Quiz 3 solutions

Optimization 2

CS312

Tatsu H

Question 1a:

A minor variation on the ‘mode connectivity’ grid – slightly higher LR than A3

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.

(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.

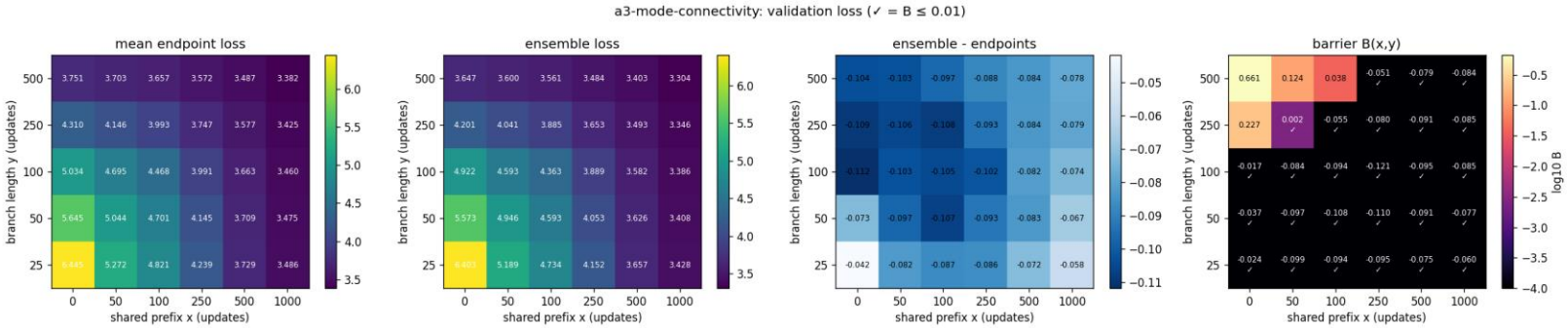
At LR .003 (Problem 1’s grid) the three cells below came out as shown. For each cell, which ordering, lowest (best) first, do you predict at LR .01? Two losses count as equal (=) when within .01.

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

Evidence from A3

Our findings from the A3 grid



Ensemble *always* better or equal to loss average

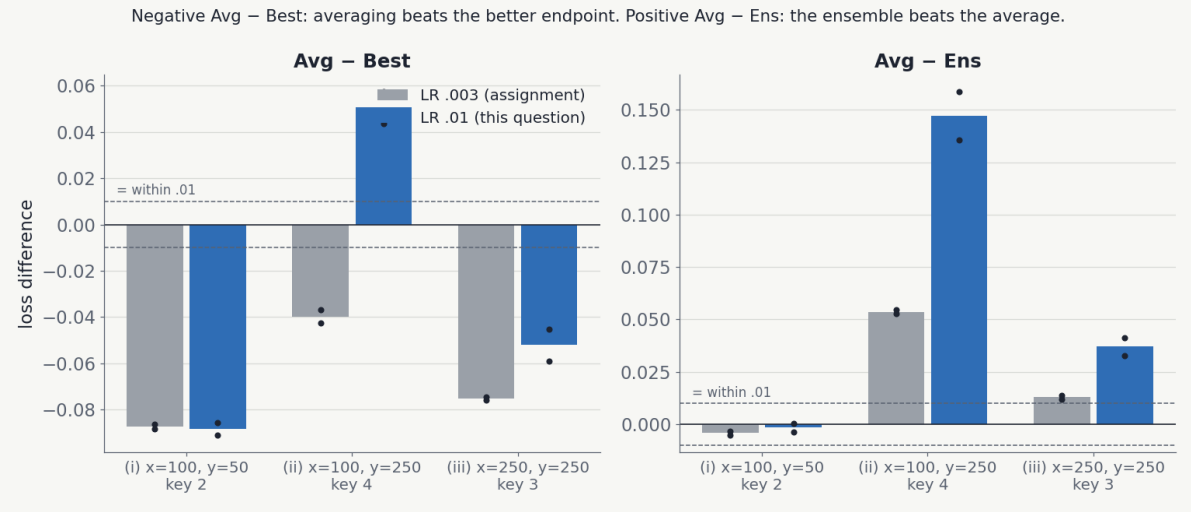
For relatively short prefix, average worse than best

