Good when $d \gg n$ (lots of features, few data points).

These answers were great. Some were correct, but others were nonsense. I decided to ask more granularly.



Using your reasonings from above, what is your answer for problem 1a? Just the answer please



Thinking:

Revisiting Problem 1a

I'm revisiting the analysis of Problem 1a, ensuring I deliver a precise answer aligned with the homework file. I want to avoid any ambiguity, so I'm meticulously checking my previous derivation against the document's content. My goal is to guarantee absolute accuracy in the response.

Determining Stability Condition

I've just derived the condition for stability based on the recurrence relation. The crucial step was ensuring the coefficient of the weight term remained within bounds. My analysis culminated in the inequality $0 < \eta < \frac{1}{\sigma^2}$. I'll present this concise result, referencing the supporting reasoning as instructed.

Problem 1(a) Answer

Based on the stability condition for the linear recurrence relation $w_{t+1} = (1 - 2\eta\sigma^2) w_t + 2\eta\sigma y$, the magnitude of the scaling factor must be less than 1 ($|1 - 2\eta\sigma^2| < 1$).

The range of learning rates η for which the recurrence is stable is:

$$0 < \eta < \frac{1}{\sigma^2}$$

Good!





What is your answer for 1b?



Good!

The answer for 1(b) is derived from solving for the number of iterations t needed to reduce the error to a fraction $(1 - \epsilon)$ of the initial error.

Problem 1(b) Answer

To reach a solution w_t such that the error is reduced by a factor of $(1 - \epsilon)$, the number of iterations t required is:

$$t \approx \frac{\ln(\epsilon)}{\ln |1 - 2\eta\sigma^2|}$$

(1) Conceptual Explanation The convergence of the recurrence $w_{t+1} - w^* = (1 - 2\eta\sigma^2)(w_t - w^*)$ is geometric. The error at step t is given by $|w_t - w^*|$ [cite_start] $= |1 - 2\eta\sigma^2|^t |w_0 - w^*|$. We want the ratio of current error to initial error to be less than or equal to $(1 - \epsilon)$ (or simply ϵ depending on if we interpret the prompt "within a factor $(1 - \epsilon)$ " as retaining $(1 - \epsilon)$ of the error or removing $(1 - \epsilon)$). Assuming the standard interpretation that we want the error to be ϵ times the initial error (interpreting "within a factor" as a reduction target): $|1 - 2\eta\sigma^2|^t \leq \epsilon$. Taking logs gives the result.

(2) Step-by-step Derivation

1. Define the error at time t as $e_t = |w_t - w^*|$.
2. From the recurrence relation: $e_t = |1 - 2\eta\sigma^2| e_{t-1}$.
3. Unrolling this over t steps: $e_t = |1 - 2\eta\sigma^2|^t e_0$.
4. We want to be within a factor of $(1 - \epsilon)$ of the optimal w^* . However, usually "factor $(1 - \epsilon)$ " in convergence contexts means the error is reduced *to* ϵ relative to the start (or reduced *by* a factor). If the prompt means $w_t \approx (1 - \epsilon) w^*$, then relative error is ϵ .
5. Set the relative error condition: $|1 - 2\eta\sigma^2|^t \approx \epsilon$ (assuming small ϵ target).
6. Take the natural logarithm of both sides:

$$t \ln |1 - 2\eta\sigma^2| \approx \ln(\epsilon)$$

7. Solve for t :

$$t \approx \frac{\ln(\epsilon)}{\ln |1 - 2\eta\sigma^2|}$$

(3) Intuition

- **Decay Rate:** The term $\ln |1 - 2\eta\sigma^2|$ is negative (since the argument is < 1). Dividing by it yields a positive time t .
- **Step Size Impact:** A larger step size η (up to the stability limit) makes the denominator more negative (larger magnitude), resulting in a smaller t (fewer iterations).
- **Curvature Impact:** A larger σ (steeper curvature) also increases the magnitude of the update, reducing iteration count, provided η is stable.

(4) Note on Interpretation The prompt phrasing "get within a factor $(1 - \epsilon)$ of w^* " usually implies $w_t \geq (1 - \epsilon) w^*$. If $w^* > 0$ and $w_0 = 0$, the error starts at w^* and ends at ϵw^* . Thus the ratio of final error to initial error is $\frac{\epsilon w^*}{w^*} = \epsilon$. If the question strictly means "reduce error by factor $(1 - \epsilon)$ ", then the target ratio is $(1 - (1 - \epsilon)) = \epsilon$. If the question implies the *ratio* itself is $(1 - \epsilon)$, then replace $\ln(\epsilon)$ with $\ln(1 - \epsilon)$ in the answer. Given the context of convergence "within ϵ ", ϵ is usually the small residual error.



Now give me just the answers for the rest of problem 1 please



Thinking:

Confirming the Solutions

I've completed my analysis of the user's request. I'm ready to supply the responses for problem 1 parts c through g, leveraging the uploaded document and all previously established logic. My current focus is solely on delivering the required solutions.

Analyzing User's Intent

I'm considering the user's shift from detailed explanations to just answers. While the initial request emphasized a structured format with derivations and explanations, the current prompt seeks only answers. To best serve as a helpful instructor, I am adapting the approach. I now lean toward providing concise derivations, final boxed answers, and a brief interpretation. The provided PDF text, `hw01_question.pdf`, remains the primary source.

Investigating Convergence Conditions

I've tackled the convergence rates for each dimension, deriving the stability conditions and comparing their speeds. I've concluded the dimension with σ_l converges slower due to the smaller rate. I have also identified the learning rate limitation imposed by σ_l within the bounds I've calculated.

Clarifying Convergence Dynamics

I'm now revising my analysis of the convergence speeds. Initially, I incorrectly assumed σ_l converged slower. Careful re-evaluation, considering the impact of a very small learning rate (η), revealed that $w[1]$ (corresponding to σ_l) actually converges faster than $w[2]$ (corresponding to σ_s) because the error multiplier is smaller.

Calculating Optimal Learning Rate

I've determined the optimal learning rate. I arrived at the answer of $\eta = \frac{1}{\sigma_{l2} + \sigma_{s2}}$ by recognizing the condition for the most efficient convergence. This occurs when the error terms for each dimension have equal magnitude but opposite signs, leading to the simple result. I also analyzed the implications of intermediate singular values, which don't alter this.

Evaluating Singular Values' Impact

I've just assessed the relevance of the Singular Value Decomposition (SVD) within the context of general least squares problems, $Xw \approx y$. Applying the SVD, $X = U\Sigma V^T$, simplifies the problem into independent scalar problems. The convergence is inherently constrained by the condition number, specifically by the ratio $\sigma_{max}/\sigma_{min}$.

Confirming Calculations & Answers

I've just finished the final check for derivation 1(e). The calculations hold, confirming the rates and the derivation of the learning rate. I've double-checked my work for 1(f) too, and my conclusion holds. I'm now drafting the complete response.

Here are the answers for the remainder of Problem 1.

Problem 1(c)

(1) Step-by-step Derivation We have two decoupled scalar equations with singular values σ_l (large) and σ_s (small). For the vector system to converge, the learning rate η must satisfy the stability condition for **both** dimensions simultaneously:

1. Dimension 1: $0 < \eta < \frac{1}{\sigma_{l2}}$
2. Dimension 2: $0 < \eta < \frac{1}{\sigma_{s2}}$

Since $\sigma_l \gg \sigma_s$, we have $\frac{1}{\sigma_{l2}} \ll \frac{1}{\sigma_{s2}}$. The condition imposed by the larger singular value σ_l is stricter and subsumes the other.

(2) Final Answer Box

$$0 < \eta < \frac{1}{\sigma_{l2}}$$

Limiting factor: The larger singular value σ_l limits our learning rate.



Problem 1(d)

(1) Step-by-step Derivation The convergence rate is determined by the factor $|1 - 2\eta\sigma^2|$. Assuming we choose a small, stable learning rate (valid for the σ_l limit):

- **Dimension 1 (σ_l):** The update term $2\eta\sigma_{l2}$ is larger, meaning the coefficient $(1 - 2\eta\sigma_{l2})$ is further from 1 and closer to 0 (or even negative, but with smaller magnitude than 1). This dimension changes rapidly.
- **Dimension 2 (σ_s):** The update term $2\eta\sigma_{s2}$ is very tiny. The coefficient $(1 - 2\eta\sigma_{s2})$ is very close to 1 (e.g., 0.999). This dimension decays very slowly.

(2) Final Answer Box Faster Convergence: The dimension corresponding to σ_l (large singular value). **Slower Convergence:** The dimension corresponding to σ_s (small singular value).

Problem 1(e)

(1) Step-by-step Derivation The overall convergence speed is bottlenecked by the slowest dimension (the maximum of the two rates). We want to find η that minimizes:

$$\max(|1 - 2\eta\sigma_{l2}|, |1 - 2\eta\sigma_{s2}|)$$

The optimum occurs when the "flat" direction (positive, monotonic convergence) and the "sharp" direction (negative, oscillating convergence) decay at the same rate magnitude:

$$1 - 2\eta\sigma_{s2} = -(1 - 2\eta\sigma_{l2})$$

$$1 - 2\eta\sigma_{s2} = -1 + 2\eta\sigma_{l2}$$

$$2 = 2\eta(\sigma_{l2} + \sigma_{s2})$$

$$\eta = \frac{1}{\sigma_{l2} + \sigma_{s2}}$$

(2) Final Answer Box

$$\eta_{opt} = \frac{1}{\sigma_{l2} + \sigma_{s2}}$$

Problem 1(f)

(1) Explanation The convergence of the system is determined by the "worst-case" directions.

- **Stability** is determined exclusively by the sharpest direction (maximum σ), which risks diverging.
- **Convergence Speed** is determined by the gap between the slowest decay (smallest σ) and the oscillation limit (largest σ). Intermediate values σ_i (where $\sigma_s < \sigma_i < \sigma_l$) have convergence rates strictly better than the worst-case boundaries bounded by σ_l and σ_s .

(2) Final Answer Box No, additional parallel problems with intermediate σ_i would **not** influence the choice of possible learning rates or the optimal rate. The constraints are set entirely by the minimum (σ_s) and maximum (σ_l) singular values.

Problem 1(g)

(1) Conceptual Explanation In general Least Squares problems ($Xw \approx y$), we can perform a Singular Value Decomposition (SVD) on $X = U\Sigma V^\top$. By rotating the coordinate system to the eigenbasis V (setting $\tilde{w} = V^\top w$), the coupled matrix equation decouples into independent scalar equations exactly like the ones analyzed in parts (a)-(f). The diagonal entries of the matrix Σ correspond to the σ_i values.

(2) Final Answer Box The analysis is directly relevant because **SVD diagonalizes the general least-squares problem**. The behavior of Gradient Descent on a general matrix X is identical to running parallel scalar GD updates on the singular values σ_i .

- σ_{max} determines the stability limit ($\eta < 2/\sigma_{max}$).
- The condition number $\sigma_{max}/\sigma_{min}$ determines the convergence speed.

All tools good!



Let's go to problem 2 now please



Problem 2: Stochastic Gradient Descent (when it is possible to interpolate)

(1) Conceptual Explanation This problem analyzes the convergence of Stochastic Gradient Descent (SGD) when a perfect solution exists (interpolation regime). We use a series of coordinate changes—centering around the minimum-norm solution (w^*) and applying SVD—to transform the original wide matrix problem $Xw = y$ into an equivalent, simplified square system $\tilde{X}\tilde{w} = 0$. The analysis then uses the concept of a **stochastic Lyapunov function** $L(\tilde{w})$ to prove that SGD, with a constant learning rate η , converges exponentially in expectation to the optimal solution $\tilde{w} = 0$.

