Outcome Prediction Quiz: Optimization 2 (Solutions)

CS 312 · 130 points

Answer sheet. Write your Name and SUNet ID, mark every answer and write your rationale on the separate answer sheet at the end of this booklet (pages S1–S2). **Only the answer sheet is graded.**

Instructions. Predict the outcomes of the following experiments. Each question will have 3 instructions

- **From the assignment** provides some context, providing any needed information from the assignment.
- **This question changes** provides a natural language description of what is different from the assignment.
- **The diff** is the exact code that we run when creating the solution key (printed in Appendix A). It is a patch to the released A3 starter (`a3_release` at `57c60bb`).

Check one box in every row of the answer sheet. Options that are given as sketches are judged on matching shape; their axis values show rough magnitudes. **Scoring:** In rows whose options are ordinal choices, the adjacent option will receive half the maximal points. Each question has three rationale fields: the assignment problem you cite, what you observed in that experiment, and your rationale for choosing your answer based on those observations. Detailed proper citation, with experimental details and sound rationale can return 75% of missed points on a question.

Default setup. The standard setup of the Optimization 2 assignment: Llama-style architecture with 8 layers, hidden size 512, 8 attention heads, QK-norm, a 4k vocab size tokenizer, and about 35M parameters (`d8`); 614.4M training tokens (9,375 updates) of fresh data; AdamW with $\beta_1, \beta_2 = (0.9, 0.95)$ and $\epsilon = 10^{-8}$, linear decay learning rate schedule with 1% linear warmup (93 updates), peak learning rate .003, context length 1024, batch size 64, weight decay 0.1 (masked: hidden and readout matrices decay; embeddings and normalization gains do not), and 1.0 grad clipping; model and data seed 42. Lower validation loss is better. Seed-to-seed noise in the final validation loss is about .003.

Conventions. We follow the same conventions as the rest of A3.

- `wsd`X **schedule**: after the warmup, the peak LR is held constant and then decays linearly to zero over the final fraction X of training (`wsd0.2` decays from update 7,500).
- **Regret** (Problem 2(c)): a run's validation loss minus the control run's at the same update.
- **Sharpness**: $\lambda_{\max}$ is the top eigenvalue of the loss Hessian, measured in FP32 on the 64-sequence held-out probe. A **family block** is the Hessian restricted to one parameter family (embedding, attention, MLP, RMSNorm gains, LM head), as in Problem 4(a).
- **Full-batch runs** (Problem 3(a)): plain gradient descent on the first N training sequences, embedding initialized 16× larger, no clipping, no weight decay, constant LR η.

Prediction

Q1. Basins

Difficulty: Easy

20 points

From the assignment (reference only). Recall the forked-run sweep of Problem 1: the default recipe trains a LM at a constant LR of .003 after the warmup. A shared prefix trains for x updates from model seed 42, then two branches continue for y more updates with different data orders (`batch_order_seed` 1 and 2). For each cell we compare three validation losses: *Best*, the better of the two final losses; *Ens*, the 0.5/0.5 prediction ensemble of the two endpoints; and *Avg*, the 0.5/0.5 weight average.

The diff behind this question is Appendix A.1, page 19.

(a) Average, ensemble, or neither, at a higher LR (12 points, 4 per cell).

This question changes. The whole sweep (prefix and both branches) is repeated at a constant LR of **.01** instead of .003, with the same warmup, model seed and data orders.

At LR .003 (Problem 1's grid) the three cells below came out as shown. For each cell, which ordering, lowest (best) first, do you predict at LR .01? Two losses count as equal (=) when within .01.

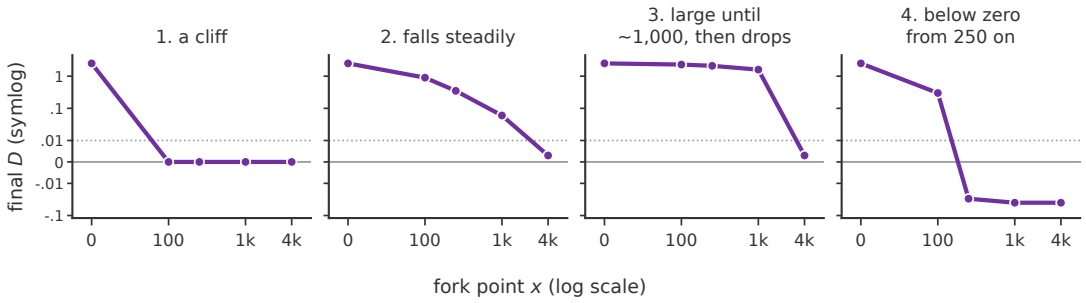
Cell	x	y	At LR .003
(i)	100	50	Avg = Ens < Best
(ii)	100	250	Ens < Avg < Best
(iii)	250	250	Ens < Avg < Best

· 1. Avg < Ens < Best
· 2. Avg = Ens < Best
· 3. Ens < Avg < Best
· 4. Ens < Best < Avg
· 5. Ens < Avg = Best

(b) How much shared training is enough? (8 points).

This question changes. The **full default recipe** (peak LR .003, linear decay to zero, 9,375 updates) is trained with `batch_order_seed` 1 from model seed 42. Second runs **fork from its checkpoint** (weights and optimizer state) **at update** $x \in \{0, 100, 250, 1{,}000, 4{,}000\}$ ($x = 0$: a fresh run from the same init) and train the remaining $y = 9{,}375 - x$ updates with `batch_order_seed` 2. All runs end at a validation loss of about 2.93.

Let $D(x) = L(\text{Avg}) - \min(L_1, L_2)$ for the two final models. Which sketch of D against the fork point x (log axis) matches? The D axis is log scale, linear within $\pm.01$; the dotted line marks $D = .01$.



· 1. · 2. · 3. · 4.

Solution. (a) (i) 2 (Avg = Ens < Best), (ii) 4 (Ens < Best < Avg), (iii) 3 (Ens < Avg < Best). (b) 2 (falls steadily, still above .01 at $x = 1{,}000$).