Answer (2, 4, 3)

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



i) 2: Training is short, same as LR=0.003

ii) 4: Best and avg flip – at high LR and longer training, not in the same basin

iii) 3: Long prefix – same as LR=0.003

Support in the literature

Disentangling Linear Mode Connectivity

Gül Sena Altıntaş **Gregor Bachmann** **Lorenzo Noci** **Thomas Hofmann**
ETH Zürich
{galtintas, gregorb, lnoci}@ethz.ch

Abstract

Linear mode-connectivity (LMC) (or lack thereof) is one of the intriguing characteristics of neural network loss landscapes. While empirically well established, it unfortunately still lacks a proper theoretical understanding. Even worse, although empirical data points are abound, a systematic study of when networks exhibit LMC is largely missing in the literature. In this work we aim to close this gap. We explore how LMC is affected by three factors: (1) architecture (sparsity, weight-sharing), (2) training strategy (optimization setup) as well as (3) the underlying dataset. We place particular emphasis on minimal but non-trivial settings, removing as much unnecessary complexity as possible. We believe that our insights can guide future theoretical works on uncovering the inner workings of LMC.

Table 2: Error barriers presented in percentage when changing the optimization setup for the minimal model, MLP with one hidden layer and ReLU activation.

	SGD ↑ Lr	SGD ↑ Lr W-up	ADAM ↓ Lr	ADAM ↑ Lr	ADAM ↑ Lr + W-up
$\%\mathcal{B}_{\text{train}}$	0.00	0.00	0.00	2.92	0.17
$\%\mathcal{B}_{\text{test}}$	0.05	0.02	0.06	2.86	0.23

Similar observations in the literature – Altintas et al
Higher LR leads to higher error barriers in mode connectivity

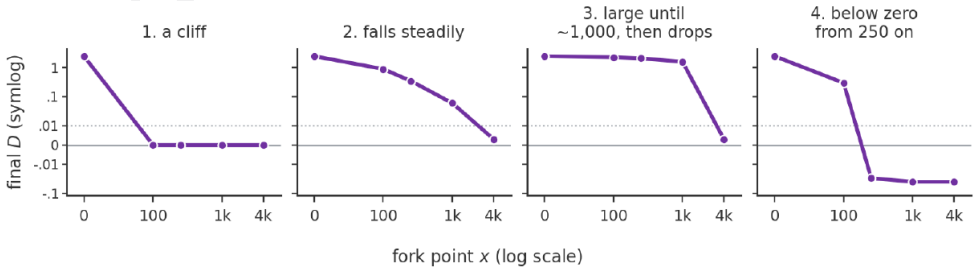
https://arxiv.org/abs/2312.09832

Question 1b

(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 at update** x (weights and optimizer state), for $x \in \{0, 100, 250, 1{,}000, 4{,}000\}$, continue from that point using `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 $D = .01$.