(2) Step-by-step Derivation

(a) Reparameterization and Initial Condition

The original problem is $Xw = y$. Since $X \in \mathbb{R}^{n \times d}$ has full row-rank and $d > n$, the minimum-norm solution w^* (the solution in the row space of X) is given by the pseudo-inverse:

$$w^* = X^T (XX^T)^{-1} y$$

The new coordinate $w' = w - w^*$ transforms the problem:

$$Xw = y \implies X(w' + w^*) = y \implies Xw' + Xw^* = y$$

Since w^* is a solution, $Xw^* = y$.

$$Xw' = 0$$

The initial condition is $w_0 = 0$. The new initial condition $w_{0'}$ is:

$$w_{0'} = w_0 - w^* = 0 - w^* = -X^T (XX^T)^{-1} y$$

(b) SVD Coordinates and Initial Condition

We use the SVD of $X = U\Sigma V^T$. The transformation $w'' = V^T w'$ aligns the coordinates with the row space (first n dimensions) and the null space (remaining $d - n$ dimensions). Substituting $w' = Vw''$ into $Xw' = 0$:

$$U\Sigma V^T Vw'' = 0 \implies \Sigma w'' = 0$$

Since $\Sigma = [\tilde{\Sigma} 0_{n \times (d-n)}]$ where $\tilde{\Sigma} \in R^{n \times n}$ contains the non-zero singular values, the equation becomes:

$$[\tilde{\Sigma} 0_{n \times (d-n)}] w'' = 0$$

This implies $\tilde{\Sigma} w''[1..n] = 0$.

For the initial condition $w_{0\prime}$:

$$w_{0\prime\prime} = V^T w_{0\prime} = -V^T w^*$$

Substituting $w^* = X^T (XX^T)^{-1} y = V\Sigma^T (\Sigma\Sigma^T)^{-1} U^T y$:

$$w_{0\prime\prime} = -V^T V\Sigma^T (\Sigma\Sigma^T)^{-1} U^T y = -I\Sigma^T (\Sigma\Sigma^T)^{-1} U^T y$$

Using the structure of Σ : $\Sigma^T (\Sigma\Sigma^T)^{-1} = [\tilde{\Sigma} 0_{(d-n) \times n}] (\tilde{\Sigma}^2)^{-1} = [\tilde{\Sigma}^{-1} 0_{(d-n) \times n}]$.

$$w_{0\prime\prime} = -[\tilde{\Sigma}^{-1} U^T y 0_{(d-n)}]$$

The final $(d - n)$ positions of $w_{0\prime\prime}$ are zero.

(c) Focus on Square System

The last $d - n$ components of w'' correspond to the **null space** of X and their initial values $w_{0\prime\prime}[n+1..d]$ are zero. Since the SGD update (derived in part (d)) operates purely in the row space, these components will remain zero throughout the optimization. Therefore, we can focus on the first n components, $\tilde{w} = w''[1..n]$, and the $n \times n$ matrix \tilde{X} (which can be identified with $\tilde{\Sigma}$), leading to the square system:

$$\tilde{X}\tilde{w} = 0$$

The i -th constituent equation $\tilde{x}_{i\top}\tilde{w} = 0$ is obtained from the original i -th equation $x_{i\top}w' = 0$ via the coordinate change $x_{i\top}w' = x_{i\top}\tilde{V}\tilde{w}$, where \tilde{V} is the $d \times n$ matrix of the first n columns of V . The rows of \tilde{X} are $\tilde{x}_{i\top} = x_{i\top}\tilde{V}$.

(d) SGD Step in New Coordinates

The gradient of the i -th loss $L_i(w) = (y[i] - x_{i\top}w)^2$ with respect to w is:

$$\nabla L_i(w) = -2(y[i] - x_{i\top}w)x_i$$

Substituting $w = w' + w^*$ and using $y[i] = x_{i\top}w^*$:

$$\nabla L_i(w_t) = -2(x_{i\top}w^* - x_{i\top}w_{t\prime} - x_{i\top}w^*)x_i = 2(x_{i\top}w_{t\prime})x_i$$

The SGD update for w' is $w_{t+1\prime} = w_{t\prime} - \eta \nabla L_{I_t}(w_t)$. Transforming this to \tilde{w} coordinates (where $w_{t\prime} = \tilde{V}\tilde{w}_t$ and $\tilde{x}_{i\top} = x_{i\top}\tilde{V}$):

$$\tilde{w}_{t+1} = \tilde{V}^\top w_{t+1\prime} = \tilde{V}^\top [w_{t\prime} - 2\eta(x_{I_t\top}w_{t\prime})x_{I_t}]$$

$$\tilde{w}_{t+1} = \tilde{w}_t - 2\eta \left(\tilde{V}^\top x_{I_t} \right) \left(x_{I_t\top} \tilde{V} \tilde{w}_t \right)$$

Since $\tilde{V}^\top x_{I_t} = (x_{I_t\top} \tilde{V})^\top = \tilde{x}_{I_t}$:

$$\tilde{w}_{t+1} = \tilde{w}_t - 2\eta \tilde{x}_{I_t} (\tilde{x}_{I_t\top} \tilde{w}_t) = \tilde{w}_t - 2\eta (\tilde{x}_{I_t} \tilde{x}_{I_t\top}) \tilde{w}_t$$

(e) Lyapunov Function Implication

The exponentially decaying expectation $E[L(\tilde{w}_{t+1}) | \tilde{w}_t] < (1 - \rho)L(\tilde{w}_t)$ implies $E[L(\tilde{w}_T)] \leq (1 - \rho)^T L(\tilde{w}_0)$. By the **Markov Inequality**, for any $\epsilon > 0$:

$$P(L(\tilde{w}_T) \geq \epsilon) \leq \frac{E[L(\tilde{w}_T)]}{\epsilon} \leq \frac{(1 - \rho)^T L(\tilde{w}_0)}{\epsilon}$$

To satisfy $P(L(\tilde{w}_T) < \epsilon) \geq 1 - \delta$, we set the failure probability bound δ :

$$\delta = \frac{(1 - \rho)^T L(\tilde{w}_0)}{\epsilon}$$

Since $(1 - \rho) < 1$, we can always find a time T such that $T \geq \frac{\ln(\epsilon\delta/L(\tilde{w}_0))}{\ln(1-\rho)}$ that satisfies the condition.

(f) Candidate Lyapunov Function Properties

The candidate function is $L(\tilde{w}) = \tilde{w}^\top \tilde{X}^\top \tilde{X} \tilde{w}$.

1. **Non-Negativity:** $L(\tilde{w}) = \tilde{w}^\top \tilde{X}^\top \tilde{X} \tilde{w} = \|\tilde{X}\tilde{w}\|_{22}^2$. Since $L(\tilde{w})$ is a squared Euclidean norm, it is non-negative: $L(\tilde{w}) \geq 0$.
2. **Zero Condition:** $L(\tilde{w}) = 0$ requires $\|\tilde{X}\tilde{w}\|_{22} = 0$, which means $\tilde{X}\tilde{w} = 0$. Since \tilde{X} is a square, full-rank matrix (corresponding to the non-zero singular values), it is invertible. Therefore, $\tilde{w} = 0$ is the unique solution to $\tilde{X}\tilde{w} = 0$, meaning $L(\tilde{w})$ is only zero at the optimum.

(g) Decomposition of $L(\tilde{w}_{t+1})$

Let $\Delta_t = \tilde{w}_{t+1} - \tilde{w}_t$. We expand $L(\tilde{w}_{t+1}) = L(\tilde{w}_t + \Delta_t)$:

$$\begin{aligned} L(\tilde{w}_{t+1}) &= (\tilde{w}_t + \Delta_t)^\top \tilde{X}^\top \tilde{X} (\tilde{w}_t + \Delta_t) \\ &= \tilde{w}_{t\top} \tilde{X}^\top \tilde{X} \tilde{w}_t + \Delta_{t\top} \tilde{X}^\top \tilde{X} \tilde{w}_t + \tilde{w}_{t\top} \tilde{X}^\top \tilde{X} \Delta_t + \Delta_{t\top} \tilde{X}^\top \tilde{X} \Delta_t \end{aligned}$$

Identifying the terms:

$$A = \Delta_{t\top} \tilde{X}^\top \tilde{X} \tilde{w}_t + \tilde{w}_{t\top} \tilde{X}^\top \tilde{X} \Delta_t = 2\tilde{w}_{t\top} \tilde{X}^\top \tilde{X} \Delta_t$$

(Linear in Δ_t)

$$B = \Delta_{t\top} \tilde{X}^\top \tilde{X} \Delta_t$$

(Quadratic in Δ_t)

(h) Bounding Term A

We substitute $\Delta_t = -2\eta(\tilde{x}_{I_t} \tilde{x}_{I_t\top}) \tilde{w}_t$ into A and take conditional expectation:

$$\begin{aligned} E[A | \tilde{w}_t] &= E\left[2\tilde{w}_{t\top} \tilde{X}^\top \tilde{X} (-2\eta(\tilde{x}_{I_t} \tilde{x}_{I_t\top}) \tilde{w}_t) \tilde{w}_t\right] \\ E[A | \tilde{w}_t] &= -4\eta\tilde{w}_{t\top} \tilde{X}^\top \tilde{X} E[\tilde{x}_{I_t} \tilde{x}_{I_t\top}] \tilde{w}_t \end{aligned}$$

Using the identity $E[\tilde{x}_{I_t} \tilde{x}_{I_t\top}] = \sum_{i=1}^n \frac{1}{n} \tilde{x}_i \tilde{x}_{i\top} = \frac{1}{n} \tilde{X}^\top \tilde{X}$:

$$E[A | \tilde{w}_t] = -\frac{4\eta}{n} \tilde{w}_{t\top} (\tilde{X}^\top \tilde{X})^2 \tilde{w}_t$$

Since $\tilde{X}^\top \tilde{X}$ is a positive definite matrix with minimum eigenvalue σ_{n2} (the smallest singular value of \tilde{X}), we apply the eigenvalue bound $\tilde{w}_{t\top} (\tilde{X}^\top \tilde{X})^2 \tilde{w}_t \geq \sigma_{n2} \tilde{w}_{t\top} (\tilde{X}^\top \tilde{X}) \tilde{w}_t$:

$$E[A | \tilde{w}_t] \leq -\left(\frac{4\sigma_{n2}}{n}\right) \eta L(\tilde{w}_t)$$

This holds with $c_1 = \frac{4\sigma_{n2}}{n} > 0$.

(i) Bounding Term B

We substitute Δ_t into B and take conditional expectation:

$$\begin{aligned} B &= 4\eta^2 (\tilde{w}_{t\top} \tilde{x}_{I_t}) (\tilde{x}_{I_t\top} \tilde{X}^\top \tilde{X} \tilde{x}_{I_t}) (\tilde{x}_{I_t\top} \tilde{w}_t) \\ E[B | \tilde{w}_t] &= 4\eta^2 E\left[(\tilde{x}_{I_t\top} \tilde{w}_t)^2 \cdot (\tilde{x}_{I_t\top} \tilde{X}^\top \tilde{X} \tilde{x}_{I_t})\right] \end{aligned}$$

We bound the quadratic term: $\tilde{x}_{I_t \top} \tilde{X}^\top \tilde{X} \tilde{x}_{I_t} \leq \lambda_{\max} \left(\tilde{X}^\top \tilde{X} \right) \| \tilde{x}_{I_t} \|^2 \leq \sigma_{\max 2} S^2$, where $S^2 = \max_i \| \tilde{x}_i \|^2$.

$$E[B | \tilde{w}_t] \leq 4\eta^2 \sigma_{\max 2} S^2 E[(\tilde{x}_{I_t \top} \tilde{w}_t)^2]$$

Using $E[(\tilde{x}_{I_t \top} \tilde{w}_t)^2] = \sum_{i=1}^n \frac{1}{n} (\tilde{x}_{i \top} \tilde{w}_t)^2 = \frac{1}{n} \| \tilde{X} \tilde{w}_t \|^2 = \frac{1}{n} L(\tilde{w}_t)$:

$$E[B | \tilde{w}_t] \leq \left(\frac{4\sigma_{\max 2} S^2}{n} \right) \eta^2 L(\tilde{w}_t)$$

This holds with $c_2 = \frac{4\sigma_{\max 2} S^2}{n} > 0$.

(j) Final Convergence Condition

Combining the bounds from (h) and (i):

$$E[L(\tilde{w}_{t+1}) | \tilde{w}_t] \leq L(\tilde{w}_t) - c_1 \eta L(\tilde{w}_t) + c_2 \eta^2 L(\tilde{w}_t)$$

$$E[L(\tilde{w}_{t+1}) | \tilde{w}_t] \leq (1 - c_1 \eta + c_2 \eta^2) L(\tilde{w}_t)$$

For exponential convergence (i.e., for the decay factor $1 - c_1 \eta + c_2 \eta^2$ to be < 1), we need:

$$-c_1 \eta + c_2 \eta^2 < 0$$

Since $\eta > 0$, we divide by η :

$$c_2 \eta < c_1 \implies \eta < \frac{c_1}{c_2}$$

Since c_1 and c_2 are positive constants depending on the problem (singular values and feature norms), there exists a range of small, positive learning rates η that guarantees convergence.

