



1. Optimizers as Penalized Linear Improvement with different norm penalties

In lecture, you saw the locally linear perspective of a neural network and the loss by Taylor expanding the loss around the current value of the parameters. This approximation is only very good in a near neighborhood of those values. One way to proceed with optimization is to consider the size of the neighborhood as a hyperparameter and to bound our update to stay within that neighborhood while minimizing our linear approximation to the loss. You saw in lecture that the choice of norm in defining that neighborhood also matters.

In this problem, you will work out for yourself a slightly different perspective. Instead of treating the norm as a constraint (with the size of the acceptable norm as a hyperparameter), we can do an unconstrained optimization with a weighted penalty that corresponds to the squared norm — where that weight is a hyperparameter.

At each iteration, we wish to maximize linear improvement of the objective (as defined by the dot product between the gradient and the update) locally regularized by a penalty on the size of the update. This can be expressed (in traditional minimization form) as:

$$u = \underset{\Delta\theta}{\operatorname{argmin}} \underbrace{g^T \Delta\theta}_{\text{Linear Improvement}} + \frac{1}{\alpha} \underbrace{d(\Delta\theta)}_{\text{Distance Penalty}}, \quad (1)$$

where $g = \nabla f(\theta)$ is the gradient of the loss, α is a scalar, and d is a scalar-output distance function $\mathbb{R}^{d(\theta)} \rightarrow \mathbb{R}^+$.

(a) Let's assume *Euclidean distance* is the norm that captures our sense of relevant neighborhoods in parameter space. Then our objective can be:

$$u = \underset{\Delta\theta}{\operatorname{argmin}} g^T \Delta\theta + \frac{1}{\alpha} \|\Delta\theta\|_2^2. \quad (2)$$

What is the analytical solution for u in the above problem? What standard optimizer does this recover?



I'm studying Deep Learning, help me solve the following problems. The first problem is the following:



[Ver razonamiento](#) ▾

Based on the image provided, here is the step-by-step analytical solution for problem (a).

Problem Statement

We want to find the update vector u (which represents $\Delta\theta$) that minimizes the following objective function:

$$u = \underset{\Delta\theta}{\operatorname{argmin}} \left(g^T \Delta\theta + \frac{1}{\alpha} \|\Delta\theta\|_2^2 \right)$$

Where:

- $g = \nabla f(\theta)$ is the gradient.
- $\Delta\theta$ is the parameter update vector.
- α is a scalar hyperparameter.
- $\|\Delta\theta\|_2^2$ is the squared Euclidean norm (L2 norm).

Analytical Solution

To find the minimum, we need to take the derivative of the objective function with respect to $\Delta\theta$, set it to zero, and solve for $\Delta\theta$.

1. Rewrite the objective function in vector notation: Recall that the squared Euclidean norm of a vector can be written as the dot product of the vector with itself: $\|\Delta\theta\|_2^2 = \Delta\theta^T \Delta\theta$.

Let $J(\Delta\theta)$ be our objective function:

$$J(\Delta\theta) = g^T \Delta\theta + \frac{1}{\alpha} \Delta\theta^T \Delta\theta$$

2. Compute the gradient with respect to $\Delta\theta$: We apply vector calculus rules:

- The derivative of $g^T \Delta\theta$ with respect to $\Delta\theta$ is simply g .
- The derivative of $\Delta\theta^T \Delta\theta$ with respect to $\Delta\theta$ is $2\Delta\theta$.

So, the gradient is:

$$\nabla_{\Delta\theta} J(\Delta\theta) = g + \frac{2}{\alpha}(\Delta\theta)$$

3. Set the gradient to zero and solve for $\Delta\theta$: To find the optimum u , we set the gradient to zero:

$$g + \frac{2}{\alpha}\Delta\theta = 0$$

Now, rearrange terms to solve for $\Delta\theta$:

$$\frac{2}{\alpha}\Delta\theta = -g$$

$$\Delta\theta = -\frac{\alpha}{2}g$$

Gemini do it great, also the chain of thought was good.

Therefore, the analytical solution for u is:

$$u = -\frac{\alpha}{2}\nabla f(\theta)$$

What standard optimizer does this recover?

This recovers **Gradient Descent** (or Stochastic Gradient Descent).

Reasoning: The standard update rule for Gradient Descent is:

$$\Delta\theta = -\eta\nabla f(\theta)$$

Where η is the **learning rate**.

