

Special Participation A



hw2.pdf
PDF


You are a deep learning tutor. This homework has already been released, but we are going to evaluate your capabilities for one-shot questions. I will guide you towards the correct answer should you make a mistake.

Please solve the written questions on this worksheet: 1, 2, and 5. Answer one question one response by response (with all its subparts).



Thought for 1m 33s >

Question 1 (Optimizers as Penalized Linear Improvement)

We're solving, at each step, an update u (i.e., $\Delta\theta^*$) that minimizes a **linearized loss** plus a **distance penalty**. 

(a) Euclidean (squared ℓ_2) penalty

Problem:

$$u = \arg \min_{\Delta\theta} g^\top \Delta\theta + \frac{1}{\alpha} \|\Delta\theta\|_2^2.$$

This is a convex quadratic. Take gradient w.r.t. $\Delta\theta$ and set to zero:

$$\begin{aligned} \nabla_{\Delta\theta} \left(g^\top \Delta\theta + \frac{1}{\alpha} \|\Delta\theta\|_2^2 \right) &= g + \frac{2}{\alpha} \Delta\theta = 0 \\ \Rightarrow \Delta\theta^* &= -\frac{\alpha}{2} g. \end{aligned}$$

So the update is:

$$u = -\frac{\alpha}{2} g.$$

Optimizer recovered: vanilla gradient descent / SGD step (just with learning rate $\alpha/2$).

(b) ℓ_∞ penalty hw2

Problem as written:

$$u = \arg \min_{\Delta\theta} g^\top \Delta\theta + \frac{1}{\alpha} \|\Delta\theta\|_\infty.$$

Let $r = \|\Delta\theta\|_\infty$. For a fixed r , the best (most loss-decreasing) choice is to push every coordinate to the boundary with the sign opposite the gradient:

$$\Delta\theta_i = -r \operatorname{sign}(g_i),$$

which gives

$$g^\top \Delta\theta = -r \sum_i |g_i| = -r \|g\|_1, \quad \|\Delta\theta\|_\infty = r.$$

So the objective becomes

$$\phi(r) = -r \|g\|_1 + \frac{1}{\alpha} r = r \left(\frac{1}{\alpha} - \|g\|_1 \right).$$

From this:

- If $\|g\|_1 < \frac{1}{\alpha}$, the minimum is at $r = 0 \Rightarrow u = 0$.
- If $\|g\|_1 > \frac{1}{\alpha}$, the objective is unbounded below as $r \rightarrow \infty$ (no finite minimizer).
- If $\|g\|_1 = \frac{1}{\alpha}$, any $r \geq 0$ with $\Delta\theta = -r \operatorname{sign}(g)$ ties.

Direction-wise, the solution (when it's finite/meaningful) is the **sign direction** $-\operatorname{sign}(g)$, i.e. the steepest-descent direction under an ℓ_∞ notion of step size—this is the idea behind **signSGD / sign-based updates**.

(If the intended penalty was the squared version $\frac{1}{\alpha} \|\Delta\theta\|_\infty^2$ to guarantee a finite optimum, then you'd get $r^ = \frac{\alpha}{2} \|g\|_1$ and $u_i = -\frac{\alpha}{2} \|g\|_1 \operatorname{sign}(g_i)$, which is an explicit sign-based optimizer.)*

– **Document parser reads off L1 penalty instead of L2 penalty, but catches ambiguity and provides correct result anyways**

If you want, I'll move on to **Question 2** next (all subparts) in the next message.



↪ $\alpha 1 \|\Delta\theta\|_\infty$

Correct! I think you had a parsing error here, because this indeed was the desired penalty. However, you got the right answer in the end. Let's move on to Question 2.



Thought for 1m 4s >

Question 2 (Optimizers and their convergence) hw2

Let $a = [1, 0.1, 0.01]^\top$ and $f(\theta) = (1 - a^\top \theta)^2$. hw2

Then $\nabla f(\theta) = -2(1 - a^\top \theta) a$.