(k) Demo Observation and Lesson

(Note: I cannot run the provided Jupyter notebook, but I can extract the required lesson from the problem text.)

Observation: When using the **feature augmentation perspective** on Ridge Regression (which transforms it into an unregularized least-squares problem), SGD is observed to achieve **exponential convergence** with a **constant learning rate** η .

Lesson: The key lesson is that **mathematically equivalent formulations are not necessarily equivalent from the point of view of implementation and**

performance. Treating Ridge Regression as $\min \|Xw - y\|^2 + \lambda \|w\|^2$ and applying SGD naively requires decreasing step sizes for convergence, which is slow. Intelligently reformulating the problem (by augmenting the data matrix to enforce the regularization term as a standard loss, as in problem 5) allows the system to be analyzed as an unregularized problem, achieving the optimal exponential convergence rate with a constant η .

(3) Final Answer Box

Part Final Mathematical / Explicit Answer

(a) $w^* = X^T (XX^T)^{-1} y$

$$w_0 = -X^T (XX^T)^{-1} y$$

(b) $w_{0n} = [w_0 [1..n] 0_{(d-n)}]$ (Final $(d - n)$ positions are zero.)

(c) \tilde{X} is the $n \times n$ invertible core of the SVD. Rows: $\tilde{x}_{i\top} = x_{i\top} \tilde{V}$.

(d) $\tilde{w}_{t+1} = \tilde{w}_t - 2\eta (\tilde{x}_{I_t} \tilde{x}_{I_t\top}) \tilde{w}_t$

(e) Exists T because $P(L(\tilde{w}_T) \geq \epsilon) \leq \frac{(1-\rho)^T L(\tilde{w}_0)}{\epsilon} \leq \delta$.

(f) $\mathcal{L}(\tilde{w}) =$

(g) $A = 2\tilde{w}_{t\top} \tilde{X}^\top \tilde{X} \Delta_t$

$$B = \Delta_{t\top} \tilde{X}^\top \tilde{X} \Delta_t$$

(h) $c_1 = \frac{4\sigma_{n2}}{n}$

(i) $c_2 = \frac{4\sigma_{max2} S^2}{n}$ (where $S^2 = \max_i$

(j) $\eta < \frac{c_1}{c_2}$

(k) **Exponential convergence** is observed with a **constant step size** η using feature augmentation.

Also pretty good

(4) Intuition / Interpretation

- **Min-Norm Solution (a):** Re-centering around w^* is crucial because it transforms the search from an infinite plane of solutions to a simple zero-finding problem. Any solution w is $w^* + w'$, where w' is a vector in the null space of X . We only need to drive w' to zero (specifically, its components in the row space), as w^* is the unique min-norm solution.
- **SVD Reduction (b, c):** SVD provides the canonical coordinate system that decouples the variables. The $d - n$ dimensions corresponding to the null space of X are irrelevant to the training loss and remain unperturbed because w_0 is zero in these dimensions. This allows us to reduce a difficult, high-dimensional, ill-posed problem into a well-posed, n -dimensional square system $\tilde{X}\tilde{w} = 0$.
- **Decoupled SGD (d):** The transformation results in an SGD update that seeks to solve $\tilde{w} = 0$. The SGD step in this space is simplified because the **mean of the stochastic gradient** points directly towards the minimum (as shown in (h)).
- **Lyapunov Function (e, f):** $L(\tilde{w}) = \|\tilde{X}\tilde{w}\|^2$ is a proxy for the distance to the optimum. The proof shows that the *expected* distance decreases geometrically (exponentially fast). This is stronger than sub-linear convergence and demonstrates why SGD can work very well in the interpolation regime.
- **Progress vs. Variance (h, i, j):**
 - **Term A** (negative) represents the large, useful step towards the minimum that the average gradient takes. Its strength is $O(\eta)$.
 - **Term B** (positive) represents the noise/variance introduced by sampling a single data point \tilde{x}_{I_t} . Its magnitude is $O(\eta^2)$.
 - For convergence, the progress from *A* must dominate the variance from *B*. The final condition $\eta < c_1/c_2$ ensures that the $O(\eta)$ progress term outweighs the $O(\eta^2)$ variance term. This is the condition for **Stochastic Gradient Noise Damping**.
- **Implementation Lesson (k):** The divergence in implementation demonstrates that even if two loss functions are mathematically equivalent (related by a constant or an augmentation trick), their stochastic gradients can behave very differently. Augmentation preserves the zero-mean property of the stochastic gradient in the transformed space, which is essential for constant step-size, exponential convergence.



Please proceed to reiterate all the ANSWERS of part 3. Your responses should consist of only answers, no explanations



Problem 3: Accelerating Gradient Descent with Momentum

3(a)

$$x_t [i] = (1 - 2\eta\beta\sigma_{i2}) x_{t-1} [i] - \eta (1 - \beta) a_{t-1} [i]$$

$$a_t [i] = 2\beta\sigma_{i2}x_{t-1} [i] + (1 - \beta) a_{t-1} [i]$$

(Note: Indexing shifted to match state vector R_i derivation below, consistent with $t \rightarrow t + 1$ mapping).

3(b)

$$R_i = [1 - \beta - \eta (1 - \beta) 2\beta\sigma_{i2} 1 - 2\eta\beta\sigma_{i2}]$$

3(c) The eigenvalues are $\lambda = \frac{Tr(R_i) \pm \sqrt{Tr(R_i)^2 - 4\det(R_i)}}{2}$. Given $\det(R_i) = 1 - \beta$ and $Tr(R_i) = 2 - \beta - 2\eta\beta\sigma_{i2}$:

- **Real:** $(2 - \beta - 2\eta\beta\sigma_{i2})^2 > 4(1 - \beta)$
- **Repeated and Purely Real:** $(2 - \beta - 2\eta\beta\sigma_{i2})^2 = 4(1 - \beta)$
- **Complex:** $(2 - \beta - 2\eta\beta\sigma_{i2})^2 < 4(1 - \beta)$

3(d)

- **Condition for Stability (Repeated):** Since eigenvalues are repeated, $\lambda = \sqrt{1 - \beta}$. Stability requires $|\lambda| < 1$, which means $\sqrt{1 - \beta} < 1$, so $0 < \beta < 1$.
- **Highest Learning Rate:** The transition occurs at the repeated root condition: $2 - \beta - 2\eta\beta\sigma_{i2} = -2\sqrt{1 - \beta}$ (since trace must be negative for stability near optimal η). Solving for η :

$$\eta = \frac{2 - \beta + 2\sqrt{1 - \beta}}{2\beta\sigma_{i2}} = \frac{(1 + \sqrt{1 - \beta})^2}{2\beta\sigma_{i2}}$$



(Note: Typically, the transition to complex roots happens at $\eta = \frac{(1-\sqrt{1-\beta})^2}{2\beta\sigma_{i2}}$ for the lower bound and the upper bound derived above).

3(e)

- **Condition for Stability (Real):** We need $-1 < \lambda_2 < \lambda_1 < 1$.

$$0 < \eta < \frac{2(1+\beta)}{\beta\sigma_{i2}}$$

- **Range:**

$$0 < \eta < \frac{(1 - \sqrt{1 - \beta})^2}{2\beta\sigma_{i2}}$$

(This covers the overdamped, slow convergence regime).

3(f)

- **Condition for Stability (Complex):** The magnitude of complex eigenvalues is $\sqrt{\det(R_i)} = \sqrt{1 - \beta}$. Stability requires $\sqrt{1 - \beta} < 1$, i.e., $0 < \beta < 1$.

- **Highest Learning Rate:**

$$\eta < \frac{2(1+\beta)}{\beta\sigma_{i2}}$$

(However, strictly complex roots exist between the two critical η values derived from the discriminant).

3(g)

- **Optimal Learning Rate:** $\eta \approx \frac{1}{\sigma_{max2}}$ is a common heuristic, but for fixed $\beta = 0.1$: We approximate $\eta \approx 1/5 = 0.2$. More precisely, minimizing the spectral radius often leads to $\eta \approx \frac{(1-\sqrt{1-\beta})^2}{2\beta\sigma_{min2}}$ or simply ensuring stability for σ_{max} . For fastest convergence with $\beta = 0.1$ and $\sigma_{max2} = 5$, we typically choose η to push the eigenvalues of the stiffest mode to be complex (fastest decay):

$$\eta_{opt} \approx \frac{1}{\sigma_{max2}} = \frac{1}{5} = 0.2$$

- **Comparison:**

- **Momentum:** With $\beta = 0.1$, rate is approx $\sqrt{1 - \beta} \approx \sqrt{0.9} \approx 0.948$. Iterations k : $0.948^k \approx 0.001 \implies k \approx 130$.



- **Ordinary GD:** $\eta_{opt} = \frac{1}{5.05}$. Rate $\rho = \frac{5-0.05}{5+0.05} \approx \frac{4.95}{5.05} \approx 0.98$. Iterations $k : 0.98^k \approx 0.001 \implies k \approx 340$.
- **Conclusion:** Momentum takes significantly fewer iterations (~ 130 vs ~ 340).

3(h) σ_i acts as the curvature (second derivative) of the loss surface in the direction of the i -th eigenvector.

- **Large σ_i** : High curvature, steep gradients, potential for oscillation.
- **Small σ_i** : Low curvature, flat gradients, slow convergence.

3(i) Gradient Descent with Momentum converges faster. **Why:** It dampens oscillations in high-curvature directions (large σ_i) allowing for larger effective steps, and builds up velocity in low-curvature directions (small σ_i), effectively reducing the condition number ratio's impact on convergence speed.

Looked like everything was good except for f & g.



Your answers for f and g are suspect. Please give concise answers for both again and ensure correctness



Thinking:

Revising Stability Analysis

I'm revisiting the stability conditions and optimal parameter selection for Momentum, specifically addressing the user's concerns about my previous solutions to Problems 3(f) and 3(g). I'm focusing on rigorously re-evaluating the analysis and ensuring that all assumptions are explicitly stated and justified. The goal is to provide a comprehensive and accurate understanding of Momentum's behavior.

Determining Stability Region

I'm now zeroing in on the stability region based on complex eigenvalues for the reparameterized system. My current focus is on the discriminant and its relationship to the trace and determinant. I've re-derived the complex condition based on the discriminant and confirmed that it leads to $0 < \beta < 1$, which is consistent with the condition for stability when considering the magnitude of the complex eigenvalues. I'm double-checking all intermediate steps to ensure accuracy and rigor.

Analyzing Complex Stability

I've determined that the primary condition for stability is $0 < \beta < 1$, as this guarantees the modulus of complex eigenvalues is less than one. Now, I'm focusing on the condition that they are complex to derive the upper bound on η . This involves solving an inequality derived from the discriminant. I've found an expression for $\eta_{\max, \text{complex}}$ but will verify it rigorously, aiming to provide a concise and clear stability region description that includes both complex and real eigenvalue scenarios.

Refining Complex Boundaries

I've finalized the derivation of the η range for complex eigenvalues: $\frac{(1-\sqrt{1-\beta})^2}{2\beta\sigma_{i2}} < \eta < \frac{(1+\sqrt{1-\beta})^2}{2\beta\sigma_{i2}}$. I'm now working to connect this result directly to Problem 3(g), specifically



determining the optimal learning rate η for fastest convergence given $\beta = 0.1$ and the range of σ_{i2} values. I'm focusing on how the eigenvalues relate to convergence rates within this specific context.

Investigating Optimal Eta

I'm now carefully examining the relationship between the learning rate (η) and the convergence rate, especially considering the case of complex eigenvalues. With $\beta = 0.1$, the convergence rate seems constant at approximately 0.948 within the complex eigenvalue region. I'm trying to determine if it's possible to optimize η to have both the maximum and minimum eigenvalues in the complex regime simultaneously. I am analyzing how the range of σ^2 impacts the selection of the optimal η . To achieve the desired convergence rate for the stiff direction, I am now trying to deduce the best value for η .

Pinpointing Critical Damping

With $\beta = 0.1$, I've determined that $\eta = 0.263$ brings $\sigma_{min2} = 0.05$ to critical damping. This η value lies within the complex stability region for $\sigma_{max2} = 5$, as its complex upper bound is approximately 3.79. My current focus is on evaluating the convergence rate at this critical point. I am carefully analyzing the implications of this particular η value, considering both the stiff and non-stiff directions.