Comparing this to our result ($u = -\frac{\alpha}{2}g$), we can see that they are identical if we define the learning rate as $\eta = \frac{\alpha}{2}$. This derivation shows that Gradient Descent can be interpreted as minimizing a local linear approximation of the loss function subject to a penalty on the distance moved (Euclidean distance).

↶ ↷ ⌂ ☐ :

(b) Now, consider an alternative way of capturing local neighborhood size – the squared infinity norm over parameters. Recall that this is defined as $\|x\|_\infty = \max_i |x_i|$. Our objective is now:

$$u = \underset{\Delta\theta}{\operatorname{argmin}} \quad g^T\Delta\theta + \frac{1}{\alpha}\|\Delta\theta\|_\infty^2. \quad (6)$$

What is the analytical solution for u in this case? Which optimizer does this correspond to?



Ver razonamiento ▾

Based on the image provided, here is the step-by-step derivation and solution for problem (b).

Problem Statement

We want to find the update vector u (representing $\Delta\theta$) that minimizes:

$$u = \underset{\Delta\theta}{\operatorname{argmin}} \left(g^T\Delta\theta + \frac{1}{\alpha}\|\Delta\theta\|_\infty^2 \right)$$

Where:

- $\|\Delta\theta\|_\infty = \max_i |\Delta\theta_i|$ (The infinity norm).
- $g = \nabla f(\theta)$ is the gradient.

Analytical Solution

The infinity norm $\max_i |\Delta\theta_i|$ makes direct differentiation difficult because it is not a smooth

function (it contains `max` and absolute values). We can solve this by decomposing the problem into finding the optimal **direction** and the optimal **magnitude**.

1. Decompose the variable: Let $k = \|\Delta\theta\|_\infty$. This is a scalar representing the maximum absolute value of any element in our update vector. Since k is the maximum value, every element $\Delta\theta_i$ must satisfy constraints:

$$-k \leq \Delta\theta_i \leq k$$

2. Optimize the direction for a fixed magnitude k : Let's first minimize the term $g^T \Delta\theta$ assuming k is fixed.

$$g^T \Delta\theta = \sum_i g_i \Delta\theta_i$$

To make this sum as negative as possible (minimize it) given the constraint $|\Delta\theta_i| \leq k$:

- If the gradient component g_i is **positive**, we want $\Delta\theta_i$ to be the smallest possible number: $-k$.
- If the gradient component g_i is **negative**, we want $\Delta\theta_i$ to be the largest possible number: k .

Mathematically, this means we set $\Delta\theta_i$ to be opposite to the sign of the gradient:

$$\Delta\theta_i = -k \cdot \text{sign}(g_i)$$

Substituting this back into the dot product:

$$g^T \Delta\theta = \sum_i g_i (-k \cdot \text{sign}(g_i)) = -k \sum_i g_i \text{sign}(g_i) = -k \sum_i |g_i|$$

Recall that $\sum_i |g_i|$ is the L1 norm of the gradient, denoted as $\|g\|_1$. So, $g^T \Delta\theta = -k \|g\|_1$.

3. Optimize the magnitude k : Now rewrite the original objective function in terms of the scalar k :

$$J(k) = -k \|g\|_1 + \frac{1}{\alpha} k^2$$

To find the optimal k , take the derivative with respect to k and set it to 0:

$$\frac{dJ}{dk} = -\|g\|_1 + \frac{2}{\alpha} k = 0$$

Solving for k :

$$\frac{2}{\alpha} k = \|g\|_1 \implies k = \frac{\alpha}{2} \|g\|_1$$

4. Final Analytical Solution: Substitute the optimal k back into our expression for $\Delta\theta_i$:

$$\Delta\theta_i = -\left(\frac{\alpha}{2} \|g\|_1\right) \text{sign}(g_i)$$

In vector notation, the solution is:

$$u = -\frac{\alpha}{2} \|g\|_1 \text{sign}(g)$$

It is correct and easy to understand.
The proof is also correct.

Which optimizer does this correspond to?

This corresponds to **SignSGD** (or Sign Gradient Descent).

Reasoning:

- The standard update rule for SignSGD is $\Delta\theta = -\eta \cdot \text{sign}(g)$.
- This optimizer ignores the magnitude of the specific gradient components (g_i) and only uses their direction (positive or negative). It moves equal distances in all dimensions, pointing towards the corners of the hypercube defined by the norm.
- While our derived solution includes a scalar scaling factor based on the L1 norm ($\frac{\alpha}{2} \|g\|_1$), the update **direction** is determined entirely by the `sign` function, which is the defining characteristic of SignSGD.