(a) A higher LR shrinks the connected region; inside it, nothing changes. The sweep at a constant LR of .01 on the release starter (seeds 42 / 43), with the LR .003 cells for reference (Avg and Ens themselves are Best plus the two differences):

Cell	LR	Best	Avg − Best	Avg − Ens	Ordering
(i) $x = 100$, $y = 50$	.003	4.670 / 4.685	−.086 / −.088	−.003 / −.005	Avg = Ens < Best
	.01	4.892 / 4.918	−.091 / −.086	−.004 / .000	Avg = Ens < Best
(ii) $x = 100$, $y = 250$	.003	3.991 / 3.993	−.043 / −.037	+.053 / +.055	Ens < Avg < Best
	.01	4.308 / 4.314	+.044 / +.057	+.136 / +.159	Ens < Best < Avg
(iii) $x = 250$, $y = 250$	.003	3.734 / 3.741	−.075 / −.076	+.012 / +.014	Ens < Avg < Best
	.01	3.979 / 3.942	−.045 / −.059	+.041 / +.033	Ens < Avg < Best

At .01 the branches take 3.3× larger steps, so two data orders separate faster. The barrier that at .003 appears only for $x \leq 100$ with $y = 500$ now appears already at $(100, 250)$: the average is .04–.06 worse than the better endpoint, cell (ii) flips from option 3 to option 4. Over the whole 5×5 sweep (branches of up to 1,000 updates) the number of cells in which the average beats the better endpoint by at least .01 falls from 19 to 17, with $(250, 500)$ moving to the edge, and no cell is gained. Inside the connected region the picture is the one from Problem 1: with 50-update branches the average matches the ensemble, cell (i), and after 250 updates the ensemble pulls ahead while the average still beats both endpoints, cell (iii); the larger LR only widens the Avg-versus-Ens gap there (.012–.014 to .033–.041). The ensemble beats the better endpoint by .08–.10 in every cell at both LRs. Problem 1(d): the learning rate is the strongest knob for early forks, since it sets how far the two copies travel per update. Both seeds give the same three orderings; the closest call is cell (i), where Avg − Ens is .000 at seed 43 against the .01 line.

(b) A full run needs far more shared training than a 500-update branch. Final losses of the two runs and their weight average and ensemble (release starter; seeds 42 / 43):

Fork point x	$\min(L_1, L_2)$	$D(x)$	Ens minus $\min(L_1, L_2)$
0 (same init)	2.927 / 2.930	+2.56 / +2.82	−.050 / −.050
100	2.927 / 2.930	+.82 / +.92	−.045 / −.046
250	2.927 / 2.930	+.36 / +.36	−.042 / −.042
1,000	2.928 / 2.931	+.057 / +.057	−.032 / −.032
4,000	2.928 / 2.931	−.004 / −.004	−.017 / −.017

In Problem 1's grid, forks at $x \geq 250$ stay linearly connected for 500 more updates, and the barrier grows with the branch length y and shrinks with the shared prefix x. Here the second run trains $y = 9{,}375 - x$, 5,375 to 9,375 updates, on its own, so the boundary moves to much larger x: D falls steadily with x, is still +.057 at $x = 1{,}000$, and only the fork at 4,000 averages to a tie. Even there the average does not beat the endpoints. The decay to zero has already removed the across-valley noise that averaging cancels at constant LR (Problem 1(d), cooldown branches). The ensemble helps at every x, and less the more the runs share.

The $x = 0$ point: a shared initialization does not keep two full runs in one basin. The midpoint of the interpolation is at 5.5–5.7, the loss the constant-LR runs reach after about 50 updates. The shared start still shows: two full runs from *different* seeds (42 and 43, same data orders) have a weight average at 8.16 / 7.59, about the uniform-guess loss ($\ln 4096 = 8.3$). But the barrier is not mainly a relabeling of hidden

units. On another same-init pair ($D = +2.51$ / $+2.58$), weight matching (Git Re-Basin) permutes about 80% of the MLP units of the second run to match the first, yet D drops only to $+2.30$ / $+2.43$. The two runs' MLP units end up nearly uncorrelated (activation correlation .04 unit-for-unit, .13 after the best matching), so they learned different features rather than the same features in a different order.

Q2. Fixing a mis-set peak LR mid-run

Difficulty: Hard

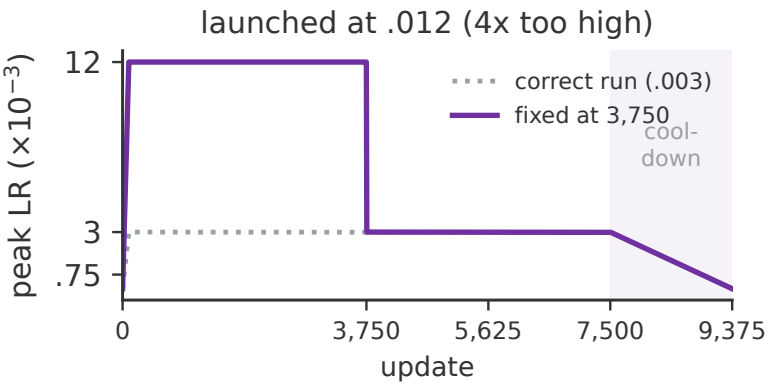
20 points

From the assignment (reference only). Recall the schedule experiments of Problem 2. With the `wsd0.2` schedule at peak LR .003, the final validation loss is 2.935. In the dip-and-rewarm experiments of Problem 2(c), we lowered this run's LR for part of training and brought it back to .003 to see the impact of LR changes.

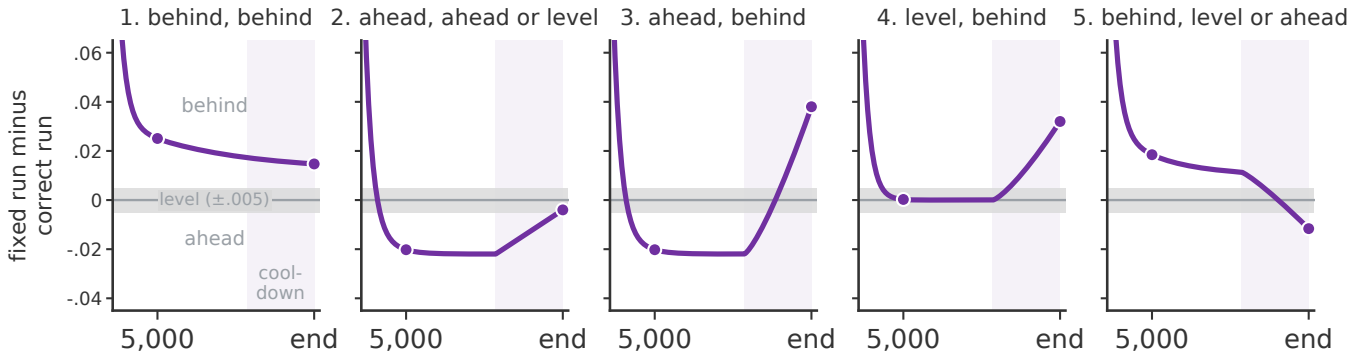
The diff behind this question is Appendix A.2, page 20.