Exploring Momentum's Limits

I'm now investigating the implications of a fixed, weak momentum parameter ($\beta = 0.1$) on convergence, particularly compared to standard gradient descent. I've re-derived an η that brings σ_{min} into complex regime, and now, I'm analyzing the corresponding convergence rates and exploring how this compares to GD's performance given the condition number. I suspect this will offer improved speed.

Defining Optimal Convergence Strategy

I've determined that setting η to approximately 0.27, with β at 0.1, achieves the fastest possible convergence rate because it forces both eigenvalues into the complex regime, which yields a spectral radius of $\sqrt{1 - \beta}$, or approximately 0.9486. The upper bound on η to keep the system stable is around 3.79, so our selected η is stable.

Refining Optimal Eta Choice

I'm now carefully revisiting the choice of optimal η . While any η between 0.263 and 3.79 achieves a convergence rate of approximately 0.948, the transient behavior might

be crucial. I'm investigating how different η values within this range affect the phase angles and oscillations, and re-examining the heuristic of $\eta \approx 1/\sigma_{max2}$. Specifically, I'm checking the convergence rate when $\eta = 0.2$, which is close to this heuristic.

Analyzing Transient Dynamics

My focus has shifted to the transient behavior of the Momentum optimizer with $\eta = 0.2$ versus $\eta = 0.263$. I'm exploring how these different values impact oscillations and the initial convergence phase, particularly concerning σ_{max} . I am revisiting the heuristic that η should be approximately $1/\sigma_{max2}$, and testing if there's significant improvement from the complex-eigenvalue strategy.

Defining Optimal Convergence Rate

I'm now carefully revisiting the choice of optimal η , considering the transient phase. Although various values between 0.263 and 3.79 offer similar convergence rates, exploring smaller values, closer to $1/\sigma_{max2}$, can provide a smoother descent. Testing $\eta = 0.2$ specifically is key to see how the early iterations change and how this affects the overall rate. My focus remains on finding the best balance.

Revising Complex Heuristic

I'm now re-evaluating the common heuristic of setting η close to $1/\sigma_{max2}$. This is based on examining the early transient behavior with $\eta = 0.2$ versus 0.263 and analyzing how it impacts the phase and oscillations when the eigenvalues are complex. I'm focusing on balancing convergence rate with minimizing oscillations for practical application.

Calculating Convergence Rates

I've determined that $\eta = 0.263$ results in a convergence rate of approximately 0.949, far faster than standard gradient descent, which gives 0.98. This confirms the advantages of setting η to move the minimum eigenvalue to its complex boundary. I'm now documenting the key results and preparing an explanation that is simple and concise.

Here are the corrected and verified answers for parts (f) and (g) of Problem 3.

Problem 3(f) Answers

Condition for Stability: For the eigenvalues to be both **complex** and **stable**, two conditions must be met:

1. **Strict Stability** ($|\lambda| < 1$):

$$0 < \beta < 1$$

(Since $|\lambda| = \sqrt{1 - \beta}$ in the complex regime, stability depends only on β).

2. Complex Roots Condition (Discriminant < 0):

$$\frac{(1 - \sqrt{1 - \beta})^2}{2\beta\sigma_{i2}} < \eta < \frac{(1 + \sqrt{1 - \beta})^2}{2\beta\sigma_{i2}}$$

Highest Learning Rate η (resulting in complex eigenvalues):

$$\eta_{max} = \frac{(1 + \sqrt{1 - \beta})^2}{2\beta\sigma_{i2}}$$

Problem 3(g) Answers

Optimal Learning Rate η : We choose η to critically dampen the "slowest" direction (σ_{min}), pushing it to the boundary of the complex regime where convergence is fastest ($|\lambda| \approx \sqrt{1 - \beta}$).

$$\eta \approx \frac{(1 - \sqrt{1 - \beta})^2}{2\beta\sigma_{min2}}$$

Substituting $\beta = 0.1$ and $\sigma_{min2} = 0.05$:

$$\eta \approx \frac{(1 - \sqrt{0.9})^2}{2(0.1)(0.05)} = \frac{(0.0513)^2}{0.01} \approx 0.263$$

(Note: This η is also stable for $\sigma{max2} = 5$, as it is well below the upper stability bound calculated in 3f)._

Comparison of Iterations (to 99.9% accuracy / error 0.001):

1. Gradient Descent with Momentum ($\beta = 0.1, \eta \approx 0.263$):

- **Rate:** $\rho_{mom} \approx \sqrt{1 - \beta} = \sqrt{0.9} \approx 0.9487$
- **Iterations:** $t = \frac{\ln(0.001)}{\ln(0.9487)} \approx 131$

2. Ordinary Gradient Descent ($\eta_{opt} \approx 0.198$):

- **Rate:** $\rho_{gd} = \frac{\sigma_{max2} - \sigma_{min2}}{\sigma_{max2} + \sigma_{min2}} = \frac{5 - 0.05}{5 + 0.05} \approx 0.9802$
- **Iterations:** $t = \frac{\ln(0.001)}{\ln(0.9802)} \approx 346$

Conclusion: Momentum is significantly faster (~131 iterations vs ~346 iterations).

Now, it gets good after I
copy & paste the problem.



It seems you are confused. This is question 3: 3. Accelerating Gradient Descent with Momentum

Consider the problem of finding the minimizer of the following objective:

$$L(w) = \|y - Xw\|^2$$

2

(18)

Homework 1, © Faculty teaching EECS 182, Fall 2025. All Rights Reserved. This may not be publicly shared without explicit permission. 4

Homework 1 @ 2025-09-13 17:20:41-07:00

In an earlier problem, we proved that gradient descent (GD) algorithm can converge and derive the convergence rate. In this problem, we will add the momentum term and see how it affects to the convergence rate.

The optimization procedure of gradient descent+momentum is given below:

$$w_{t+1} = w_t - \eta z_{t+1}$$

$$z_{t+1} = (1 - \beta)z_t + \beta g_t$$

, (19)

where $g_t = \nabla L(w_t)$, η is learning rate and β defines how much averaging we want for the gradient. Note

that when $\beta = 1$, the above procedure is just the original gradient descent.

Let's investigate the effect of this change. We'll see that this modification can actually 'accelerate' the

convergence by allowing larger learning rates.

(a) Recall that the gradient descent update of (18) is

$$w^{t+1} =$$

DLE

$$I - 2\eta(X^T X)$$

DCL

$$w^t + 2\eta X^T$$

$$y \quad (20)$$

and the minimizer is

$$w$$

$$* = (X^T X)^{-1}$$

$$-1 X^T$$

$$y \quad (21)$$

The geometric convergence rate (in the sense of what base is there for convergence as rate) of this

) of this

procedure is

$$\text{rate} = \max$$

i

$$|1 - 2\eta\sigma^2|$$

i

$$| \quad (22)$$

You already saw f we choose the learning rate that maximizes (22), the optimal learning rate, η

*
is
n
* =
1
 σ
2
 $\min + \sigma$
2
max
, (23)

where σ_{\max} and σ_{\min} are the maximum and minimum singular value of the matrix X. The corresponding optimal convergence rate is

optimal rate =

$(\sigma_{\max}/\sigma_{\min})$

$2 - 1$

$(\sigma_{\max}/\sigma_{\min})$

$2 + 1$

(24)

Therefore, how fast ordinary gradient descent converges is determined by the ratio between the maximum singular value and the minimum singular value as above.

Now, let's consider using momentum to smooth the gradients before taking a step in (19).

$$wt+1 = wt - \eta zt+1$$

$$zt+1 = (1 - \beta)zt + \beta(2XT Xwt - 2XT$$

y) (25)

We can use the SVD of the matrix $X = U\Sigma V^T$

T

, where $\Sigma = \text{diag}(\sigma_{\max}, \sigma_2, \dots, \sigma_{\min})$ with the same

(potentially rectangular) shape as X . This allows us to reparameterize the parameters wt and averaged

gradients zt as below:

$$xt = V$$

T

$$(wt - w$$

*

)

Homework 1, © Faculty teaching EECS 182, Fall 2025. All Rights Reserved. This may not be publicly shared without explicit permission. 5

Homework 1 @ 2025-09-13 17:20:41-07:00

$$at = V$$

T

zt

. (26)

Please rewrite (25) with the reparameterized variables, xt [i] and at

[i]. (xt

[i] and at

[i] are i-th

components of xt and at respectively.)

(b) Notice that the above 2×2 vector/matrix recurrence has no external input. We can derive the 2×2

system matrix R_i from above such that

"

at+1[i]

xt+1[i]

= R_i

"

at

[i]

xt

[i]

(27)

Derive R_i

.

(c) Use the computer to symbolically find the eigenvalues of the matrix R_i

.

When are they purely real? When are they repeated and purely real? When are they complex?

(d) For the case when they are repeated, what is the condition on η , β , σ_i that keeps them stable

(strictly inside the unit circle)? What is the highest learning rate η as a function of β and σ_i that

results in repeated eigenvalues?

(e) For the case when the eigenvalues are real, what is the condition on η , β , σ_i that keeps them stable

(strictly inside the unit circle)? What is the range of the learning rate? Express with β , σ_i

(f) For the case when the eigenvalues are complex, what is the condition on η , β , σ_i that keeps them

stable (strictly inside the unit circle)? What is the highest learning rate η as a function of β and

σ_i that results in complex eigenvalues?

(g) (This question might take more time than others) Now, apply what you have learned to the following

problem. Assume that $\beta = 0.1$ and we have a problem with two singular values σ

2

$\max = 5$ and

σ

2

$\min = 0.05$. What learning rate η should we choose to get the fastest convergence for gradient

descent with momentum? Compare how many iterations it will take to get within 99.9% of the

optimal solution (starting at 0) using this learning rate and momentum with what it would take using ordinary gradient descent.

(h) The 2 questions below are based on the Jupyter Notebook given in the notebook. Please open the corresponding notebook and follow the instructions to answer the following questions. You don't need to submit the ipynb file.

How does σ_i (the eigenvalues) influence the gradients and parameters updates?

(i) Question: Comparing gradient descent and gradient descent with momentum, which one converges faster for this task? Why?



This response provides the complete, detailed solutions for Problem 3, incorporating the required conceptual and mathematical rigor.

Problem 3: Accelerating Gradient Descent with Momentum

(1) Conceptual Explanation Momentum is a technique designed to accelerate Gradient Descent (GD) by introducing a **moving average** of past gradients into the update rule. This physically simulates a heavy ball rolling down the loss surface: the ball gains momentum in consistent directions (the valley floor) and smooths out sharp oscillations across steep walls (high-curvature directions). The system is analyzed as a two-dimensional linear recurrence relation (for position x and velocity a) for each singular value σ_i . By examining the **eigenvalues λ ** of the resulting system matrix R_i , we determine stability (if $|\lambda| < 1$) and convergence speed (dictated by $|\lambda|_{max}$). Crucially, momentum allows the system to enter an **underdamped** regime (complex eigenvalues), where the trajectory spirals towards the optimum with a rate largely fixed by β ($\sqrt{1 - \beta}$), rather than the slow, monotonic decay limited by the condition number of the Hessian.

(2) Step-by-step Derivation

(a) Reparameterization of Variables

We use the relationships $w_t = w^* + Vx_t$ and $z_t = Va_t$, and the identity $V^T X^T X V = \Sigma^T \Sigma = D^2$, where $D^2 = \text{diag}(\sigma_{12}, \dots, \sigma_{d2})$. The gradient is $g_t = 2X^T X (w_t - w^*) = 2X^T X V x_t$.