2. Optimizers and their convergence

Consider \mathcal{O} : a simplified Adam-style optimizer without weight decay that has iterates

$$\theta_{t+1} \leftarrow \theta_t - \alpha_t M_t \nabla f_t(\theta_t) \quad (14)$$

where f_t is the loss at iteration t and α_t is the step size (learning rate).

Further suppose that the adaptive scaling matrix, M_t , is recomputed over each epoch of training and just consists of a diagonal populated by the inverses of the square roots of the mean squared value for the gradients during the epoch for that specific coordinate.

For this part, we have exactly $n = 1$ training point corresponding to the single equation

$$[1, 0.1, 0.01]\theta = 1 \quad (15)$$

with a 3-dimensional learnable parameters θ . Suppose that we start with $\theta_0 = 0$ and use squared loss $f_t(\theta) = (1 - [1, 0.1, 0.01]\theta)^2$.

- (a) What specific vector θ would standard vanilla SGD (i.e. (14) with $M_t = I$ and $\alpha_t = \alpha$) converge to assuming $\alpha > 0$ was small enough to give convergence?

Ver razonamiento ▾

Here are the step-by-step solutions for the two problems presented in the images.

Problem 1 (b): Optimization with Squared Infinity Norm

Problem Statement: Find the analytical solution for:

$$u = \underset{\Delta\theta}{\operatorname{argmin}} \left(g^T \Delta\theta + \frac{1}{\alpha} \|\Delta\theta\|_\infty^2 \right)$$

For some reason it starts repeating all the problems.

where $\|\Delta\theta\|_\infty = \max_i |\Delta\theta_i|$.

1. Decompose the problem into Direction and Magnitude The infinity norm looks at the single largest component of the vector. To minimize the dot product $g^T \Delta\theta$ while being penalized by the maximum component size, we should make every component of $\Delta\theta$ contribute as much as possible to reducing the objective, up to the limit allowed by the maximum size.

Let $k = \|\Delta\theta\|_\infty$ be the maximum magnitude (a scalar ≥ 0). To minimize the linear term $g^T \Delta\theta = \sum g_i \Delta\theta_i$ for a fixed max-magnitude k , we should set every coordinate $\Delta\theta_i$ to its maximum allowed amplitude in the direction opposite to the gradient:

$$\Delta\theta_i = -k \cdot \text{sign}(g_i)$$

2. Substitute back to find the optimal magnitude (k) Now we substitute this definition of $\Delta\theta_i$ back into the objective function to solve for the optimal scalar k .

- Linear term:

$$g^T \Delta\theta = \sum_i g_i (-k \cdot \text{sign}(g_i)) = -k \sum_i |g_i| = -k \|g\|_1$$

(Note: $\sum |g_i|$ is the L1 norm of the gradient).

- Penalty term:

$$\frac{1}{\alpha} \|\Delta\theta\|_\infty^2 = \frac{1}{\alpha} k^2$$

The objective function in terms of k is:

$$J(k) = -k\|g\|_1 + \frac{1}{\alpha}k^2$$

Taking the derivative with respect to k and setting to 0:

$$-\|g\|_1 + \frac{2}{\alpha}k = 0 \implies k = \frac{\alpha}{2}\|g\|_1$$

3. Final Analytical Solution Now, substitute optimal k back into the vector expression $\Delta\theta_i = -k \cdot \text{sign}(g_i)$:

$$u = -\frac{\alpha}{2}\|g\|_1\text{sign}(g)$$

Which optimizer does this correspond to? This corresponds to **SignSGD** (Sign Gradient Descent).

While standard SignSGD typically uses a fixed learning rate ($\Delta\theta = -\eta \cdot \text{sign}(g)$), the defining characteristic is using the **sign** of the gradient rather than its magnitude for the update direction. This derivation proves that using the sign of the gradient is optimal when the "cost" of moving in parameter space is measured by the Infinity Norm (hypercube geometry) rather than the Euclidean Norm (spherical geometry).

Problem 2 (a): Convergence of Vanilla SGD

Problem Statement: We have a single training point equation $[1, 0.1, 0.01]\theta = 1$ with squared loss. We use standard SGD with initialization $\theta_0 = \mathbf{0}$. Find the specific vector θ that SGD converges to.