(a) A too-high start, before and after the cooldown (10 points).

This question changes. The `wsd0.2` run is launched with the **wrong peak LR**, .012 (4× too high). Partway through, the mistake is caught and the LR is **set to** .003 for the rest of training, with the same cooldown.



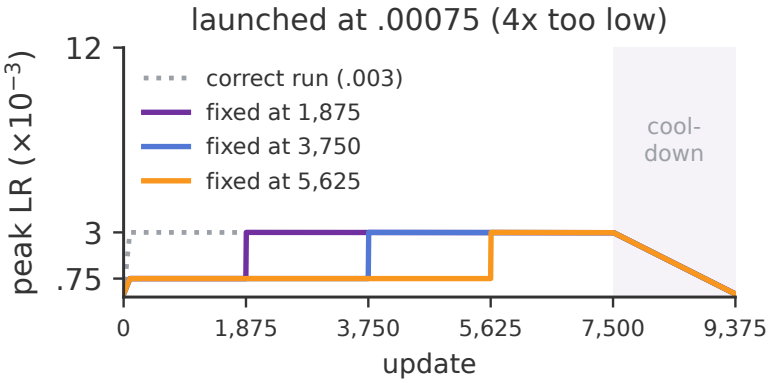
The LR .012 run is fixed at update 3,750 (See Figure). Compare its validation loss with the baseline run (`wsd0.2` with LR .003) at update 5,000 (before the cooldown) and at the end of training. The fixed run is *ahead* if its loss is lower by more than .005, *level* if the two are within .005, and *behind* otherwise. Which sketch matches the fixed run at update 5,000 and at the end?



· 1. · 2. · 3. · 4. · 5.

(b) A too-low start, fixed early or late (10 points).

This question changes. The `wsd0.2` run is launched at .00075 (4× too low) and fixed the same way, at one of three points, or left alone.



Consider the four 0.00075 LR runs: three are fixed at updates 1,875, 3,750 or 5,625 (20%, 40%, 60% of training, see figure) and one is left alone. Rank the choices of when (and whether) to fix the LR based on validation loss. Each option has the lowest loss choice first.

- 1. 20% < 40% < 60% < left alone
- 2. 20% < 40% < left alone < 60%
- 3. 20% < left alone < 40% < 60%
- 4. 20% = 40% = 60% (within .005) < left alone

Solution. (a) 3 (ahead at 5,000, behind at the end). (b) 2 (20% < 40% < left alone < 60%). Final validation losses (release starter, `wsd0.2`; seed 42 is the runs of the diff above; seed 43 uses an equivalent step schedule, the LR $\times 4$ or $\times 0.25$ until the fix, instead of a fork):

Run	Final loss	Minus correct run
Correct, .003 throughout	2.9356 / 2.9344	
.012, never fixed	3.0123 / 3.0124	+.077 / +.078
.012, fixed at 3,750	2.9698 / 2.9705	+.034 / +.036
.00075, never fixed	2.9635 / 2.9622	+.028 / +.028
.00075, fixed at 1,875	2.9386 / 2.9379	+.003 / +.004
.00075, fixed at 3,750	2.9529 / 2.9522	+.017 / +.018
.00075, fixed at 5,625	2.9761 / 2.9751	+.041 / +.041

(a) The too-high run is ahead until the cooldown, then behind. At update 5,000 the fixed run is .022 / .023 below the correct run; at the end it is .034 / .036 above it. The 4× LR doubled the RMS of every weight matrix (.228 vs .111 in a layer-3 MLP matrix at the fix). RMSNorm makes the loss blind to that scale, so a fixed AdamW step at .003 is a smaller relative step on the inflated matrices: the first ∼2,000 updates after the fix act like a partial cooldown, and the run moves ahead. Weight decay shrinks the matrices back only on the timescale $1/(\eta \cdot \text{wd}) = 3{,}333$ updates (the transient time of Problem 2(c)); the RMS is still 1.22× the correct run's at the cooldown start. Until then the run makes less progress along the river, and the shared cooldown exposes the deficit. A short LR sweep without a cooldown would rank this run as better than the correct one. Seed 44: −.024 at 5,000 and +.033 at the end.

(b) Fixing a too-low LR late is worse than leaving it alone. The earlier the fix, the better, but the fix at 60% ends .012 / .013 above the run left alone. Treat the too-low phase as one long Problem 2(c) dip to 0.25×. The LR area removed by update 5,625 is $0.75 \times 5{,}580 \approx 4{,}180$ peak-LR updates, so the dip law ($\approx 4 \times 10^{-5}$ per update removed) gives a cost of about .17 right after the raise. It decays as $e^{-1{,}875/3{,}000} \approx .54$ by the cooldown, and about half of what remains survives the cooldown: +.05 over the correct run, or ≈ 2.98, above the 2.963 of the run left alone. The same arithmetic for the fix at 3,750 gives +.017 (measured 2.953), and for a fix at 1,875 it gives +.005. The raised run pays a rewarm transient that has too little time to decay before the cooldown; the run left alone keeps its low-LR valley position. Seed 44: $2.939 < 2.954 < 2.961 < 2.975$, the same order.

Q3. Weight averaging with an EMA

Difficulty: Medium

20 points

From the assignment (reference only). Recall the evaluation-only EMA of the weights in Problem 2(b) (decay .999, a window of about 1,000 updates). It helps a lot at constant LR and slightly hurts at the end of schedules that decay to zero. At a constant LR of .003 after the warmup, the final validation loss is 3.099 for the raw weights and 2.973 for the EMA.

The diff behind this question is Appendix A.3, page 20.

(a) The best peak LR for constant LR plus EMA (10 points).

This question changes. The constant-LR run at peak LR **.0015 and .006**, with the same EMA.