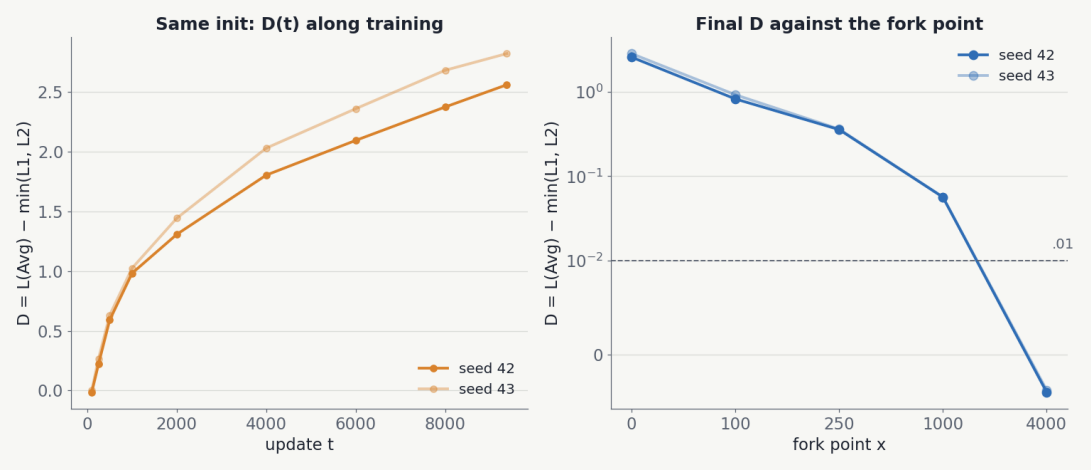


· 1. · 2. · 3. · 4.

How rapidly does the gain from weight averaging go away as we vary the fork point?

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 $D = .01$.

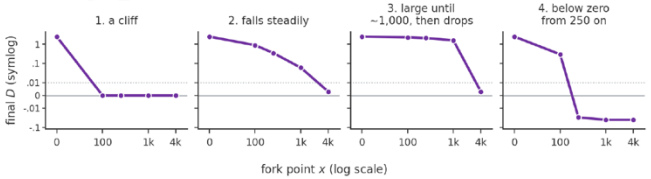
Q1b



(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 at update** x (weights and optimizer state), for $x \in \{0, 100, 250, 1{,}000, 4{,}000\}$, continue from that point using `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.

Option 2 is the right answer –

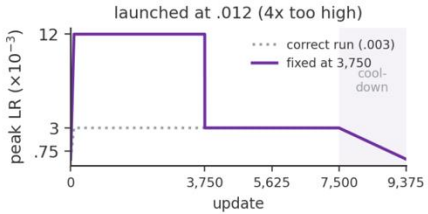
- Not cliff: all fork points are small relative to the rest of the run (9k vs up to 4k)
- Not below zero: linear decay means that no big win from averaging
- Continuous, and not below zero → option 2

Q2a

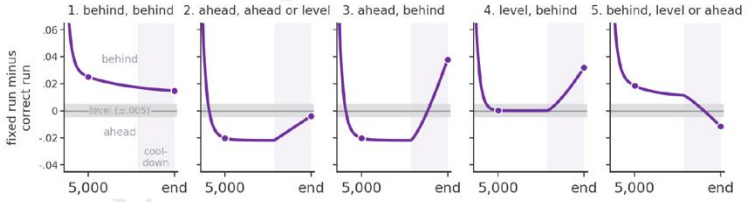
From the assignment (reference only). Recall the schedule experiments of Problem 2. With the `wsd0.2` schedule at peak LR .003, the final validation loss is 2.935. In the dip-and-rewarm experiments of Problem 2(c), we lowered this run's LR for part of training and brought it back to .003 to see the impact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

(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.