1. Analyze the Gradient Direction Let the input vector be $x = [1, 0.1, 0.01]^T$. The loss function is $L(\theta) = (1 - x^T\theta)^2$. The gradient is:

$$\nabla L(\theta) = -2(1 - x^T\theta)x$$

Notice that the gradient is **always** a scalar multiple of the input vector x . Since we start at $\theta_0 = \mathbf{0}$ and every update step subtracts a multiple of the gradient, the parameter vector θ will never leave the linear span of x . Therefore, the final solution θ^* must be of the form:

$$\theta^* = c \cdot x$$

for some scalar constant c .

2. Satisfy the Convergence Condition SGD converges when the gradient is zero (or the loss is zero, since it's a realizable single equation).

$$1 - x^T\theta^* = 0 \implies x^T\theta^* = 1$$

Substitute $\theta^* = c \cdot x$ into this equation:

$$x^T(c \cdot x) = 1$$

$$c(x^T x) = 1$$

$$c\|x\|_2^2 = 1 \implies c = \frac{1}{\|x\|_2^2}$$

3. Calculate the specific values First, calculate the squared norm of x :

$$\|x\|_2^2 = 1^2 + (0.1)^2 + (0.01)^2$$

$$\|x\|_2^2 = 1 + 0.01 + 0.0001 = 1.0101$$

So, the constant c is:

$$c = \frac{1}{1.0101}$$

The final vector θ^* is:

$$\theta^* = \frac{1}{1.0101} \begin{bmatrix} 1 \\ 0.1 \\ 0.01 \end{bmatrix} \approx \begin{bmatrix} 0.9900 \\ 0.0990 \\ 0.0099 \end{bmatrix}$$

The answer is correct and the explanation is really easy to understand.

Answer: The specific vector is:

$$\theta = \frac{1}{1.0101} [1, 0.1, 0.01]^T$$

(This is the "minimum norm solution" to the linear system).

(b) What specific vector θ would the simplified version of Adam C converge to assuming appropriate step-sizes $\alpha_t > 0$ to give convergence?

◆ Ver razonamiento ▾

Here are the solutions for the two problems shown in your screenshots.

Problem 1 (b): Analytical Solution for Squared Infinity Norm

Problem Statement: Find the analytical solution for:

$$u = \underset{\Delta\theta}{\operatorname{argmin}} \left(g^T \Delta\theta + \frac{1}{\alpha} \|\Delta\theta\|_\infty^2 \right)$$

It continues repeating previous problems.

where $\|\Delta\theta\|_\infty = \max_i |\Delta\theta_i|$.

Step 1: Decompose Direction and Magnitude The infinity norm is determined by the single largest component of the vector. To minimize the objective, we must align every component of the update vector $\Delta\theta$ such that it reduces the dot product $g^T \Delta\theta$ as much as possible, up to the maximum allowed limit.

Let k be the maximum magnitude of any component (i.e., $k = \|\Delta\theta\|_\infty$). To minimize the linear term $\sum g_i \Delta\theta_i$ subject to $|\Delta\theta_i| \leq k$, we set each component to its limit in the opposite direction of the gradient:

$$\Delta\theta_i = -k \cdot \operatorname{sign}(g_i)$$

Step 2: Solve for the optimal Magnitude (k) Substitute this expression back into the objective function:

1. Linear Term:

$$g^T \Delta\theta = \sum_i g_i (-k \cdot \operatorname{sign}(g_i)) = -k \sum_i |g_i| = -k \|g\|_1$$

$$\frac{1}{\alpha} \|\Delta\theta\|_\infty^2 = \frac{1}{\alpha} k^2$$

Now minimize the scalar function $J(k)$:

$$J(k) = -k\|g\|_1 + \frac{1}{\alpha} k^2$$

Take the derivative w.r.t k and set to 0:

$$-\|g\|_1 + \frac{2}{\alpha} k = 0 \implies k = \frac{\alpha}{2} \|g\|_1$$

Step 3: Final Solution Substitute the optimal k back into the vector expression:

$$u = -\frac{\alpha}{2} \|g\|_1 \text{sign}(g)$$

Which optimizer does this correspond to? This corresponds to **SignSGD** (Sign Gradient Descent). While standard SignSGD typically uses a fixed step size, the defining characteristic is that the update direction is determined solely by the sign of the gradient components ($\text{sign}(g)$), not their magnitudes.