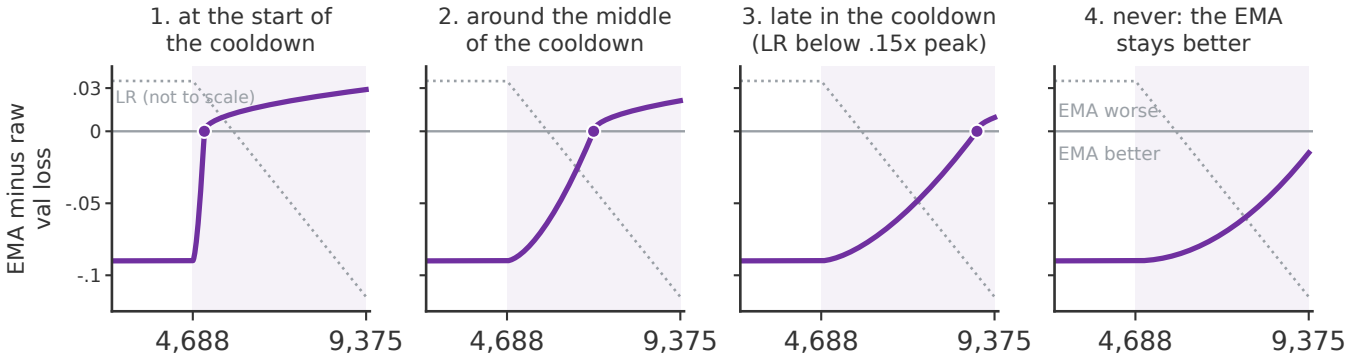
Let $L_{.0015}$, $L_{.003}$ and $L_{.006}$ be the final validation losses of the EMA weights at the three peak LRs. Which ranking do you predict?

· 1. $L_{.0015} < L_{.003} < L_{.006}$ · 2. $L_{.003}$ is the lowest of the three
· 3. $L_{.006} < L_{.003} < L_{.0015}$ · 4. All three within .005

(b) When the EMA stops helping (10 points).

This question changes. The `wsd0.5` schedule at peak LR .003 (constant, then a linear decay to zero over the last 50% of training, from update 4,688), with the same EMA.

In the `wsd0.5` run, let $\Delta(t)$ be the EMA's validation loss minus the raw weights' at update t. When does Δ become positive (EMA worse) for good?



· 1. · 2. · 3. · 4.

Solution. (a) 1 ($L_{.0015} < L_{.003} < L_{.006}$). (b) 3 (late in the cooldown).

(a) An EMA wants a lower peak LR than a decaying schedule. Final validation losses at constant LR (release starter; seeds 42 / 43):

Peak LR	Raw	EMA .999	EMA gain
.0015	3.058 / 3.057	2.961 / 2.960	.098
.003	3.100 / 3.098	2.973 / 2.973	.126
.006	3.185 / 3.182	3.029 / 3.028	.156

The EMA's gain grows with the LR, but not enough to reverse the raw ordering. Seed 44: EMA 2.959 / 2.979 / 3.033, the same order. Problem 2(a): the best peak LR falls as a schedule spends more of training at the peak; linear decay is best at .003, while `wsd0.2` (80% at the peak) is flat from .0015 to .003 and loses the most at .006. Constant LR spends all of training at the peak. Problem 2(b): the EMA removes about 73% of the constant-vs-linear gap at .003; what is left is the part that averaging cannot fix, and it grows with the LR. The linear-decay losses at the same peaks are 2.947 / 2.927 / 2.942, so the gap between EMA and linear decay grows with the peak: .014, .046, .087.

(b) The EMA turns worse only late in the cooldown. Δ becomes positive for good at update 9,024 in both seeds, where the LR is .075$\times$ the peak. At update 8,672 (LR .15$\times$) it is still $-.008$ / $-.009$, and at the start of the cooldown it is about $-.09$. Problem 2(b), linear row: at update 7,500 the LR is .2$\times$ the peak and the EMA is still .025 ahead, and it ends .004 behind. The EMA beats the raw weights while averaging out the LR noise is worth more than its lag along the river, which holds until the LR is very small.

Q4. Dips and rewarms

Difficulty: Medium

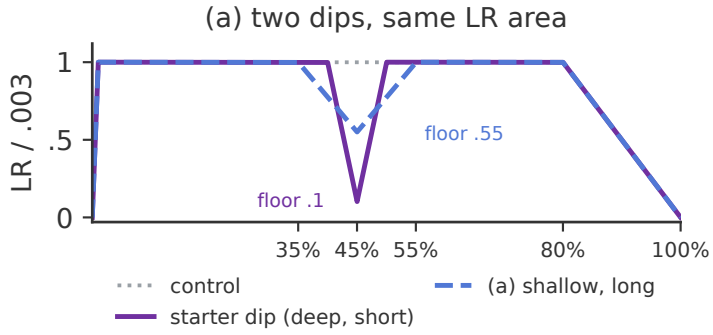
20 points

From the assignment (reference only). Recall the dip-and-rewarm experiments of Problem 2(c). A dip multiplies the LR of the `wsd0.2` control (peak .003) by a factor that falls linearly from 1 to a floor and then returns linearly to 1. The starter dip runs from 40% to 50% of training with a floor of 0.1× at 45%, and gives a large temporary gain at its bottom.

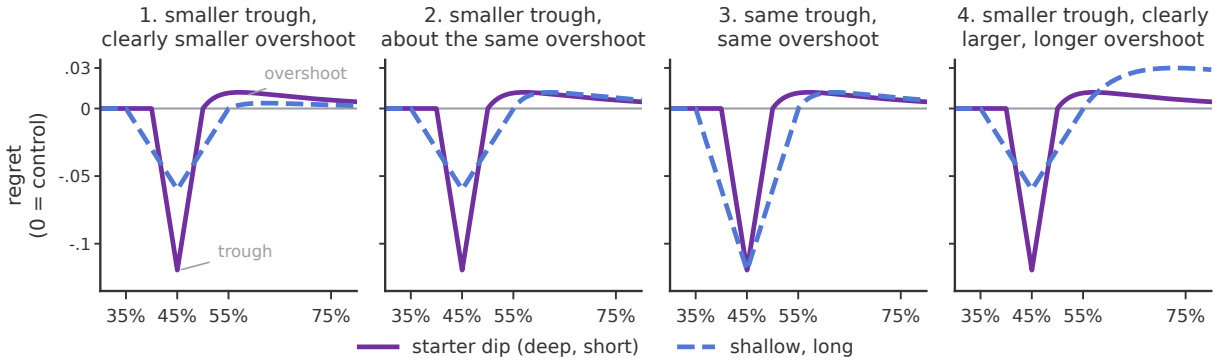
The diff behind this question is Appendix A.4, page 21.

(a) Same area, different shape (10 points).

This question changes. A **shallow, long dip** from 35% to 55% of training with a floor of 0.55× at 45%. It removes the same LR area as the starter dip (422 updates' worth of peak LR).