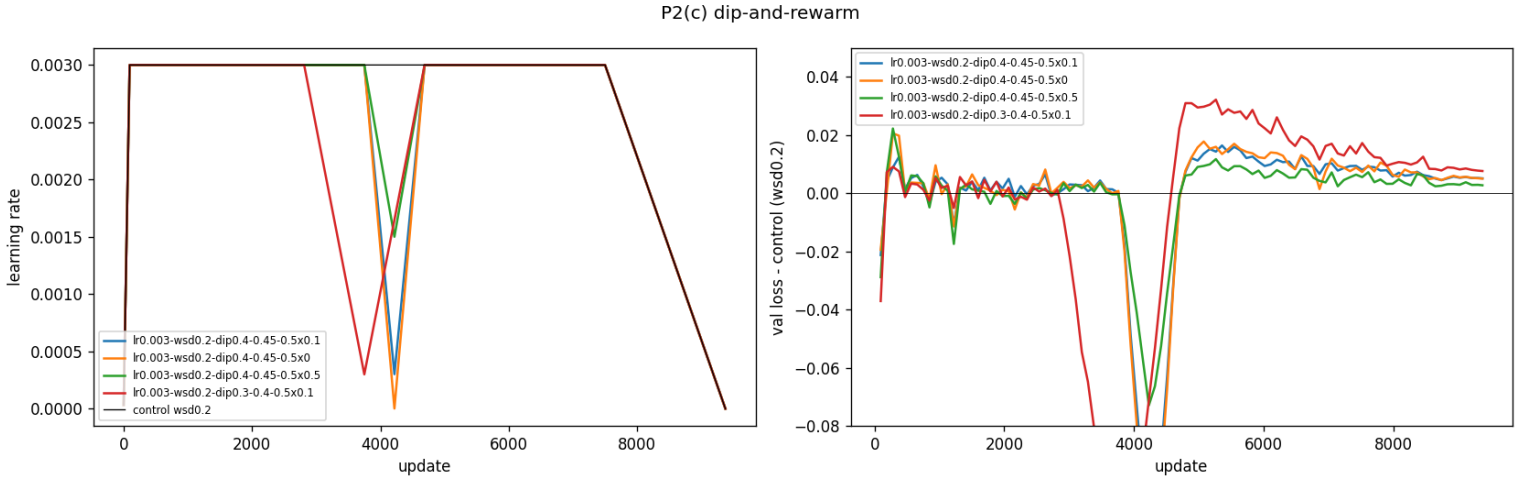
The LR .012 run is fixed at update 3,750 (See Figure). Compare its validation loss with the baseline run (`wsd0.2` with LR .003) at update 5,000 (before the cooldown) and at the end of training. The fixed run is *ahead* if its loss is lower by more than .005, *level* if the two are within .005, and *behind*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



1. 2. 3. 4. 5.