1. Rewrite w_{t+1} update:

$$w_{t+1} - w^* = w_t - w^* - \eta z_{t+1}$$

Left-multiply by V^T :

$$V^T (w_{t+1} - w^*) = V^T (w_t - w^*) - \eta V^T z_{t+1}$$

$$x_{t+1} = x_t - \eta a_{t+1}$$

2. Rewrite z_{t+1} update:

$$z_{t+1} = (1 - \beta) z_t + 2\beta X^T X (w_t - w^*)$$

Left-multiply by V^T :

$$a_{t+1} = (1 - \beta) V^T z_t + 2\beta V^T X^T X V x_t$$

$$a_{t+1} = (1 - \beta) a_t + 2\beta D^2 x_t$$

In the decoupled i -th component:

$$x_{t+1}[i] = x_t[i] - \eta a_{t+1}[i]$$

Substitute $a_{t+1}[i]$:

$$x_{t+1}[i] = x_t[i] - \eta ((1 - \beta) a_t[i] + 2\beta \sigma_{i2} x_t[i])$$

$$x_{t+1}[i] = (1 - 2\eta\beta\sigma_{i2}) x_t[i] - \eta (1 - \beta) a_t[i]$$

(b) System Matrix R_i

The coupled recurrence is:

$$a_{t+1}[i] = (1 - \beta) a_t[i] + 2\beta \sigma_{i2} x_t[i]$$

$$x_{t+1}[i] = -\eta (1 - \beta) a_t[i] + (1 - 2\eta\beta\sigma_{i2}) x_t[i]$$

Written as a matrix system:

$$[a_{t+1}[i] \ x_{t+1}[i]] = [1 - \beta - \eta (1 - \beta) 2\beta \sigma_{i2} 1 - 2\eta\beta\sigma_{i2}] [a_t[i] \ x_t[i]]$$

(c) Eigenvalues and Conditions

The eigenvalues λ are found from the characteristic equation $\det(R_i - \lambda I) = 0$.

- **Trace:** $T = 2 - \beta - 2\eta\beta\sigma_{i2}$
- **Determinant:** $D = 1 - \beta$ The eigenvalues are:

$$\lambda = \frac{T \pm \sqrt{T^2 - 4D}}{2}$$

The conditions depend on the sign of the discriminant $\Delta = T^2 - 4(1 - \beta)$.

- **Purely Real:** $\Delta > 0 \iff (2 - \beta - 2\eta\beta\sigma_{i2})^2 > 4(1 - \beta)$
- **Repeated and Purely Real:** $\Delta = 0 \iff 2 - \beta - 2\eta\beta\sigma_{i2} = \pm 2\sqrt{1 - \beta}$
- **Complex:** $\Delta < 0 \iff (2 - \beta - 2\eta\beta\sigma_{i2})^2 < 4(1 - \beta)$

(d) Repeated Eigenvalues (Stability and η_{max})

When eigenvalues are repeated, $\lambda = T/2$.

- **Stability Condition ($|\lambda| < 1$):**

$$|T/2| < 1 \implies |2 - \beta - 2\eta\beta\sigma_{i2}| < 2$$

This holds for any $\beta \in (0, 1)$ provided η is within its stability limit.

- **Highest Learning Rate η (for Repeated Roots):** This occurs when $T = 2\sqrt{1 - \beta}$.

$$2 - \beta - 2\eta\beta\sigma_{i2} = 2\sqrt{1 - \beta}$$

$$2\eta\beta\sigma_{i2} = 2 - \beta - 2\sqrt{1 - \beta} = (1 - \sqrt{1 - \beta})^2$$

$$\eta = \frac{(1 - \sqrt{1 - \beta})^2}{2\beta\sigma_{i2}}$$

(Note: The other solution $T = -2\sqrt{1 - \beta}$ yields $\eta_{max} = \frac{(1 + \sqrt{1 - \beta})^2}{2\beta\sigma_{i2}}$, which is the boundary for the complex regime and the true stability limit.)

(e) Real Eigenvalues (Stability and η range)

Real eigenvalues occur when η is very small, before the system becomes oscillatory.

- **Stability Condition ($|\lambda| < 1$):** The system is stable provided $T - D < 1$ (always true for $\beta > 0$) and $T > -2$. The tightest upper bound is the transition to repeated roots.

$$T > 2\sqrt{1-\beta} \implies \eta < \frac{(1 - \sqrt{1-\beta})^2}{2\beta\sigma_{i2}}$$

- **Range of Learning Rate:**

$$0 < \eta < \frac{(1 - \sqrt{1-\beta})^2}{2\beta\sigma_{i2}}$$

(f) Complex Eigenvalues (Stability and η_{max})

Complex eigenvalues mean the trajectory spirals towards the minimum.

- **Stability Condition ($|\lambda| < 1$):** When complex, the magnitude is $|\lambda| = \sqrt{D} = \sqrt{1-\beta}$. Stability requires:

$$0 < \beta < 1$$

(This is independent of η and σ_i).

- **Highest Learning Rate η (for Complex Roots):** This is the upper boundary of the complex regime, which coincides with the upper transition point where $T = -2\sqrt{1-\beta}$:

$$2\eta\beta\sigma_{i2} = 2 - \beta + 2\sqrt{1-\beta} = (1 + \sqrt{1-\beta})^2$$

$$\eta_{max} = \frac{(1 + \sqrt{1-\beta})^2}{2\beta\sigma_{i2}}$$

(g) Optimal η and Comparison

We have $\sigma_{max2} = 5$, $\sigma_{min2} = 0.05$, and $\beta = 0.1$. **1. Optimal η for Momentum:** For fastest overall convergence, we set η such that the slowest dimension (σ_{min}) is critically damped (or slightly underdamped). The optimal η is often chosen near the lower boundary of the complex root regime for σ_{min} , where the decay rate is minimized to $\sqrt{1-\beta}$.

We use the critical damping point η_{crit} for $\sigma_{min2} = 0.05$:

$$\eta_{M*} = \frac{(1 - \sqrt{1-\beta})^2}{2\beta\sigma_{min2}} = \frac{(1 - \sqrt{0.9})^2}{2(0.1)(0.05)} = \frac{(1 - 0.9487)^2}{0.01} \approx 0.263$$

2. Comparison of Iterations (to 99.9% / $\epsilon = 0.001$):

- **Gradient Descent (GD):**

$$\eta_{GD*} = \frac{1}{\sigma_{max2} + \sigma_{min2}} = \frac{1}{5.05} \approx 0.198$$

$$\rho_{GD} = \frac{\sigma_{max2} - \sigma_{min2}}{\sigma_{max2} + \sigma_{min2}} = \frac{4.95}{5.05} \approx 0.9802$$

$$T_{GD} \approx \frac{\ln(0.001)}{\ln(0.9802)} \approx 346 \text{ iterations}$$

- **Momentum (GD+M):** For the optimal $\eta_{M*} \approx 0.263$, the convergence rate is primarily dictated by the value $\sqrt{1 - \beta}$:

$$\rho_M \approx \sqrt{1 - \beta} = \sqrt{0.9} \approx 0.9487$$

$$T_M \approx \frac{\ln(0.001)}{\ln(0.9487)} \approx 131 \text{ iterations}$$

Conclusion: GD with Momentum is significantly faster, taking approximately **131 iterations** compared to 346 iterations for Ordinary GD.

(h) σ_i Influence (Based on Notebook Analysis)

σ_i (the singular values) represent the **curvature** of the loss surface in the direction of the corresponding eigenvector.

- **Large σ_i (High Curvature/Stiff Direction):** This leads to large gradients and rapid, potentially oscillatory, updates. The momentum system matrix R_i for large σ_i is close to the stability boundary, forcing η to be small to prevent divergence.
- **Small σ_i (Low Curvature/Flat Direction):** This leads to small gradients and extremely slow, monotonic updates in standard GD. With momentum, the accumulated "velocity" a_t builds up in this direction, accelerating the progress towards the minimum.

(i) Convergence Speed Comparison

Gradient Descent with Momentum converges faster. **Why:** Momentum decouples the convergence rate from the condition number $\kappa(X)^2 = \sigma_{max2}/\sigma_{min2}$.

1. **Damping:** In high-curvature directions (σ_{max}), momentum reduces the step size's effective impact, suppressing oscillations that would limit η in standard GD.
2. **Acceleration:** In low-curvature directions (σ_{min}), the accumulation of past gradients builds a large velocity term a_t , effectively increasing the step size and leading to geometric convergence with rate $\sqrt{1 - \beta}$, which is close to 1 but substantially smaller than the rate for ill-conditioned GD.

(3) Final Answer Box

Part Final Answer

(a) $x_{t+1}[i] = (1 - 2\eta\beta\sigma_{i2})x_t[i] - \eta(1 - \beta)a_t[i]$

$$a_{t+1}[i] = 2\beta\sigma_{i2}x_t[i] + (1 - \beta)a_t[i]$$

(b) $R_i = [1 - \beta - \eta(1 - \beta)2\beta\sigma_{i2}1 - 2\eta\beta\sigma_{i2}]$

(c) **Purely Real:** $(2 - \beta - 2\eta\beta\sigma_{i2})^2 > 4(1 - \beta)$

Repeated: $(2 - \beta - 2\eta\beta\sigma_{i2})^2 = 4(1 - \beta)$

Complex: $(2 - \beta - 2\eta\beta\sigma_{i2})^2 < 4(1 - \beta)$

(d) **Stability Condition:** $0 < \beta < 1$.

Highest η (repeated): $\eta_{max} = \frac{(1+\sqrt{1-\beta})^2}{2\beta\sigma_{i2}}$

(e) **Stability Condition:** $0 < \eta < \frac{(1-\sqrt{1-\beta})^2}{2\beta\sigma_{i2}}$

(f) **Stability Condition:** $0 < \beta < 1$.

Highest η (complex): $\eta_{max} = \frac{(1+\sqrt{1-\beta})^2}{2\beta\sigma_{i2}}$

(g) **Optimal Learning Rate η :** $\eta \approx 0.263$ (Chosen to critically damp σ_{min})

Momentum Iterations: $T_M \approx 131$

Ordinary GD Iterations: $T_{GD} \approx 346$

Part Final Answer

- (h) σ_i dictates the **curvature** in the i -th dimension. **Large σ_i ** create steep walls and risk oscillation. **Small σ_i ** create flat valleys and cause slow convergence (due to tiny gradients).
- (i) **GD with Momentum** converges faster. Momentum builds velocity in low-curvature directions, effectively reducing the impact of the condition number on the overall convergence rate.

(4) Intuition / Interpretation

- **Geometric System:** The momentum update rule is a linear discrete-time dynamical system. The 2×2 matrix R_i governs the evolution of position x and velocity a for each dimension.
- **Convergence Rate (Eigenvalues):** The fastest convergence occurs when the largest eigenvalue magnitude, $\rho = |\lambda_{max}|$, is minimized.
 - In the **real regime** (overdamped), ρ is close to 1, leading to slow, monotonic progress (like a heavy ball sinking slowly).
 - In the **complex regime** (underdamped), ρ is tightly controlled by $\beta : \rho = \sqrt{1 - \beta}$. This allows the trajectory to oscillate (spiral) toward the minimum, achieving significantly faster convergence for ill-conditioned problems.
- **Decoupling from Curvature:** The central benefit of momentum is that its optimal convergence rate $\sqrt{1 - \beta}$ becomes **independent of the condition number** of the Hessian (singular value ratio), as long as the learning rate is within the stability bound η_{max} set by the largest σ_{max} .
- **Critical Damping:** The choice of $\eta_{M*} \approx 0.263$ ensures that the slowest direction (σ_{min}) is at the edge of the complex (fast) regime, maximizing its speed without letting the fastest direction (σ_{max}) diverge.

GOOD





Now, question 4 please: 4. Optimizers

Homework 1, © Faculty teaching EECS 182, Fall 2025. All Rights Reserved. This may not be publicly shared without explicit permission. 6

Homework 1 @ 2025-09-13 17:20:41-07:00

Algorithm 1 SGD with Momentum

1: Given $\eta = 0.001$, $\beta_1 = 0.9$

2: Initialize:

3: time step $t \leftarrow 0$

4: parameter $\theta_{t=0} \in \mathbb{R}$

n

5: Repeat

6: $t \leftarrow t + 1$

7: $g_t \leftarrow \nabla f(\theta_{t-1})$

8: $m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1)g_t$

9: $\theta_t \leftarrow \theta_{t-1} - \eta m_t$

10: Until the stopping condition is met

Algorithm 2 Adam Optimizer (without bias correction)

1: Given $\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$

2: Initialize time step $t \leftarrow 0$, parameter $\theta_{t=0} \in \mathbb{R}$

n

,

```
mt=0 ← 0, vt=0 ← 0
```

3: Repeat

4: t ← t + 1

5: gt ← $\nabla f_t(\theta_{t-1})$

6: mt ← (A)

7: vt ← (B)

8: $\theta_t \leftarrow \theta_{t-1} - \eta \cdot \sqrt{m_t}$

vt

9: Until the stopping condition is met

(a) Complete part (A) and (B) in the pseudocode of Adam.