(a) Vanilla SGD ($M_t = I$, $\alpha_t = \alpha$) hw2

From $\theta_0 = 0$, every update is proportional to a , so $\theta_t \in \text{span}(a)$. Write $\theta_t = c_t a$. At convergence we need $a^\top \theta = 1$, i.e.

$$a^\top (ca) = c \|a\|_2^2 = 1 \quad \Rightarrow \quad c = \frac{1}{\|a\|_2^2}.$$

$$\|a\|_2^2 = 1^2 + 0.1^2 + 0.01^2 = 1.0101.$$

So SGD converges to

$$\theta_{\text{SGD}}^* = \frac{1}{1.0101} [1, 0.1, 0.01]^\top \approx [0.99000099, 0.099000099, 0.0099000099]^\top.$$

(b) Simplified Adam O hw2



Here $n = 1$ example per epoch, so the “mean squared gradient over the epoch” for coordinate i is just $g_{t,i}^2$. Thus

$$M_t = \text{diag} \left(\frac{1}{|g_{t,1}|}, \frac{1}{|g_{t,2}|}, \frac{1}{|g_{t,3}|} \right) \Rightarrow M_t g_t = \text{sign}(g_t) \text{ (elementwise).}$$

But $g_t = -2(1 - a^\top \theta_t)a$, and all entries of a are positive, so all coordinates share the same sign. Therefore the iterate stays on the line $\theta_t = v_t[1, 1, 1]^\top$. At convergence we must satisfy $a^\top \theta = 1$:

$$v(1 + 0.1 + 0.01) = v \cdot 1.11 = 1 \Rightarrow v = \frac{1}{1.11}.$$

So (with step sizes α_t chosen to actually converge) Adam O converges to

$\boxed{\theta^* = \frac{1}{1.11}[1, 1, 1]^\top \approx [0.9009009, 0.9009009, 0.9009009]^\top}$:contentReference[oaicite:4]{index=4} --- ##### (c) Feature rescale to unit second moment \rightarrow SGD \rightarrow map back :contentReference[oaicite:5]{index=5} With one training point, the (empirical) second moment of feature (i) is (a_i^2) . Rescaling to unit second moment means divide feature (i) by $(|a_i|)$, giving a rescaled feature vector $z = \left[\frac{1}{1}, \frac{0.1}{0.1}, \frac{0.01}{0.01} \right]^\top = [1, 1, 1]^\top$.

Run SGD from 0 on $(1 - z^\top \phi)^2$. By the same logic as part (a), it converges to

$$\phi^* = \frac{1}{\|z\|_2^2} z = \frac{1}{3}[1, 1, 1]^\top. \text{ LaTeX error! But correct result.}$$

Mapping back to original coordinates uses $\phi_i = |a_i|\theta_i$ (here $a_i > 0$), so $\theta_i = \phi_i/a_i$:

$$\theta^* = \left[\frac{1/3}{1}, \frac{1/3}{0.1}, \frac{1/3}{0.01} \right]^\top = \left[\frac{1}{3}, \frac{10}{3}, \frac{100}{3} \right]^\top \approx [0.3333, 3.3333, 33.3333]^\top.$$

If you want, I'll do **Question 5** next (all its subparts) in the next response.



↳ $\boxed{\theta_0^* = \frac{1}{1.11} [1, 1, 1]^T \approx [0.9009009, 0.9009009, 0.9009009]^T}$:contentReference[oaicite:4]{index=4}
 --- ##### (c) Feature rescale to unit second moment → SGD → map back :contentReferenc...

You had a LaTeX error! But your answers are otherwise correct. Can you rewrite your answer for 2) before we move on?



Question 2 (rewritten cleanly)

Given

$$a = \begin{bmatrix} 1 \\ 0.1 \\ 0.01 \end{bmatrix}, \quad f(\theta) = (1 - a^\top \theta)^2,$$

the gradient is

$$\nabla f(\theta) = -2(1 - a^\top \theta) a.$$

(a) Vanilla SGD ($M_t = I$, $\alpha_t = \alpha$)

Starting from $\theta_0 = 0$, every update is a scalar multiple of a , so $\theta_t \in \text{span}(a)$. Write $\theta_t = c_t a$. At convergence, we need $a^\top \theta = 1$, so

$$a^\top (ca) = c \|a\|_2^2 = 1 \quad \Rightarrow \quad c = \frac{1}{\|a\|_2^2}.$$