A3 information

Remember your LR transient perturbation experiments

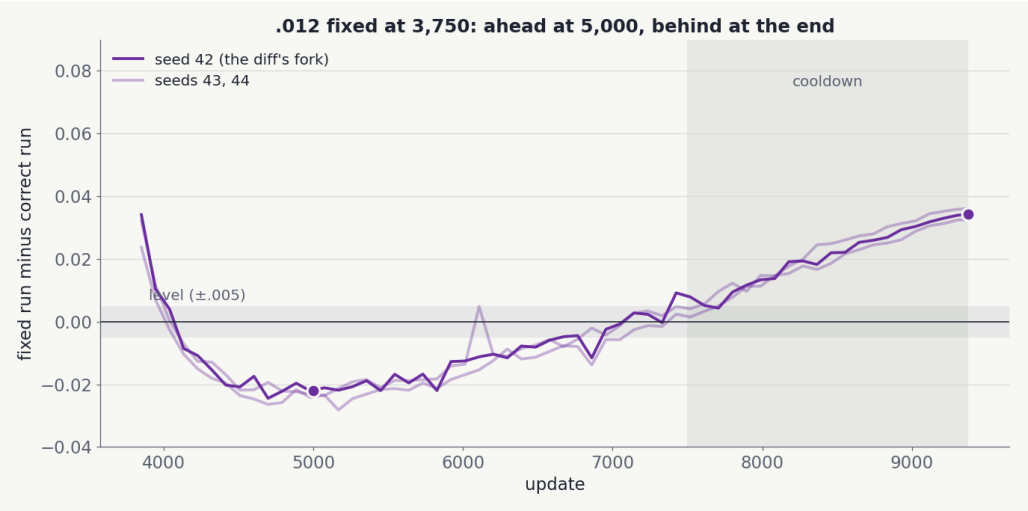


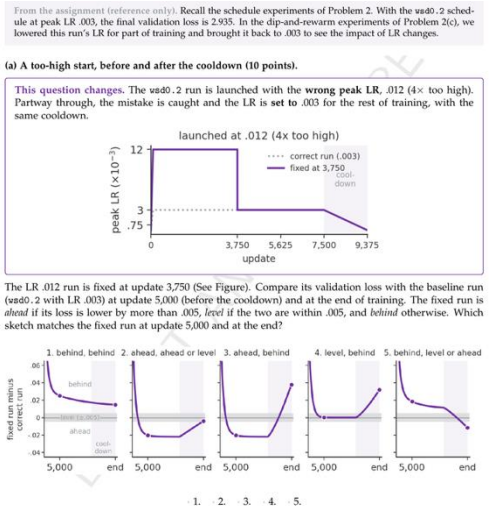
Rough rule of thumb – an LR change leaves a transient that goes away on the WD timescale, $1/(\eta * wd) = 1/(.003 * .1) \approx 3{,}333$ updates

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?

Q2 Answer

Correct option: 3 (ahead, then behind)





Worse at start – LR too high (*and norm too high)*
Drop in LR causes strong annealing effects, leading to wins. ~ 3k step transient
Once WD kicks in in the real run, we lose out since that's a better form of annealing

Rationale

One thing to look at – norm of weights

RMS, layer-3 MLP matrix	At the fix (3,750)	Cooldown start (7,500)
Correct run	.111	.118
.012, fixed	.228 (2x)	.144 (1.22x)

Norm of weights in the high LR run is higher

Can reason about this from the unit 1 results, or from ‘bounces’ in the river valley

Leads to lower effective LR, so it’s as if we’re “cooling down early”

$i \to \infty$ gives us the equilibrium norm listed in table 1

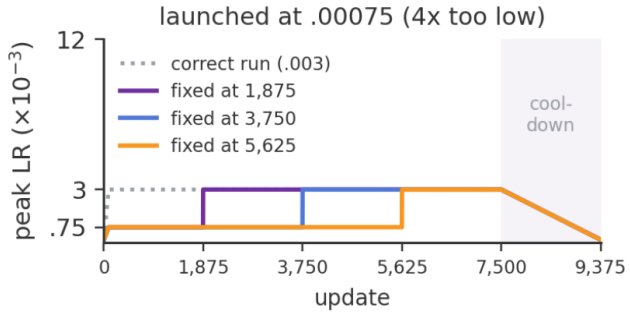
$$\widehat{\|\boldsymbol{\omega}\|} = \sqrt{\frac{\eta C}{2\lambda - \eta\lambda^2}} \approx \sqrt{\frac{\eta C}{2\lambda}} \quad \text{(for } \lambda\eta \ll 2\text{)} \qquad (18)$$

From: **Konsson et al** arXiv 2305.17212

Q2b

(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.