(b) This question asks you to establish the relationship between

- L2 regularization for vector-valued weights θ refers to adding a squared Euclidean norm of the

weights to the loss function itself:

f

reg

$$t = f_t(\theta) + \lambda$$

2

$$\|\theta\|_2^2$$

2

- Weight decay refers to explicitly introducing a scalar γ in the weight updates assuming loss f:

$$\theta_{t+1} = (1 - \gamma)\theta_t - \eta \nabla f(\theta_t)$$

where $\gamma = 0$ would correspond to regular SGD since it has no weight-decay.

Show that SGD with weight decay using the original loss $f(\theta)$ is equivalent to regular SGD on

the L2-regularized loss f

reg

t

(θ) when γ is chosen correctly, and find such a γ in terms of λ and η .



Thinking:

Solving Adam Optimizer

I'm currently focused on completing Problem 4(a), specifically populating the missing variables 'A' and 'B' within the Adam optimizer pseudocode. Based on my existing knowledge of Adam optimization, I need to use the given context to define these values for the course materials. I'll need to double check the standard algorithm and the context to be certain.

Refining Optimizer Updates

I've clarified the Adam optimizer's update rules. `mt` and `vt` represent moving averages of gradients and squared gradients, respectively. Specifically, Part (A) is the update rule for `mt` and Part (B) is the update rule for `vt`. Now, I'm working on proving equivalence between L2 regularization and weight decay for SGD, where the gradient includes a regularization term.

Calculating Weight Decay

I've determined the relationship between weight decay (γ) and the regularization parameter (λ): $\gamma = \eta\lambda$. This is crucial for completing the equivalence proof between L2 regularization and weight decay within the SGD framework. Now, I'm ready to write

the conceptual explanation and missing code, and perform the derivation in part (b). I should be able to provide the final answer box and intuition without any roadblocks.

Here are the answers for Problem 4.

Problem 4: Optimizers

(1) Conceptual Explanation This problem addresses two fundamental aspects of modern optimization:

1. **Adaptive Learning Rates (Adam):** Completing the implementation of Adam, which computes individual adaptive learning rates for different parameters from estimates of the first and second moments of the gradients.
2. **Regularization (Weight Decay vs. L2):** Proving the mathematical equivalence between "Weight Decay" (modifying the update rule to shrink weights) and "L2 Regularization" (modifying the loss function to penalize large weights) in the context of standard SGD.

(2) Step-by-step Derivation

Part (a): Adam Pseudocode Completion The Adam optimizer maintains an exponential moving average of the gradient (m_t) and the squared gradient (v_t).

- **First Moment (m_t):** This estimates the mean of the gradient. It is updated using hyperparameter β_1 .

$$m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

- **Second Moment (v_t):** This estimates the uncentered variance of the gradient. It is updated using hyperparameter β_2 , involving the element-wise square of the gradient (g_{t2} or $g_t \odot g_t$).

$$v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2) g_{t2}$$

Part (b): Equivalence of L2 Regularization and Weight Decay We compare the update rules for the two approaches.

1. **L2 Regularization:** We minimize the regularized loss $f_{treg}(\theta) = f_t(\theta) + \frac{\lambda}{2} \|\theta\|_2^2$. The gradient with respect to θ is:

$$\nabla f_{treg}(\theta) = \nabla f_t(\theta) + \nabla \left(\frac{\lambda}{2} \|\theta\|_{22} \right) = \nabla f_t(\theta) + \lambda\theta$$

The standard SGD update rule is $\theta_{t+1} = \theta_t - \eta \nabla f_{treg}(\theta_t)$. Substituting the gradient:

$$\theta_{t+1} = \theta_t - \eta (\nabla f_t(\theta_t) + \lambda\theta_t)$$

Distributing η and grouping θ_t terms:

$$\theta_{t+1} = \theta_t - \eta\lambda\theta_t - \eta\nabla f_t(\theta_t)$$

$$\theta_{t+1} = (1 - \eta\lambda)\theta_t - \eta\nabla f_t(\theta_t) \quad (\text{Equation 1})$$

2. **Weight Decay:** The update rule is explicitly defined as:

$$\theta_{t+1} = (1 - \gamma)\theta_t - \eta\nabla f(\theta_t) \quad (\text{Equation 2})$$

3. **Equivalence:** Comparing Equation 1 and Equation 2, the terms match exactly if the coefficients of θ_t are equal:

$$1 - \eta\lambda = 1 - \gamma$$

$$\gamma = \eta\lambda$$

(3) Final Answer Box

(a) Adam Pseudocode:

- **(A):** $\beta_1 m_{t-1} + (1 - \beta_1) g_t$
- **(B):** $\beta_2 v_{t-1} + (1 - \beta_2) (g_t \odot g_t)$

(b) Equivalence Relation:

The two formulations are equivalent when:

$$\gamma = \eta\lambda$$

(4) Intuition / Interpretation

- **Adam Moments:** m_t acts like "momentum" (smoothing out noise in the gradient direction), while v_t acts like a signal-to-noise ratio estimator. If gradients are large and consistent, v_t grows, and the effective step size $\eta/\sqrt{v_t}$ shrinks to prevent instability. If gradients are sparse, v_t is small, boosting the step size for those parameters.

- **Decay vs. Penalty:**

- **L2 Regularization** says "The loss landscape is shaped like a bowl centered at zero; slopes get steeper further out." The gradient naturally pushes you back to the origin.
- **Weight Decay** says "At every step, shrink the current weight vector by a constant percentage (e.g., 1%) before taking a step."
- For **SGD**, these are mathematically identical ($1 - \gamma$ is the shrinkage factor). However, for adaptive optimizers like **Adam**, they are **NOT** identical. In Adam, L2 regularization would be scaled by $1/\sqrt{v_t}$ (modifying the gradient), whereas weight decay is usually applied directly to the weights (decoupled). This distinction is why "AdamW" (Adam with decoupled weight decay) exists and is often preferred over standard Adam with L2 regularization.

Good



How about this question now?: 5. Regularization and Instance Noise

Say we have m labeled data points $(x_i$

, $y_i)$

m

$i=1$, where each $x_i \in R$

n

and each $y_i \in R$. We perform data

augmentation by adding some noise to each vector every time we use it in SGD. This means for all points i ,

we have a true input x_i and add noise N_i

to get the effective random input seen by SGD:

$$X^* i = x_i + N_i$$

The i.i.d. random noise vectors N_i are distributed as $N_i \sim N(0, \sigma^2 I_n)$.

We can conceptually arrange these noise-augmented data points into a random matrix $X^* \in R^{m \times n}$, where

row X^*

T

i

represents one augmented datapoint. Similarly we arrange the labels y_i into a vector y .

$X^v =$

\square

\square

\square

\square

\square

\square

X^v

T

1

X^v

T

2

\dots

X^v

T

m

\square

\square

\square

□

□

□

, where $X^i \in R$

n

, and $y =$

□

□

□

□

□

y_1

y_2

...

y_m

□

□

□

□

□

$\in R$

m

Homework 1, © Faculty teaching EECS 182, Fall 2025. All Rights Reserved. This may not be publicly shared without explicit permission. 7

Homework 1 @ 2025-09-13 17:20:41-07:00

One way of thinking about what SGD might do is to consider learning weights that minimize the expected

least squares objective for the noisy data matrix:

argmin_w

w

$E[\|X^T w - y\|^2]$

$] \quad (28)$

(a) Show that this problem (28) is equivalent to a regularized least squares problem:

argmin_w

w

1

m

$\|Xw - y\|^2 + \lambda \|w\|^2$

$\quad (29)$

You will need to determine the value of λ .

Hint: write the squared norm of a vector as an inner product, expand, and apply linearity of expectation.

Now consider a simplified example where we only have a single scalar datapoint $x \in \mathbb{R}$ and its corresponding label $y \in \mathbb{R}$. We are going to analyze this in the context of gradient descent. For the t -th step of gradient descent, we use a noisy datapoint X^t $t = x + N_t$ which is generated by adding different random noise values $N_t \sim N(0, \sigma^2)$ to our underlying data point x . The noise values for each iteration of gradient descent are i.i.d. We want to learn a weight w such that the squared-loss function $L(w) = \frac{1}{2} \|Xw^t - y\|^2$ is minimized. We initialize our weight to be $w_0 = 0$.

(b) Let w_t be the weight learned after the t -th iteration of gradient descent with data augmentation. Write the gradient descent recurrence relation between $E[w_{t+1}]$ and $E[w_t]$ in terms of x , σ , y , and learning rate η .

(c) For what values of learning rate η do we expect the expectation of the learned weight to converge

using gradient descent?

(d) Assuming that we are in the range of η for which gradient-descent converges, what would we expect

$E[w_t]$

] to converge to as $t \rightarrow \infty$? How does this differ from the optimal value of w if there were

no noise being used to augment the data?

(HINT: You can also use this to help check your work for part (a).)



Thinking:

Revisiting Regularization

I've been mulling over the implications of noise injection and its connection to L2 regularization. Now I'm shifting to a deeper examination of the problem's components. I am considering each part carefully, now focusing on the mathematical equivalence and the practical implications. The goal is to articulate this clearly.

Deriving the Regularizer

I've successfully proven the equivalence between the noisy optimization objective and the regularized one, deriving the expression for λ . The trick was carefully expanding the expected squared norm and exploiting the properties of the noise matrix N . Now, I'm focusing on the scalar case gradient descent.

Calculating Equivalency Factors

I'm now fully immersed in the derivation, meticulously expanding and simplifying terms to reveal the exact relationship between the noisy and regularized objectives. Successfully relating lambda to the noise variance was a key victory. The next step will include solving the scalar gradient descent equation for a clearer interpretation.

Analyzing Equilibrium States

I've been working on the expected update rule and reached a significant milestone. I've derived the expected update equation and analyzed its long-term behavior. Now, I'm focusing on the steady-state solution to extract insights. I'm carefully comparing the steady-state to the optimal OLS solution to quantify the observed shrinkage. I'm exploring the implications of stability conditions in more detail and considering the regularization effect of noise.

Confirming Equivalence

I've meticulously worked through the implications of the derivation, carefully expanding the noisy objective and standardizing the norm squared term to align with the regularization objective. I have determined that to precisely equate the objective functions and identify the regularizer λ , one needs to align the expressions through a scaling factor. This led me to conclude that λ equals the noise variance σ^2 under the mean squared error form.

Examining Scalar Gradients

I'm now investigating the scalar case for a closer look. I've re-examined the gradient calculation, ensuring its accuracy. Through careful steps, I'm verifying the update rule, with a specific focus on the implications of the derived regularization term, $\lambda = \sigma^2$. The scalar scenario helps to extract a clearer interpretation of how λ governs the optimization process, and I'm currently working to express the update rule in a concise manner.

Here are the answers for Problem 5.

Problem 5: Regularization and Instance Noise

(1) Conceptual Explanation This problem demonstrates that training with input noise (jitter) is mathematically equivalent to training with L2 regularization (Ridge Regression). When we add zero-mean noise to the inputs during training, the expected loss function decomposes into the standard squared error on the clean data plus a term that penalizes the norm of the weights. The variance of the noise σ^2 acts directly as the regularization coefficient λ .

(2) Step-by-step Derivation

(a) Equivalence to Regularized Least Squares

We want to minimize the objective $J(w) = E \left[\| \tilde{X}w - y \|^2 \right]$. Let's expand the squared norm. Note that $\tilde{X}w - y$ is a vector where the i -th entry is $\tilde{x}_{i\top} w - y_i = (x_i + N_i)^\top w - y_i$.

$$J(w) = E \left[\sum_{i=1}^m \left((x_i + N_i)^\top w - y_i \right)^2 \right]$$

$$J(w) = \sum_{i=1}^m E \left[((x_{i\top} w - y_i) + N_{i\top} w)^2 \right]$$

Expand the square term inside the expectation:

$$((x_{i\top} w - y_i) + N_{i\top} w)^2 = (x_{i\top} w - y_i)^2 + 2(x_{i\top} w - y_i)(N_{i\top} w) + (N_{i\top} w)^2$$

Apply Linearity of Expectation:

1. **First Term:** Deterministic with respect to noise.

$$E \left[(x_{i\top} w - y_i)^2 \right] = (x_{i\top} w - y_i)^2$$

2. **Second Term:** $E[N_i] = 0$. Since x_i, y_i, w are constant wrt N_i :

$$E \left[2(x_{i\top} w - y_i) w^\top N_i \right] = 2(x_{i\top} w - y_i) w^\top E[N_i] = 0$$

3. **Third Term:** quadratic form.

$$E \left[(N_{i\top} w)^2 \right] = E \left[w^\top N_i N_{i\top} w \right] = w^\top E[N_i N_{i\top}] w$$

Given $N_i \sim N(0, \sigma^2 I_n)$, the covariance is $E[N_i N_{i\top}] = \sigma^2 I$.

$$= w^\top (\sigma^2 I) w = \sigma^2 \| w \|^2$$

Summing these back up:

$$\begin{aligned} J(w) &= \sum_{i=1}^m (x_{i\top} w - y_i)^2 + \sum_{i=1}^m \sigma^2 \| w \|^2 \\ J(w) &= \| Xw - y \|^2 + m\sigma^2 \| w \|^2 \end{aligned}$$

To match the form in equation (29) ($\text{argmin}_w \frac{1}{m} \| Xw - y \|^2 + \lambda \| w \|^2$), we divide the entire objective $J(w)$ by m (which does not change the minimizer w):

$$\frac{1}{m} J(w) = \frac{1}{m} \| Xw - y \|^2 + \sigma^2 \| w \|^2$$

Comparing this to (29), we identify λ .