Problem 2 (b): Convergence of Simplified Adam

Problem Statement: We have the equation $[1, 0.1, 0.01]\theta = 1$. The optimizer updates are $\theta_{t+1} \leftarrow \theta_t - \alpha_t M_t \nabla f_i(\theta)$. M_t is a diagonal matrix where entries are the inverse of the absolute value of the gradients (since $n = 1$, the "root mean square" of a single value is just its absolute value).

Step 1: Analyze the Update Direction Let the input vector be $x = [1, 0.1, 0.01]^T$. The gradient is $\nabla f(\theta) \propto x$. Let's call the gradient vector g . The components are $g_i \propto x_i$.

The optimizer applies the matrix M_t , which scales each component by $1/|g_i|$. The effective update vector u is:

$$u_i = -(M_t g)_i = -\frac{1}{|g_i|} g_i = -\text{sign}(g_i)$$

Since the input vector $x = [1, 0.1, 0.01]^T$ has all positive components, the gradient components g_i will all have the same sign (either all positive or all negative, depending on the current error). Therefore, the update vector will always be proportional to:

$$\Delta\theta \propto [1, 1, 1]^T$$

Step 2: Determine the Convergence Path Since we initialize at $\theta_0 = \mathbf{0}$ and every update moves equal amounts in all dimensions (along the vector $[1, 1, 1]^T$), the final parameter vector θ^* must lie on the line:

$$\theta^* = c \cdot [1, 1, 1]^T$$

for some scalar c .

Step 3: Solve for the Solution Vector The algorithm converges when the loss is 0, satisfying the equation:

$$x^T \theta^* = 1$$

Substitute $\theta^* = c[1, 1, 1]^T$.

$$[1, 0.1, 0.01] \cdot \begin{bmatrix} c \\ c \\ c \end{bmatrix} = 1$$

$$c(1 + 0.1 + 0.01) = 1$$

$$c(1.11) = 1$$

$$c = \frac{1}{1.11}$$

Answer: The specific vector θ is:

$$\theta = \frac{1}{1.11} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \approx \begin{bmatrix} 0.9009 \\ 0.9009 \\ 0.9009 \end{bmatrix}$$

The answer is correct and the explanation is really easy to understand. I like how it tries to give the answer in both possible notations.

Note: This contrasts with Vanilla SGD (part a), which would converge to a vector proportional to $[1, 0.1, 0.01]$. This illustrates how Adam-style optimizers normalize the update steps, ensuring features with smaller gradients (like the 0.01 component) learn just as fast as features with large gradients.

(c) Consider a learning approach that first did training input feature rescaling (so that each feature had unit second-moment), then ran SGD to convergence, and then converted the solution for the rescaled problem back to the original units. What specific vector θ would it give as its final solution (for use in original coordinates)?

Ver razonamiento ▾

Based on the images provided, here are the step-by-step solutions for the problems.

Problem 2(c): Feature Rescaling + SGD

Problem Statement: We perform feature rescaling so each feature has a "unit second-moment" (magnitude of 1), run SGD to convergence, and then convert the solution back to original units. Original Equation: $[1, 0.1, 0.01]\theta = 1$. Initialization: $\theta_0 = \mathbf{0}$.

Step 1: Rescale the Inputs We need to scale the input vector $x = [1, 0.1, 0.01]^T$ into a new vector \tilde{x} such that every component has a magnitude of 1 ($\tilde{x}_i^2 = 1$).

- For $x_1 = 1$: Scale by 1 $\rightarrow \tilde{x}_1 = 1$.
- For $x_2 = 0.1$: Scale by 10 $\rightarrow \tilde{x}_2 = 1$.
- For $x_3 = 0.01$: Scale by 100 $\rightarrow \tilde{x}_3 = 1$.

The scaling matrix S (diagonal) is:

$$S = \text{diag}(1, 10, 100)$$

The rescaled input is $\tilde{x} = [1, 1, 1]^T$.