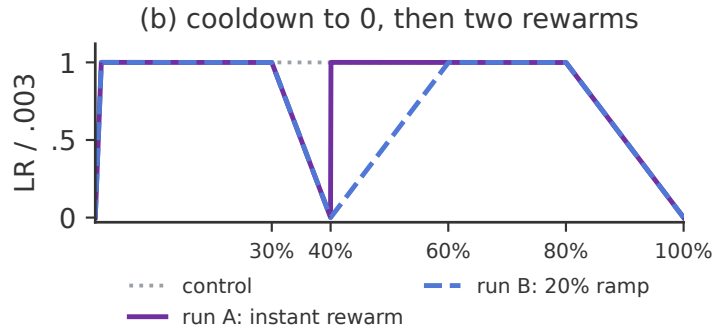
Which sketch matches the two dips' regret curves? ("About the same overshoot": within a factor of 1.5.)



· 1. · 2. · 3. · 4.

(b) Instant or gradual rewarm (10 points).

This question changes. Two runs on the control: run A cools **linearly to zero** between 30% and 40% of training and then **jumps back** to .003; run B cools the same way and then **re-warms linearly** from 0 to .003 between 40% and 60%.



Let L_A and L_B be the final validation losses of runs A and B. Which interval contains $L_A - L_B$?

· 1. $(-\infty, -.005)$ · 2. $[-.005, .005)$ · 3. $[.005, \infty)$

Solution. (a) 2 (smaller trough, about the same overshoot). (b) 1 (A better by more than .005).

(a) The overshoot follows the LR area, not the shape. Regret against the `wsd0.2` control (release starter; seeds 42 / 43):

Dip	Trough	Mean overshoot, updates 5,300–7,400	Final regret
Starter (deep, short)	$-.117$ / $-.119$	$+.0103$ / $+.0092$	$+.003$ to $+.005$
Shallow, long	$-.070$ / $-.071$	$+.0115$ / $+.0120$	$+.003$ to $+.005$

The overshoot ratio (shallow over deep) is 1.1 / 1.3. Problem 2(c) fits the post-rewarm overshoot as proportional to the LR area removed ($\approx 4 \times 10^{-5}$ per peak-LR update), decaying on the weight-decay timescale of about 3,000 updates. Both dips remove 422 updates, so both pay about the same overshoot. The trough measures how low the LR goes: the deep dip reaches further down the valley walls. Seed 44: troughs $-.117$ / $-.069$, overshoots $+.0113$ / $+.0112$ (ratio 1.0).

(b) An instant rewarm is better. Final regret against the control: run A $+.0045$ / $+.0041$, run B $+.0147$ / $+.0151$, so $L_A - L_B = -.010$ / $-.011$. Both cooldowns remove the same triangle of LR area ($0.1 \times 9{,}375/2 \approx 469$ updates), and B's ramp removes another $0.2 \times 9{,}375/2 \approx 938$. By the dip law that extra area costs $4 \times 10^{-5} \times 938 \approx .037$ at the end of the ramp; it decays by $e^{-1{,}875/3{,}000} \approx .54$ before the cooldown and roughly halves through it, leaving $\approx .011$. The rewarm transient after A's instant jump is short; a gradual ramp gives up peak-LR progress that the run cannot recover before the cooldown. Seed 44: $L_A - L_B = -.0105$.

Q5. Full-batch gradient descent at the edge

Difficulty: Medium

20 points

From the assignment (reference only). Recall the full-batch experiments of Problem 3(a): gradient descent on the first N training sequences for 1,500 updates, with $\lambda_{\max}$ measured on the training set every 25 updates. In the clean case ($N = 32$, $\eta = .01$, `d8`), $\eta\lambda_{\max}$ first exceeds 2 at update 375.

The diff behind this question is Appendix A.5, page 21.

(a) How far above 2 at a large step size (10 points).

This question changes. $N = \mathbf{256}$ sequences and $\eta = \mathbf{0.2}$.

From update 100 on, while the training loss is still above 1 (before the sequences are memorized), which interval contains the median of $\eta\lambda_{\max}$?

· 1. $(0, 2.2)$ · 2. $[2.2, 3)$ · 3. $[3, 6)$ · 4. $[6, \infty)$ · 5. The run diverges

(b) Depth and progressive sharpening (10 points).

This question changes. The clean case at **depth 4 and depth 16** (width 512, 64-dimensional heads), next to the standard depth 8.

Let t_L be the first update at which $\eta\lambda_{\max} > 2$ at depth L. Which ranking do you predict?

· 1. $t_4 < t_8 < t_{16}$ · 2. All three within 20% of each other
· 3. $t_{16} < t_8 < t_4$ · 4. t_8 is the earliest of the three

Solution. (a) 3 ($[3, 6)$). (b) 3 ($t_{16} < t_8 < t_4$).

(a) The edge is a band that rises with the step size. The median of $\eta\lambda_{\max}$ is 4.66 / 4.63 (release starter, seeds 42 / 43); the loss passes 1 near update 965 / 838 and the run never diverges. Our earlier run of the same recipe gives 4.48 (10th–90th percentile 3.9–6.9), and the same statistic at $\eta = .1$ is 3.72. Problem 3(a): the plateau sits above $2/\eta$ and the overshoot grows with η, from a median of about 2.6 at $\eta = .01$ to 2.9 at .02 and 3.3 at .03, roughly $+.4$ to $+.5$ per doubling. $\eta = .2$ is 2.7 doublings past .03, which extrapolates to about 4.5. A run above the edge sheds sharpness instead of diverging: when η was doubled mid-run in Problem 3(a), $\lambda_{\max}$ fell from 256 to 193 within 25 updates. The value 2 is exact only for a quadratic.

(b) Deeper networks reach the edge sooner. First crossings at depth 4 / 8 / 16: updates 575 / 375 / 225 (seed 43: 500 / 350 / 225). Problem 3(a): sharpening follows training progress, and deeper networks fit faster per update; they also start sharper ($\lambda_{\max}$ at initialization 17 / 22 / 32). The intuition that larger networks are lazier and flatter does not describe this regime.

Q6. Weight decay as a sharpness knob

Difficulty: Medium

20 points

From the assignment (reference only). Recall Problem 4(b): weight decay moves $\lambda_{\max}$ at nearly fixed loss through rescaling symmetries. The default mask decays the hidden and readout matrices (including the LM head) but not the embedding or the RMSNorm gains. With this mask and peak LR .003, weight decay 0, .1 and .3 give a final $\lambda_{\max}$ of about 2.5, 6 and 15.