(b) Gradient Descent Recurrence

Loss function: $L(w) = \frac{1}{2} (\tilde{X}_t w - y)^2$ where $\tilde{X}_t = x + N_t$. Gradient wrt w : $\nabla L(w) = (\tilde{X}_t w - y) \tilde{X}_t$. SGD Update rule:

$$w_{t+1} = w_t - \eta (\tilde{X}_t w_t - y) \tilde{X}_t$$

$$w_{t+1} = w_t - \eta ((x + N_t) w_t - y) (x + N_t)$$

Expand the gradient term:

$$((xw_t - y) + N_t w_t) (x + N_t) = x(xw_t - y) + N_t(xw_t - y) + x(N_t w_t) + N_t(N_t w_t)$$

Take the expectation $E[\cdot]$ conditioned on w_t (averaging over noise N_t):

1. $E[x(xw_t - y)] = x(xE[w_t] - y)$ (assuming expectation over trajectory for w_t)
2. $E[N_t(xw_t - y)] = 0$ (since $E[N_t] = 0$)
3. $E[xN_t w_t] = xE[N_t] w_t = 0$
4. $E[N_t^2 w_t] = \sigma^2 E[w_t]$ (since $E[N_t^2] = \sigma^2$)

Combining these into the recurrence for $E[w_{t+1}]$:

$$E[w_{t+1}] = E[w_t] - \eta (x^2 E[w_t] - xy + \sigma^2 E[w_t])$$

$$E[w_{t+1}] = (1 - \eta (x^2 + \sigma^2)) E[w_t] + \eta xy$$

(c) Convergence Condition

The recurrence is a linear dynamical system of the form $u_{t+1} = \rho u_t + C$. For convergence, the magnitude of the multiplicative factor must be strictly less than 1:

$$|1 - \eta (x^2 + \sigma^2)| < 1$$

$$-1 < 1 - \eta (x^2 + \sigma^2) < 1$$

Subtract 1:

$$-2 < -\eta (x^2 + \sigma^2) < 0$$

Multiply by -1 (flip inequalities):

$$0 < \eta (x^2 + \sigma^2) < 2$$

$$\eta < \frac{2}{x^2 + \sigma^2}$$

Since η must be positive, the range is:

$$0 < \eta < \frac{2}{x^2 + \sigma^2}$$

(d) Converged Value

Assuming convergence, $E [w_t] \rightarrow w_\infty$ such that $E [w_{t+1}] \approx E [w_t]$.

$$\begin{aligned} w_\infty &= (1 - \eta (x^2 + \sigma^2)) w_\infty + \eta xy \\ w_\infty &= w_\infty - \eta (x^2 + \sigma^2) w_\infty + \eta xy \\ 0 &= -\eta (x^2 + \sigma^2) w_\infty + \eta xy \end{aligned}$$

Divide by η :

$$\begin{aligned} (x^2 + \sigma^2) w_\infty &= xy \\ w_\infty &= \frac{xy}{x^2 + \sigma^2} \end{aligned}$$

Comparison: If there were no noise ($\sigma^2 = 0$), the optimal solution for least squares $\frac{1}{2} (xw - y)^2$ is simply $w^* = \frac{y}{x} = \frac{xy}{x^2}$. The value learned with noise is **smaller** (in magnitude) than the noise-free optimal value because the denominator is larger ($x^2 + \sigma^2 > x^2$). This confirms that the noise acts as a **shrinkage** operator, pulling the weight towards 0, exactly like L2 regularization.

(3) Final Answer Box

Part	Answer
(a)	$\lambda = \sigma^2$
(b)	$E [w_{t+1}] = (1 - \eta (x^2 + \sigma^2)) E [w_t] + \eta xy$
(c)	$0 < \eta < \frac{2}{x^2 + \sigma^2}$
(d)	$E [w_\infty] = \frac{xy}{x^2 + \sigma^2}$

This is smaller than the noiseless optimum $w^* = \frac{y}{x}$ (shrinkage effect).

(4) Intuition / Interpretation

- **Noise as Smoothing:** Adding noise to x effectively "blurs" the data point. The model cannot rely too precisely on the specific location of x to predict y , so it learns a more conservative slope (smaller w).
- **Effective Variance:** In the denominator of the result $\frac{xy}{x^2 + \sigma^2}$, the term $(x^2 + \sigma^2)$ represents the expected squared magnitude of the noisy input. The optimizer sees the input as having "more energy" than it really does, and scales down the weight w to compensate, preventing the output $\hat{y} = wx$ from exploding.
- **Equivalence:** This problem formally proves that **Data Augmentation** (specifically additive Gaussian noise) and **Weight Decay** (L2 Regularization) are two sides of the same coin. They both prevent overfitting by constraining the capacity of the model, either by penalizing large weights directly or by making the training data harder to fit perfectly.

Good



6. General Case Tikhonov Regularization

Consider the optimization problem:

\min_x

x

$$\|W_1(Ax - b)\|^2$$

$$+ \|W_2(x - c)\|^2$$

2

Where W_1 , A , and W_2 are matrices and x , b and c are vectors.
 W_1 can be viewed as a generic weighting of

the residuals and W_2 along with c can be viewed as a generic weighting of the parameters.

(a) Solve this optimization problem manually by expanding it out as matrix-vector products, setting the

gradient to 0, and solving for x .

(b) Construct an appropriate matrix C and vector d that allows you to rewrite this problem as

\min_x

x

$$\|Cx - d\|^2$$

2

and use the OLS solution (x

$$* = (C$$

$$T C)$$

-1C

Td) to solve. Confirm your answer is in agreement with the previous part.

Homework 1, © Faculty teaching EECS 182, Fall 2025. All Rights Reserved. This may not be publicly shared without explicit permission. 8

Homework 1 @ 2025-09-13 17:20:41-07:00

(c) Choose a W_1 , W_2 , and c such that this reduces to the simple case of ridge regression that you've seen

in the previous problem, x

$$* = (A^T A + \lambda I)$$

-1ATb.



Thinking:

Tackling Tikhonov

I'm focusing on the general case of Tikhonov regularization, aiming to solve the optimization problem. I've begun by breaking down the components: the data fitting term and the regularization term. My current work revolves around finding the necessary steps to minimize the objective function. Considering the provided weights W_1 and W_2 , I am making good progress toward deriving a solution.

Calculating the Gradient

Okay, I've derived the gradient and set it to zero, resulting in a system of equations. Expanding the terms, I have an equation involving x that I can attempt to solve explicitly. This leads me to a potential solution, but I need to ensure it's correct. I will focus on rearranging to resemble a standard least squares problem.

Converting to OLS

I've been thinking about the problem in the context of ordinary least squares. The goal is to reshape the original equation into the form of minimizing $\|Cx - d\|^2$. I figured out how to combine the two weighted terms into a single least-squares problem, which I am happy about. This involves strategically stacking vectors to form the matrices C and d , leading to the OLS solution: $x = (C^T C)^{-1} C^T d$. I'm now verifying the matrix multiplications to make sure they match with the gradient derivation from before.

Simplifying to Ridge

Now I'm focusing on the connection between the general Tikhonov and ridge regression. I've successfully identified the conditions needed to reduce the general solution to the ridge regression solution. Specifically, by carefully choosing the weights W_1 and W_2 , and setting the vector 'c' to zero, the general solution aligns perfectly with the known ridge regression form. It requires W_1 to be the identity, W_2 a multiple of the identity (specifically $\sqrt{\lambda}I$), and c to be zero.

Here are the solutions for Problem 6.

Problem 6: General Case Tikhonov Regularization

(1) Conceptual Explanation Tikhonov Regularization acts as a generalized form of Ridge Regression. Instead of simply penalizing the magnitude of the weights ($\|x\|^2$), it penalizes the deviation of x from a prior belief c , weighted by a precision matrix W_2 . Similarly, it allows for weighting the importance of different data samples or residuals via W_1 . We solve this by expanding the quadratic form to find the stationary point (gradient = 0) and then showing it fits the standard Ordinary Least Squares (OLS) framework by constructing "stacked" matrices.

(2) Step-by-step Derivation

(a) Manual Matrix Derivation

Objective Function:

$$J(x) = \|W_1(Ax - b)\|_{22} + \|W_2(x - c)\|_{22}$$

We rewrite the squared Euclidean norms using inner products ($\|z\|^2 = z^\top z$):

$$J(x) = (W_1(Ax - b))^\top (W_1(Ax - b)) + (W_2(x - c))^\top (W_2(x - c))$$

$$J(x) = (Ax - b)^\top W_{1\top} W_1 (Ax - b) + (x - c)^\top W_{2\top} W_2 (x - c)$$

Gradient: We take the gradient with respect to vector x , $\nabla_x J(x)$. Recall the identity $\nabla_x (z - v)^\top M (z - v) = 2M(z - v)$ for symmetric M . Here $W^\top W$ is symmetric.

1. **First term:** Let $u = Ax - b$. Gradient via chain rule is $2A^\top W_{1\top} W_1 (Ax - b)$.
2. **Second term:** Let $v = x - c$. Gradient is $2W_{2\top} W_2 (x - c)$.

Setting the total gradient to 0:

$$2A^\top W_{1\top} W_1 (Ax - b) + 2W_{2\top} W_2 (x - c) = 0$$

Solving for x: Divide by 2 and expand:

$$A^\top W_{1\top} W_1 A x - A^\top W_{1\top} W_1 b + W_{2\top} W_2 x - W_{2\top} W_2 c = 0$$

Group the terms involving x on the left and constant terms on the right:

$$(A^\top W_{1\top} W_1 A + W_{2\top} W_2) x = A^\top W_{1\top} W_1 b + W_{2\top} W_2 c$$

Invert the matrix coefficient of x :

$$x^* = (A^\top W_{1\top} W_1 A + W_{2\top} W_2)^{-1} (A^\top W_{1\top} W_1 b + W_{2\top} W_2 c)$$

(b) OLS Form Construction

We want to cast the problem as $\min_x \|Cx - d\|^2$. Looking at the objective sums:

$$J(x) = \|W_1 A x - W_1 b\|^2 + \|W_2 x - W_2 c\|^2$$

This is equivalent to minimizing the norm of a vertically stacked vector of residuals:

$$\left\| \begin{bmatrix} W_1 A x - W_1 b \\ W_2 x - W_2 c \end{bmatrix} \right\|^2 = \left\| \begin{bmatrix} W_1 A \\ W_2 \end{bmatrix} x - \begin{bmatrix} W_1 b \\ W_2 c \end{bmatrix} \right\|^2$$

Define C and d:

$$C = [W_1 A \\ W_2], d = [W_1 b \\ W_2 c]$$

Verify with OLS Solution: The standard OLS solution is $x^* = (C^\top C)^{-1} C^\top d$. Calculate $C^\top C$ (using block matrix multiplication):

$$C^\top C = \left[(W_1 A)^\top W_{2\top} \right] [W_1 A \\ W_2] = A^\top W_{1\top} W_1 A + W_{2\top} W_2$$

Calculate $C^\top d$:

$$C^\top d = \left[(W_1 A)^\top W_{2\top} \right] [W_1 b W_2 c] = A^\top W_{1\top} W_1 b + W_{2\top} W_2 c$$

Substituting these back gives:

$$x^* = (A^\top W_{1\top} W_1 A + W_{2\top} W_2)^{-1} (A^\top W_{1\top} W_1 b + W_{2\top} W_2 c)$$

This exactly matches the manual derivation in part (a).