Step 2: Solve SGD in the Rescaled Space We run SGD on the problem $\tilde{x}^T \tilde{\theta} = 1$ starting from zero. SGD moves in the direction of the input vector \tilde{x} . Thus, the solution $\tilde{\theta}^*$ will be proportional to $\tilde{x} = [1, 1, 1]^T$.

$$\tilde{\theta}^* = c \cdot [1, 1, 1]^T$$

To satisfy the convergence condition $x^T \theta^* = 1$:

$$[1, 1, 1] \cdot \begin{bmatrix} c \\ c \\ c \end{bmatrix} = 1$$

$$3c = 1 \implies c = \frac{1}{3}$$

So, the solution in the rescaled space is:

$$\tilde{\theta}^* = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

Step 3: Convert Back to Original Units The relationship between the predictions in both spaces is $x^T \theta = \tilde{x}^T \tilde{\theta}$. Since $\tilde{x} = Sx$, we have $(Sx)^T \tilde{\theta} = x^T S \tilde{\theta}$. This implies that $\theta = S \tilde{\theta}$.

We multiply our rescaled solution by the scaling factors:

$$\theta^* = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 100 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix} = \begin{bmatrix} 1/3 \\ 10/3 \\ 100/3 \end{bmatrix}$$

The answer is correct and the explanation is really easy to understand. It correctly write it on the notation expected.

Final Answer for 2(c):

$$\theta = \begin{bmatrix} 0.333... \\ 3.333... \\ 33.333... \end{bmatrix}$$

Problem 2(b): Simplified Adam Convergence

Problem Statement: Find the convergence vector for the simplified Adam optimizer \mathcal{O} on $[1, 0.1, 0.01]\theta = 1$. The optimizer updates are scaled by the inverse of the gradient magnitudes.

Step 1: Determine the Update Direction The gradient is proportional to the input $x = [1, 0.1, 0.01]^T$. The simplified Adam optimizer divides each gradient component by its absolute value (effectively taking the sign).

$$u \propto \text{sign}(x) = [1, 1, 1]^T$$

Since we start at 0, the parameter vector θ will always lie on the line defined by direction $[1, 1, 1]^T$.

Step 2: Find the Intersection with the Solution Plane We need to find a scalar c such that $\theta^* = c[1, 1, 1]^T$ satisfies the equation $[1, 0.1, 0.01]\theta = 1$.

$$1(c) + 0.1(c) + 0.01(c) = 1$$

$$1.11c = 1$$

$$c = \frac{1}{1.11} \approx 0.9009$$

Final Answer for 2(b):

$$\theta = \frac{1}{1.11} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \approx \begin{bmatrix} 0.9009 \\ 0.9009 \\ 0.9009 \end{bmatrix}$$

Problem 1(b): Squared Infinity Norm Analytical Solution

Problem Statement: Find the analytical solution for $u = \operatorname{argmin}_{\Delta\theta} (g^T \Delta\theta + \frac{1}{\alpha} \|\Delta\theta\|_\infty^2)$.

Solution: The squared infinity norm penalty forces the optimizer to limit the **maximum** component magnitude. To minimize the dot product $g^T \Delta\theta$, every component $\Delta\theta_i$ should move in the direction opposite to the gradient ($-\operatorname{sign}(g_i)$) with the same magnitude k .

1. **Direction:** $\Delta\theta_i \propto -\operatorname{sign}(g_i)$.
2. **Magnitude:** By minimizing the scalar function with respect to the magnitude k , we find $k = \frac{\alpha}{2} \|g\|_1$.

Final Answer for 1(b): The analytical solution is:

$$u = -\frac{\alpha}{2} \|g\|_1 \operatorname{sign}(g)$$

This corresponds to the **SignSGD** optimizer.



Ver razonamiento ▾

Based on the images provided, here are the solutions. I will start with **Problem 5** as it involves the table you provided, followed by the solutions for **Problem 2(b)** and **2(c)**.

Problem 5: Analyzing Distributed Training

You need to fill in the missing values for the "Number of Messages Sent" and "Size of each message" in the table.

Assumptions:

- n = number of workers.
- p = number of parameters (size of the model).

1. All-to-All Communication

- **Context:** Each worker communicates with every other worker.
- **Number of Messages Sent:** Each of the n workers sends a message to the other $n - 1$ workers.
 - Calculation: $n \times (n - 1)$.
- **Size of each message:** The table indicates the size is p . This is because each worker sends its full gradient vector to the others.

2. Parameter Server

- **Context:** Workers send gradients to a server; the server sends updated weights back.
- **Number of Messages Sent:** The table fills this as $2n$ (1 upload + 1 download per worker).
- **Size of each message:**
 - **Upload:** Each worker sends its gradients. Size = p .
 - **Download:** The server sends the updated model weights. Size = n .