The diff behind this question is Appendix A.6, page 21.

(a) Decaying every tensor (10 points).

This question changes. Weight decay .1 is applied to **every tensor**, including the embedding and all RMSNorm gains.

Which interval contains the run's final $\lambda_{\max}$ divided by the default run's (same seed)?

· 1. $(0, .8)$ · 2. $[.8, 1.2)$ · 3. $[1.2, \infty)$

(b) Trading LR for weight decay (10 points).

This question changes. The default mask, with the peak LR η and the weight decay w changed **together at a fixed product** $\eta w = 3 \times 10^{-4}$, the default value:

Run	Peak LR η	Weight decay w	ηw
S	.001	.3	3×10^{-4}
Default	.003	.1	3×10^{-4}
T	.01	.03	3×10^{-4}

Let λ_S, λ_{def} and λ_T be the final $\lambda_{\max}$ of runs S, Default and T. Which ranking do you predict?

· 1. All three within 20% of each other · 2. $\lambda_S > \lambda_{\text{def}} > \lambda_T$
· 3. $\lambda_T > \lambda_{\text{def}} > \lambda_S$ · 4. λ_{def} is the highest of the three

Solution. (a) 1 ($(0, .8)$). (b) 2 ($\lambda_S > \lambda_{\text{def}} > \lambda_T$). Final $\lambda_{\max}$ and the sharpest family block, 64-sequence probe (seed 42: the runs and measurements of the diffs above, on the release starter; seeds 43 and 44: our earlier runs of the same recipe):

Run	$\lambda_{\max}$	Sharpest block	Probe loss
Default (.003, .1)	5.97 / 4.81 / 4.55	LM head	2.947 / 2.944 / 2.948
(a) every tensor decayed, .1	4.30 / 3.27 / 3.08	LM head	2.948 / 2.948 / 2.948
S (.001, .3)	11.49 / 11.81	LM head and attention	2.967 / 2.968
T (.01, .03)	3.01 / 1.46	RMSNorm gains	2.986 / 2.992

(a) Decaying the gains removes the push that sharpens the head. The ratio is .72 / .68 / .68 at the same loss. Paired within the release starter (a default run of its own at each seed): .69 / .67 / .72 at seeds 42 / 43 / 44, so the key holds by about four standard deviations of the seed spread. Problem 4(b): the logits depend only on $W\,\mathrm{diag}(g)$ (LM head W, final RMSNorm gain g). The default mask decays W but not g, which pushes the pair along the exact symmetry $W \to W/c$, $g \to cg$ and multiplies the head's curvature by c^2; that is why more head decay sharpens the head and none flattens it. Decaying g as well removes that push, so the head ends flatter. The same holds at weight decay .3: decaying every tensor gives 12.4 / 11.2 against 15.6 / 15.1 for the default mask, though there the shrunken RMSNorm gains become the sharpest block.

(b) The product ηw does not set the sharpness. Both changes in S push $\lambda_{\max}$ up. More weight decay sharpens the head along the gain/head symmetry (the reference box: .3 gives about 15 at .003), and a lower LR tolerates more curvature (Problem 3(c): λ_{pre} scales roughly as $1/\eta$; Problem 4(a): $\lambda_{\max}$ rises as the LR decays). T reverses both, and with little weight decay the RMSNorm gains take the top block, as at weight decay 0 in Problem 4(b). The product ηw sets the timescale on which weight decay forgets, not where the run sits on the symmetry orbit, and the probe losses show that the split matters for the loss too.

Q7. Training in the top subspace

Difficulty: Hard

10 points

From the assignment (reference only). Recall Problem 4(c). From the SGDM checkpoint at 100M tokens, removing the component of each update in the top-10 Hessian eigenspace costs nothing over 300 steps, while keeping only that component makes no progress.

The diff behind this question is Appendix A.7, page 22.

Adam fed only the top subspace (10 points).

This question changes. The keep-only experiment on the **AdamW checkpoint** at 100M tokens: the minibatch gradient is replaced by its component in the top-10 eigenspace of the Hessian **before AdamW sees it**, for 300 steps, with the eigenvectors recomputed every 25 steps. AdamW's moment estimates are carried over from the checkpoint, not reset.

Which interval contains the validation loss after the 300 steps minus the validation loss at the checkpoint (positive if the loss went up)?

· 1. $(-\infty, -.05)$ · 2. $[-.05, .05)$ · 3. $[.05, .3)$ · 4. $[.3, \infty)$

Solution. 4 ($[.3, \infty)$).

Adam turns a tiny projected gradient into large, undirected steps. The validation loss goes from 3.32 to 6.86 (seed 42, the diff's job) and from 3.33 to 5.49 (seed 43); the full update reaches 3.28. Problem 4(c): with SGDM, the top-subspace component carries no useful descent. AdamW divides by $\sqrt{\hat{v}}$; the checkpoint's $\hat{v}$ is carried over, but with $\beta_2 = .95$ it is rebuilt from the projected gradient itself (2–10% of $\|g\|^2$) within a few dozen steps, so every coordinate soon takes a full-size step again. The sign pattern of those steps is no longer confined to the top subspace, so the update is large motion in no useful direction.

Appendix: the diffs behind the questions

Each question's runs are exactly these patches to the released A3 starter (the `assignments` repository, branch `a3_release` at `57c60bb`); each applies with `git apply`. Red lines are removed, green lines are added. The answer key comes from these runs.

A.1 — Q1: Basins

Part (a):

```
--- a/experiments/a3_optimization/p1_mode_connectivity.py
+++ b/experiments/a3_optimization/p1_mode_connectivity.py
@@ -62,2 +62,11 @@ BRANCHES = {

+# Quiz 3, Question 1(a): the same sweep at a constant LR of .01 (a Problem 1(d) knob), measured on
+# the cells (100, 50), (100, 250) and (250, 250).
+LR01 = replace(BASE, learning_rate=0.01)
+PREFIX_LR01 = prefix_for(LR01)
+BRANCHES_LR01 = {x: [branch(x, seed, base=LR01) for seed in BRANCH_ORDER_SEEDS] for x in (100, 250)}
+LR01_RUNS = [PREFIX_LR01] + [run for runs in BRANCHES_LR01.values() for run in runs]
+LR01_PAIRS = [((training_run_name(a), x + y), (training_run_name(b), x + y))
+              for x, y in ((100, 50), (100, 250), (250, 250)) for a, b in [BRANCHES_LR01[x]]]
+

@@ -107,3 +116,3 @@ def main():

-        launch_training_jobs([PREFIX])
+        launch_training_jobs([PREFIX, PREFIX_LR01])
     elif args.stage == "branches":
@@ -112,3 +121,4 @@ def main():
         check_prefix_checkpoints()
-        launch_training_jobs([run for runs in BRANCHES.values() for run in runs])
+        check_prefix_checkpoints(PREFIX_LR01)
+        launch_training_jobs([run for runs in BRANCHES.values() for run in runs] + LR01_RUNS[1:])
     else:
@@ -116,3 +126,3 @@ def main():

-        rows = measure_pairs_on_modal(pairs())
+        rows = measure_pairs_on_modal(pairs() + LR01_PAIRS)
         RESULTS_PATH.parent.mkdir(parents=True, exist_ok=True)
```