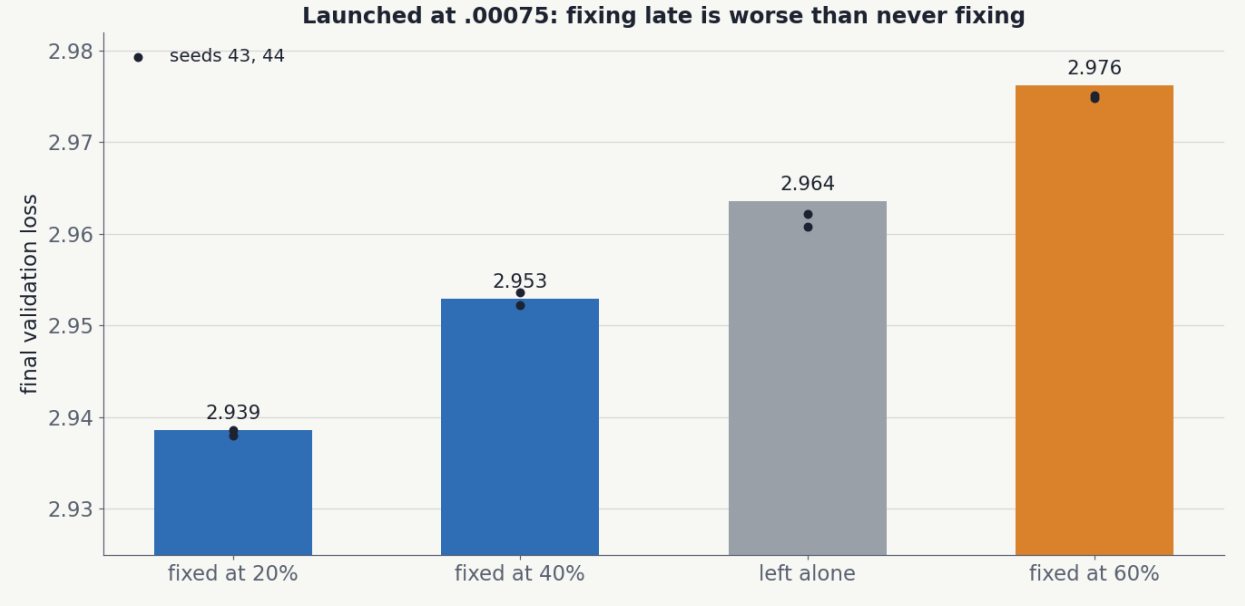
- 1. 20% < 40% < 60% < left alone
- 2. 20% < 40% < left alone < 60%
- 3. 20% < left alone < 40% < 60%
- 4. 20% = 40% = 60% (within .005) < left alone

What happens if we increase LR mid run?

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

Answer / rationale:



Correct answer – 20 < 40 < none **< 60**

Interaction to observe – we need both high LR (to make progress) and low (to anneal)

Rule of thumb – adapting to the new LR takes 1/(wd*eta) ~ 3k steps.

Q3

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.

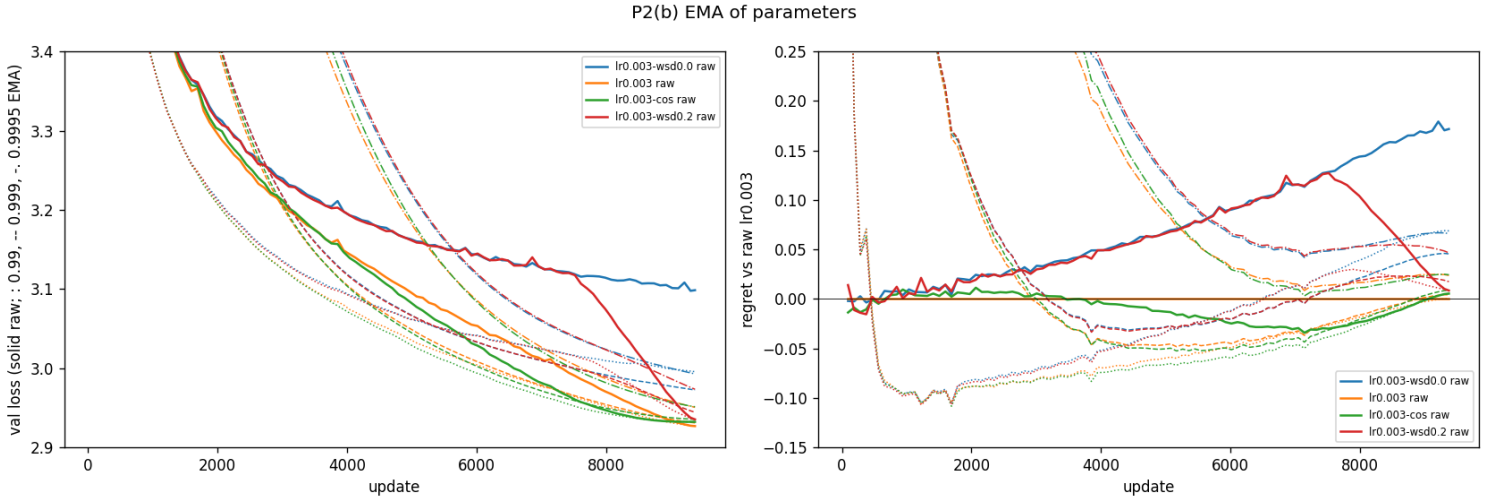
(a) The best peak LR for constant LR plus EMA (10 points).