- Download: the server sends the updated model weights. size = p .

- Answer: p .

3. Ring All-Reduce

- **Context:** Workers are in a ring. They split gradients into n chunks and pass them around.
- **Number of Messages Sent:** The table fills this as $n(2(n - 1))$.
- **Size of each message:**
 - The problem description states: "splits the gradients into n equally sized chunks and sends a chunk..."
 - Since the total size is p , one chunk has size p/n .
- **Answer:** p/n .

Completed Table Summary

Paradigm	Number of Messages Sent	Size of each message
All-to-All	$n(n - 1)$	p
Parameter Server	$2n$	p
Ring All-Reduce	$n(2(n - 1))$	p/n

 Exportar a Hojas de cálculo 

The chain of thought was perfect and the answers correct as expected in an easy exercise like this one. It gets really good the instruction through the images.

Problem 2 (b): Convergence of Simplified Adam

Problem Statement: Find the specific vector θ that the simplified Adam optimizer \mathcal{O} converges to for the equation $[1, 0.1, 0.01]\theta = 1$. The optimizer scales updates by the inverse of the absolute gradient values.

Solution:

[Continue repiting previous questions.](#)

1. **Determine Update Direction:** The gradient vector g is proportional to the input $x = [1, 0.1, 0.01]^T$. The optimizer normalizes each component by its magnitude $(1/|g_i|)$. This means the update direction is effectively the **sign** of the gradient:

$$u \propto \text{sign}(x) = [1, 1, 1]^T$$

Since initialization is $\theta_0 = 0$, the trajectory of θ is constrained to the line $\theta = c \cdot [1, 1, 1]^T$.

2. **Find Intersection:** We need to find the scalar c where this line intersects the solution plane $x^T\theta = 1$.

$$[1, 0.1, 0.01] \cdot \begin{bmatrix} c \\ c \\ c \end{bmatrix} = 1$$

$$c(1 + 0.1 + 0.01) = 1$$

$$1.11c = 1 \implies c = \frac{1}{1.11}$$

Final Answer:

$$\theta = \frac{1}{1.11} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \approx \begin{bmatrix} 0.9009 \\ 0.9009 \\ 0.9009 \end{bmatrix}$$

Problem 2 (c): Feature Rescaling + SGD

Problem Statement: Rescale features to have "unit second-moment" (magnitude of 1), run SGD to convergence, and convert back to original units. Input $x = [1, 0.1, 0.01]$.

Solution:

1. **Rescale Inputs (\tilde{x}):** We scale each component x_i by a factor α_i so that $(\alpha_i x_i)^2 = 1$.
 - $x_1 = 1 \rightarrow \alpha_1 = 1$
 - $x_2 = 0.1 \rightarrow \alpha_2 = 10$
 - $x_3 = 0.01 \rightarrow \alpha_3 = 100$ Rescaled input: $\tilde{x} = [1, 1, 1]^T$.
2. **Solve in Rescaled Space ($\tilde{\theta}$):** Standard SGD moves in the direction of the input. For $\tilde{x} = [1, 1, 1]^T$, the solution $\tilde{\theta}$ will be of the form $k[1, 1, 1]^T$. Condition: $\tilde{x}^T \tilde{\theta} = 1 \implies 1(k) + 1(k) + 1(k) = 1 \implies 3k = 1 \implies k = 1/3$.
$$\tilde{\theta} = [1/3, 1/3, 1/3]^T$$
3. **Convert Back to Original Units (θ):** The relationship between the variables is $\theta_i = \alpha_i \tilde{\theta}_i$ (to maintain the dot product invariance $x_i \theta_i = \tilde{x}_i \tilde{\theta}_i$).
 - $\theta_1 = 1 \cdot (1/3) = 1/3$
 - $\theta_2 = 10 \cdot (1/3) = 10/3$
 - $\theta_3 = 100 \cdot (1/3) = 100/3$

Final Answer:

$$\theta = \begin{bmatrix} 1/3 \\ 10/3 \\ 100/3 \end{bmatrix} = \begin{bmatrix} 0.333... \\ 3.333... \\ 33.333... \end{bmatrix}$$

↶ ↷ ⌂ ↲ ☐ :

Pregunta a Gemini

+  Herramientas

Razonamiento 



Gemini puede cometer errores, así que verifica sus respuestas.