Part (b):

```
--- a/experiments/a3_optimization/p1_mode_connectivity.py
+++ b/experiments/a3_optimization/p1_mode_connectivity.py
@@ -62,2 +62,11 @@ BRANCHES = {

+# Quiz 3, Question 1(b): the default recipe in data order 1, and a second run in data order 2
+# forked from it at update x (x = 0: a fresh run from the same initialization).
+FORKS = (0, 100, 250, 1000, 4000)
+FULL = TrainConfig(batch_order_seed=1, keep_checkpoint_steps=FORKS[1:], run_name_suffix="quiz3-q1b")
+SECOND = [replace(FULL, batch_order_seed=2, keep_checkpoint_steps=(),
+                  **(dict(fork_from_run=training_run_name(FULL), fork_from_step=x) if x else {}))
+          for x in FORKS]
+FINAL_PAIRS = [((training_run_name(FULL), None), (training_run_name(r), None)) for r in SECOND]
+

@@ -107,3 +116,3 @@ def main():

-        launch_training_jobs([PREFIX])
+        launch_training_jobs([PREFIX, FULL])
     elif args.stage == "branches":
```

```
@@ -112,3 +121,3 @@ def main():
         check_prefix_checkpoints()
-        launch_training_jobs([run for runs in BRANCHES.values() for run in runs])
+        launch_training_jobs([run for runs in BRANCHES.values() for run in runs] + SECOND)
     else:
@@ -116,3 +125,3 @@ def main():

-        rows = measure_pairs_on_modal(pairs())
+        rows = measure_pairs_on_modal(pairs() + FINAL_PAIRS)
         RESULTS_PATH.parent.mkdir(parents=True, exist_ok=True)
```

A.2 — Q2: Fixing a mis-set peak LR mid-run

```
--- a/experiments/a3_optimization/p2_river_valley.py
+++ b/experiments/a3_optimization/p2_river_valley.py
@@ -5,4 +5,6 @@

+import sys
+
 from modal_train import launch_training_jobs
-from train import TrainConfig
+from train import TrainConfig, training_run_name

@@ -40,2 +42,11 @@ RUNS = SCHEDULE_RUNS + REWARM_RUNS + HORIZON_FREE_RUNS

+# Quiz 3, Question 2: wsd0.2 launched at a wrong peak LR (4x too high or too low). The fix forks
+# the run at update u with the peak LR set to .003 (stage `quiz3-q2`, after these runs finish).
+WRONG = [TrainConfig(lr_schedule="wsd0.2", learning_rate=lr, keep_checkpoint_steps=u,
+                     run_name_suffix="quiz3-q2")
+         for lr, u in ((0.012, (3750,)), (0.00075, (1875, 3750, 5625)))]
+FIXED = [TrainConfig(lr_schedule="wsd0.2", fork_from_run=training_run_name(run), fork_from_step=u,
+                     run_name_suffix="quiz3-q2") for run in WRONG for u in run.keep_checkpoint_steps]
+RUNS += WRONG
+

@@ -49,3 +60,3 @@ RUNS = SCHEDULE_RUNS + REWARM_RUNS + HORIZON_FREE_RUNS
 def main():
-    launch_training_jobs(RUNS)
+    launch_training_jobs(FIXED if sys.argv[1:] == ["quiz3-q2"] else RUNS)
```

A.3 — Q3: Weight averaging with an EMA

```
--- a/experiments/a3_optimization/p2_river_valley.py
+++ b/experiments/a3_optimization/p2_river_valley.py
@@ -40,2 +40,7 @@ RUNS = SCHEDULE_RUNS + REWARM_RUNS + HORIZON_FREE_RUNS

+# Quiz 3, Question 3: constant LR at .0015 and .006, and wsd0.5 at .003, with the same EMAs.
+RUNS += [TrainConfig(lr_schedule=schedule, learning_rate=lr, ema_decays=EMA_DECAYS,
+                     run_name_suffix="quiz3-q3")
+         for schedule, lr in (("wsd0.0", 1.5e-3), ("wsd0.0", 6e-3), ("wsd0.5", 3e-3))]
+
```

A.4 — Q4: Dips and rewarms

```
--- a/experiments/a3_optimization/p2_river_valley.py
+++ b/experiments/a3_optimization/p2_river_valley.py
@@ -40,2 +40,7 @@ RUNS = SCHEDULE_RUNS + REWARM_RUNS + HORIZON_FREE_RUNS

+# Quiz 3, Question 4: (a) a shallow, long dip with the part (c) dip's LR area; (b) a cooldown to 0
+# over 30-40% of training, then an instant (A) or a 40-60% linear (B) return to the peak LR.
+RUNS += [TrainConfig(lr_schedule="wsd0.2", lr_dip=dip, run_name_suffix="quiz3-q4")
+         for dip in ((0.35, 0.45, 0.55, 0.55), (0.3, 0.4, 0.4, 0.0), (0.3, 0.4, 0.6, 0.0))]
+
```

A.5 — Q5: Full-batch gradient descent at the edge

```
--- a/experiments/a3_optimization/p3_edge_of_stability.py
+++ b/experiments/a3_optimization/p3_edge_of_stability.py
@@ -8,4 +8,7 @@ logging/sharpness/*. Read the results back with experiments.a3_optimization.shar

+from dataclasses import replace
+
 from metric_logging import AFTER_BACKWARD, MetricLogger
 from modal_train import launch_training_jobs
+from model_config import depth_model_config
 from sharpness_logging import SharpnessLogger
@@ -79,2 +82,8 @@ FULL_BATCH_RUNS = [
     full_batch(32, 1500, learning_rate=0.01),  # (a) i: the clean case
+    # Quiz 3, Question 5: (a) N = 256 at eta = 0.2; (b) the clean case at depth 4 and 16.
+    full_batch(256, 1500, learning_rate=0.2, run_name_suffix="quiz3-q5a"),
+    *(full_batch(32, 1500, learning_rate=0.01, run_name_suffix="quiz3-q5b",
+                 model_config=replace(depth_model_config(8), name=f"d{depth}w512",
+                                      num_hidden_layers=depth))  # width 512 at every depth
+      for depth in (4, 16)),
 ]
```