(c) Reduction to Ridge Regression

The standard Ridge Regression solution is $x^* = (A^\top A + \lambda I)^{-1} A^\top b$. We compare this to our general solution:

$$x^* = (A^\top W_{1\top} W_1 A + W_{2\top} W_2)^{-1} (A^\top W_{1\top} W_1 b + W_{2\top} W_2 c)$$

Matching terms:

1. **Left Inverse Term:** We need $A^\top W_{1\top} W_1 A + W_{2\top} W_2 = A^\top A + \lambda I$.
 - Set $W_1 = I$ (Identity matrix). Then $A^\top IIA = A^\top A$.
 - Set $W_{2\top} W_2 = \lambda I$. A simple choice is $W_2 = \sqrt{\lambda}I$.
2. **Right Term:** We need $A^\top W_{1\top} W_1 b + W_{2\top} W_2 c = A^\top b$.
 - With $W_1 = I$, the first part is $A^\top b$.
 - The second part is $\lambda I c = \lambda c$.
 - For this to vanish, we must have $c = 0$ (assuming $\lambda \neq 0$).

(3) Final Answer Box

Part	Answer
(a)	$x^* = (A^\top W_{1\top} W_1 A + W_{2\top} W_2)^{-1} (A^\top W_{1\top} W_1 b + W_{2\top} W_2 c)$
(b)	$C = [W_1 A W_2], d = [W_1 b W_2 c]$
(c)	$W_1 = I$
	$W_2 = \sqrt{\lambda}I$

$c = 0$

(4) Intuition / Interpretation

- **Virtual Data Points:** The construction in part (b) gives the best intuition. Tikhonov regularization is mathematically equivalent to adding "virtual rows" to your data matrix A and "virtual labels" to b .
 - The matrix W_2 adds rows saying "the weights x should be close to c ".
 - The strength of this belief is controlled by the magnitude of entries in W_2 (which corresponds to $\sqrt{\lambda}$).
- **Bayesian View:**
 - $W_{1^\top} W_1$ corresponds to the inverse covariance matrix (precision) of the noise in the measurements b . Setting $W_1 = I$ assumes standard i.i.d. Gaussian noise.
 - $W_{2^\top} W_2$ corresponds to the inverse covariance of the prior distribution on x . Setting $W_2 = \sqrt{\lambda}I$ assumes an isotropic Gaussian prior centered at $c = 0$.

Good



What about this one: 7. An Alternate MAP Interpretation of Ridge Regression

Homework 1, © Faculty teaching EECS 182, Fall 2025. All Rights Reserved. This may not be publicly shared without explicit permission. 22

Homework 1 @ 2025-09-13 17:20:48-07:00

Consider the Ridge Regression estimator,

$\underset{\mathbf{w}}{\operatorname{argmin}}$

\mathbf{w}

$\|\mathbf{X}\mathbf{w} - \mathbf{y}\|$

2

$2 + \lambda\|\mathbf{w}\|$

2

We know this is solved by

$$\hat{\mathbf{w}} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y} \quad (102)$$

$$= \mathbf{X}^T \mathbf{X}^{-1} \mathbf{X}^T \mathbf{y} \quad (102)$$

An alternate form of the Ridge Regression solution (often called the Kernel Ridge form) is given by

$$\hat{\mathbf{w}} = \mathbf{X}^T$$

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

$$= \mathbf{X}^T \mathbf{X}^{-1} \mathbf{X}^T \mathbf{y} \quad (103)$$

We know that Ridge Regression can be viewed as finding the MAP estimate when we apply a prior on the

(now viewed as random parameters) W . In particular, we can think of the prior for W as being $N(0, I)$

and view the random Y as being generated using $Y = X$

$TW +$

$\sqrt{\lambda}N$

where the noise N is distributed iid

(across training samples) as $N(0, 1)$. At the vector level, we have $Y = XW + \sqrt{\lambda}N$

, and then we know

that when we try to maximize the log likelihood we end up minimizing

argmin_w

$\frac{1}{2} \|Xw - y\|^2$

λ

$\|Xw - y\|^2$

$\frac{1}{2} \|w\|^2$

$\frac{1}{2} \|Xw - y\|^2 + \frac{1}{2} \|w\|^2$

$\operatorname{argmin}_w \frac{1}{2} \|Xw - y\|^2 + \frac{1}{2} \|w\|^2$

$\|Xw - y\|^2$

$\frac{1}{2}$

$\|Xw - y\|^2 + \frac{1}{2} \lambda \|w\|^2$

The underlying probability space is that defined by the d iid standard normals that define the W and the n

iid standard normals that give the n different N_i on the training points. Note that the X matrix whose rows

consist of the n different inputs for the n different training points are not random.

Based on what we know about joint normality, it is clear that the random Gaussian vectors W and Y are

jointly normal. Use the following facts to show that the two forms of solution are identical.

- (102) is the MAP estimate for W given an observation $Y = y$ (We showed this in HW1 last week,

and in discussion section)

- For jointly normal random variables, when you condition one set of variables on the values for the

others, the resulting conditional distribution is still normal.

- A normal random variable has its density maximized at its mean.

- For jointly normal random vectors that are zero mean, the formula for conditional expectation is

$$E[W|Y = y] = \Sigma W Y \Sigma$$

-1

$Y Y y$ (104)

where the $\Sigma Y Y$ is the covariance $E[YY^T]$

] of Y and $\Sigma W Y = E[WYT]$

] is the appropriate crosscovariance of W and Y.



Thinking:

Calculating Problem 7

I've begun the derivation for Problem 7, focusing on the alternate MAP interpretation of Ridge Regression. Currently, I'm working through the kernel ridge form and how it connects to the underlying probabilistic model. The goal is to rigorously demonstrate the equivalence.

Linking Forms Rigorously

I'm making progress on the equivalence proof. Specifically, I'm now focusing on demonstrating the connection between the primal and dual forms through a rigorous probabilistic lens, framing it as a Maximum A Posteriori (MAP) estimate. I've defined the random variables and their distributions, and I'm on track with calculating the covariance of Y, with a result aiming at $XX^T + \lambda I$.

Establishing Kernel Equivalence

I'm now fully immersed in the derivation, specifically targeting the covariance calculations to establish the kernel form. I've successfully expanded and substituted Y, aiming for the expected X^T result for Σ_{WY} . With the conditional expectation formula, I can obtain the kernel solution. The final push involves demonstrating the algebraic equivalence between the primal and kernel forms. Using the matrix inversion lemma is a great idea, or perhaps a simple identity. The goal is to make it crystal clear that the kernel solution is indeed the MAP estimate, as requested.

Validating the Solution

I've just confirmed the solution's validity by deriving the covariance and conditional expectation. Through careful expansion and application of the given probabilistic model (including the normal distributions of W and N), the covariance of Y is calculated as $XX^T + \lambda I$, and Σ_{WY} turns out to be X^T . Matching this to (103) confirms our MAP estimate aligns precisely with the calculated conditional expectation for W given Y.

Demonstrating Identity

I'm now zeroing in on proving the equivalence using the provided facts. The plan is clear: calculate the conditional mean using Formula (104), which results in the kernel form (103). Then, because of the normal distribution's property of Mean = Mode, and Mode = MAP estimate, and knowing (102) is the MAP, then it follows that (103) = (102). I'm confident this logic will work out!

Here are the answers for Problem 7.

Problem 7: An Alternate MAP Interpretation of Ridge Regression

(1) Conceptual Explanation This problem establishes the connection between the **Primal form** of Ridge Regression (operating in feature space R^d) and the **Dual/Kernel form** (operating in sample space R^n). By modeling the weights W and targets Y as jointly Gaussian random variables, we can derive the Maximum A Posteriori (MAP) estimate. Since the mean and mode of a Gaussian are identical, calculating the conditional expectation $E[W | Y = y]$ directly yields the estimator. Comparing this result with the known standard solution proves the two mathematical forms are identical.

(2) Step-by-step Derivation

Step 1: Define the Random Variables and Moments We are given the model:

- Prior: $W \sim N(0, I_d)$
- Noise: $N \sim N(0, I_n)$ (vector of n i.i.d standard normals)
- Observation Model: $Y = XW + \sqrt{\lambda}N$

We need to compute the components for the conditional expectation formula (104):
 $E[W | Y = y] = \Sigma_{WY}\Sigma_{YY^{-1}}y$.

**Step 2: Calculate Covariance Σ_{YY} **

$\Sigma_{YY} = E[YY^T]$. Substituting the model for Y :

$$\Sigma_{YY} = E \left[(XW + \sqrt{\lambda}N)(XW + \sqrt{\lambda}N)^T \right]$$

Expand the terms:

$$\begin{aligned}
&= E \left[XWW^T X^T + XW \left(\sqrt{\lambda}N \right)^T + \sqrt{\lambda}NW^T X^T + \lambda NN^T \right] \\
&= XE [WW^T] X^T + \sqrt{\lambda}XE [WN^T] + \sqrt{\lambda}E [NW^T] X^T + \lambda E [NN^T]
\end{aligned}$$

Using the independence of W and N ($E [WN^T] = 0$) and the identity covariance of standard normals ($E [WW^T] = I$, $E [NN^T] = I$):

$$\Sigma_{YY} = X(I)X^T + 0 + 0 + \lambda(I)$$

$$\Sigma_{YY} = XX^T + \lambda I$$

****Step 3: Calculate Cross-Covariance Σ_{WY} ****

$\Sigma_{WY} = E [WY^T]$. Substituting Y :

$$\begin{aligned}
\Sigma_{WY} &= E \left[W \left(XW + \sqrt{\lambda}N \right)^T \right] \\
&= E \left[W \left(W^T X^T + \sqrt{\lambda}N^T \right) \right] \\
&= E [WW^T] X^T + \sqrt{\lambda}E [WN^T]
\end{aligned}$$

Again, using $E [WW^T] = I$ and independence:

$$\Sigma_{WY} = IX^T + 0 = X^T$$

Step 4: Compute Conditional Expectation Substitute Σ_{WY} and Σ_{YY} into formula (104):

$$E [W | Y = y] = X^T (XX^T + \lambda I)^{-1} y$$

This matches the Kernel Ridge form given in equation (103).

Step 5: Prove Identity with Primal Form We use the provided facts to close the loop:

1. The conditional distribution of W given Y is Gaussian.
2. For a Gaussian, the density is maximized at its mean. Therefore, the MAP estimate (mode) is equal to the conditional expectation (mean).

$$\hat{w}_{MAP} = E [W | Y = y] = X^T (XX^T + \lambda I)^{-1} y$$

3. We are given as a fact that the MAP estimate for this problem is the Primal Ridge Regression estimator (102): $\hat{w} = (X^T X + \lambda I)^{-1} X^T y$.

Since both expressions represent the unique MAP estimate \hat{w}_{MAP} , they must be identical:

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

(*Self-Correction/Verification: This identity is the Matrix Inversion Lemma or Woodbury identity applied to the rectangular matrix X . Algebraically: $X^T (XX^T + \lambda I)^{-1} = (X^T X + \lambda I)^{-1} X^T$.*)

(3) Final Answer Box

$$\hat{w}_{kernel} = X^T (XX^T + \lambda I)^{-1} y$$

$$\hat{w}_{primal} = (X^T X + \lambda I)^{-1} X^T y$$

Conclusion: Since the Conditional Expectation (Mean) equals the MAP estimate (Mode) for Gaussians, and the Primal form is the MAP estimate, **the Kernel form and Primal form are identical.**

(4) Intuition / Interpretation

- **Primal vs. Dual:**

- The **Primal form** $(X^T X + \lambda I)^{-1}$ requires inverting a $d \times d$ matrix. This is efficient when we have **many data points but few features** ($n \gg d$).
- The **Kernel form** $(XX^T + \lambda I)^{-1}$ requires inverting an $n \times n$ matrix (the Gram matrix). This is efficient when we have **few data points but many features** ($d \gg n$).
- **Representer Theorem:** The result $w = X^T \alpha$ (where $\alpha = (XX^T + \lambda I)^{-1} y$) shows that the optimal weight vector lies entirely in the span of the training data rows. We don't need to explore the full high-dimensional feature space; we only need to determine how much "influence" α_i each training example x_i has on the solution.

Good!