Compute

$$\|a\|_2^2 = 1^2 + 0.1^2 + 0.01^2 = 1 + 0.01 + 0.0001 = 1.0101.$$

Therefore

$$\theta_{\text{SGD}}^* = \frac{1}{1.0101} \begin{bmatrix} 1 \\ 0.1 \\ 0.01 \end{bmatrix} \approx \begin{bmatrix} 0.99000099 \\ 0.099000099 \\ 0.0099000099 \end{bmatrix}.$$

(b) Simplified Adam \mathcal{O}

With one data point per epoch, the “mean squared gradient over the epoch” in coordinate i is just $g_{t,i}^2$, so

$$M_t = \text{diag}\left(\frac{1}{|g_{t,1}|}, \frac{1}{|g_{t,2}|}, \frac{1}{|g_{t,3}|}\right),$$

which implies elementwise

$$(M_t g_t)_i = \frac{g_{t,i}}{|g_{t,i}|} = \text{sign}(g_{t,i}).$$

But

$$g_t = -2(1 - a^\top \theta_t) a,$$

and all entries of a are positive, so all coordinates of g_t share the same sign. Hence each step moves along the $[1, 1, 1]^\top$ direction, meaning $\theta_t = v_t \mathbf{1}$ with $\mathbf{1} = [1, 1, 1]^\top$. At convergence we need $a^\top \theta = 1$:

$$a^\top (v \mathbf{1}) = v(1 + 0.1 + 0.01) = v \cdot 1.11 = 1 \quad \Rightarrow \quad v = \frac{1}{1.11}.$$

So

$$\theta_O^* = \frac{1}{1.11} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \approx \begin{bmatrix} 0.9009009 \\ 0.9009009 \\ 0.9009009 \end{bmatrix}.$$

(c) Rescale features to unit second moment, run SGD, then map back

With one example, the empirical second moment of feature i is a_i^2 . To make each feature have unit second moment, rescale coordinate i by dividing by $|a_i|$. The rescaled feature vector is

$$z = \begin{bmatrix} 1/1 \\ 0.1/0.1 \\ 0.01/0.01 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}.$$

Run SGD from 0 on $(1 - z^\top \phi)^2$. By the same reasoning as (a), it converges to

$$\phi^* = \frac{1}{\|z\|_2^2} z = \frac{1}{3} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}.$$

Mapping back: since we scaled features by $z_i = a_i/|a_i|$ and here $a_i > 0$, the parameter relationship is $\phi_i = a_i\theta_i$, so $\theta_i = \phi_i/a_i$. Therefore

$$\theta^* = \begin{bmatrix} (1/3)/1 \\ (1/3)/0.1 \\ (1/3)/0.01 \end{bmatrix} = \begin{bmatrix} 1/3 \\ 10/3 \\ 100/3 \end{bmatrix} \approx \begin{bmatrix} 0.3333 \\ 3.3333 \\ 33.3333 \end{bmatrix}.$$

Ready for Question 5.



Looks good. Let's move on to Question 5, then.



Thought for 45s >

Question 5 (Analyzing Distributed Training)

hw2

Assume n workers, model has p parameters, and p divisible by n . The filled-in communication costs per iteration are:

Paradigm	Total # messages sent	Size of each message
All-to-All	$n(n - 1)$	p
Parameter Server	$2n$	p
Ring All-Reduce	$n \cdot 2(n - 1)$	p/n

Explanation (one-liners):

- **All-to-All:** each worker sends its full gradient (size p) to each of the other $n - 1$ workers $\Rightarrow n(n - 1)$ sends total.
- **Parameter Server:** each worker sends a gradient to the server and receives updated weights back $\Rightarrow 2n$ messages, each of size p .
- **Ring All-Reduce:** gradients are split into n chunks of size p/n ; there are $2(n - 1)$ passes and each worker sends one chunk per pass $\Rightarrow n \cdot 2(n - 1)$ messages of size p/n .

hw2



Perfect!



Nice — if you have any follow-ups (e.g., deriving bandwidth/time per iteration under a network model, or comparing asymptotics), throw them at me.