A.6 — Q6: Weight decay as a sharpness knob

Part (a):

```
--- a/experiments/a3_optimization/p4_inside_the_hessian.py
+++ b/experiments/a3_optimization/p4_inside_the_hessian.py
@@ -56,2 +56,3 @@ RUNS = [
     # TrainConfig(<setting>=<value>, wandb_tags=(EXPERIMENT_KEY,)),
+    TrainConfig(run_name_suffix="quiz3-q6a"),  # Quiz 3, Question 6(a)
 ]
--- a/optimizers.py
+++ b/optimizers.py
@@ -13,2 +13,3 @@ def is_norm_module(module):
 def should_apply_weight_decay(module, parameter_name):
+    return True  # Quiz 3, Question 6(a): decay every tensor
     if isinstance(module, torch.nn.Embedding):
```

Part (b):

```
--- a/experiments/a3_optimization/p4_inside_the_hessian.py
+++ b/experiments/a3_optimization/p4_inside_the_hessian.py
@@ -56,2 +56,5 @@ RUNS = [
     # TrainConfig(<setting>=<value>, wandb_tags=(EXPERIMENT_KEY,)),
+    # Quiz 3, Question 6(b): peak LR and weight decay at the default product, 3e-4.
+    TrainConfig(learning_rate=1e-3, weight_decay=0.3, run_name_suffix="quiz3-q6b"),
+    TrainConfig(learning_rate=1e-2, weight_decay=0.03, run_name_suffix="quiz3-q6b"),
 ]
```

A.7 — Q7: Training in the top subspace

```
--- a/experiments/a3_optimization/p4_inside_the_hessian.py
+++ b/experiments/a3_optimization/p4_inside_the_hessian.py
@@ -74,2 +74,4 @@ SUBSPACE_JOBS = [
 ]
+# Quiz 3, Question 7: the top arm on the AdamW checkpoint.
+SUBSPACE_JOBS += [{"name": "quiz3-q7a", "continue": {**START, "run": "adamw", "transform": "top"}}]
```

Answer sheet: Outcome Prediction Quiz: Optimization 2 **130 points**

Name _____ **SUNet ID** _____

your Stanford login, e.g. jdoe; not the 8-digit ID number

Key: ☑ full credit; ½ adjacent option in an ordered row, half credit.

Q1. Basins — 20 points

(a) at LR .01 4 points per row	1. Avg<Ens<Best	2. Avg=Ens<Best	3. Ens<Avg<Best	4. Ens<Best<Avg	5. Ens<Avg=Best
(i) $x = 100, y = 50$	☐	☑	☐	☐	☐
(ii) $x = 100, y = 250$	☐	☐	☐	☑	☐
(iii) $x = 250, y = 250$	☐	☐	☑	☐	☐

(b) sketch of $D(x)$ 8 points ☐1. a cliff ☑2. falls steadily ☐3. large until ~1,000, then drops ☐4. below zero from 250 on

Assignment problem cited	**Observations from the experiment**	**Your rationale**

Q2. Fixing a mis-set peak LR mid-run — 20 points

(a) at 5,000, at the end 10 points ☐1. behind, behind ☐2. ahead, ahead or level ☑3. ahead, behind ☐4. level, behind ☐5. behind, level or ahead

(b) ranking 10 points ☐1. 20% < 40% < 60% < alone ☑2. 20% < 40% < alone < 60% ☐3. 20% < alone < 40% < 60% ☐4. 20% = 40% = 60% < alone

Assignment problem cited	**Observations from the experiment**	**Your rationale**

Q3. Weight averaging with an EMA — 20 points

(a) ranking 10 points ☑1. $L_{.0015} < L_{.003} < L_{.006}$ ☐2. $L_{.003}$ lowest ☐3. $L_{.006} < L_{.003} < L_{.0015}$ ☐4. All within .005

(b) EMA worse for good 10 points ☐1. at the start of the cooldown ½2. around its middle ☑3. late (LR below .15× peak) ½4. never

Assignment problem cited	**Observations from the experiment**	**Your rationale**

Name _____ **SUNet ID** _____

your Stanford login, e.g. `jdoe`; not the 8-digit ID number

Q4. Dips and rewarms — 20 points

(a) sketch 10 points ☐1. smaller trough, smaller overshoot ☑2. smaller trough, similar overshoot ☐3. same trough, same overshoot ☐4. smaller trough, larger, longer overshoot

(b) $L_A - L_B$ 10 points ☑1. $(-\infty, -.005)$ ½2. $[-.005, .005)$ ☐3. $[.005, \infty)$

Assignment problem cited	Observations from the experiment	Your rationale

Q5. Full-batch gradient descent at the edge — 20 points

(a) median $\eta\lambda_{\max}$ 10 points ☐1. $(0, 2.2)$ ½2. $[2.2, 3)$ ☑3. $[3, 6)$ ½4. $[6, \infty)$ ☐5. Diverges

(b) ranking 10 points ☐1. $t_4 < t_8 < t_{16}$ ☐2. Within 20% ☑3. $t_{16} < t_8 < t_4$ ☐4. t_8 earliest

Assignment problem cited	Observations from the experiment	Your rationale

Q6. Weight decay as a sharpness knob — 20 points

(a) $\lambda_{\max}$ ratio 10 points ☑1. $(0, .8)$ ½2. $[.8, 1.2)$ ☐3. $[1.2, \infty)$

(b) ranking 10 points ☐1. Within 20% ☑2. $\lambda_S > \lambda_{\text{def}} > \lambda_T$ ☐3. $\lambda_T > \lambda_{\text{def}} > \lambda_S$ ☐4. λ_{def} highest

Assignment problem cited	Observations from the experiment	Your rationale

Q7. Training in the top subspace — 10 points

loss change 10 points ☐1. $(-\infty, -.05)$ ☐2. $[-.05, .05)$ ½3. $[.05, .3)$ ☑4. $[.3, \infty)$

Assignment problem cited	Observations from the experiment	Your rationale