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

A3 data

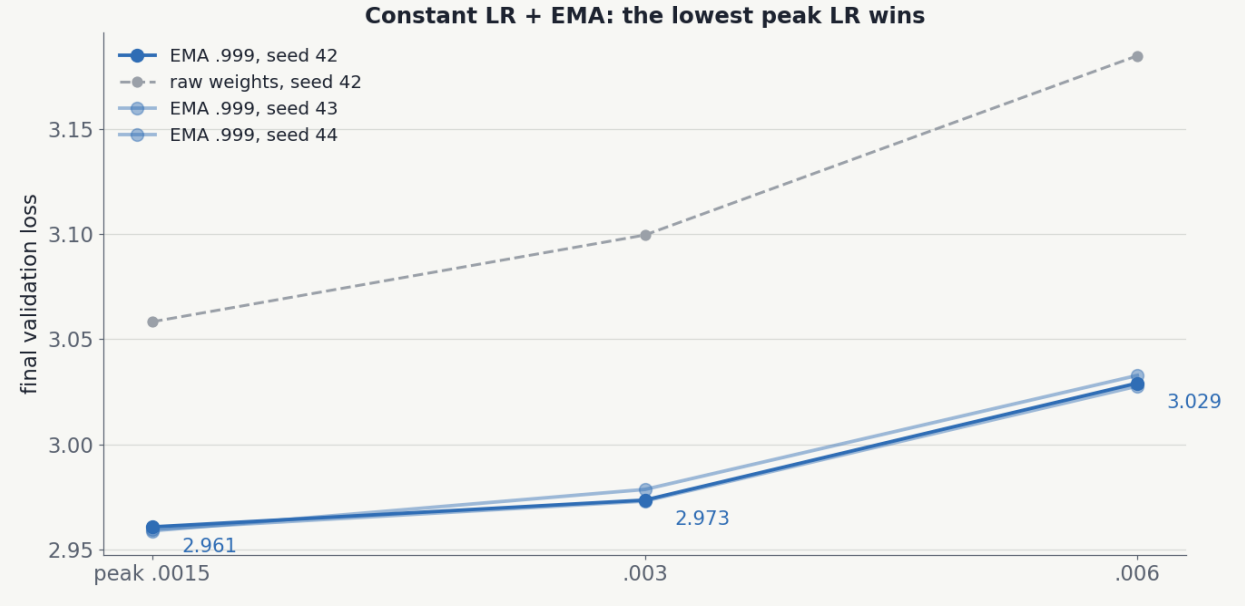


Remember your EMA runs - at constant LR the EMA closes ~73% of the gap to linear decay; under a decay to zero it only (slightly) hurts at the very end

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?

Q3 answer

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



The answer: option 1

We have *constant* lr, so LR is too high. EMA should help

Does EMA bring the optimal LR down enough? No, losses are way high!

Some refs to the literature

Scaling Laws and Compute-Optimal Training Beyond Fixed Training Durations

Alexander Hägele[1*] Elie Bakouch[2] Atli Kosson[1] Loubna Ben Allal[2]
Leandro Von Werra[2] Martin Jaggi[1]
[1]EPFL [2]Hugging Face
*alexander.hagele@epfl.ch

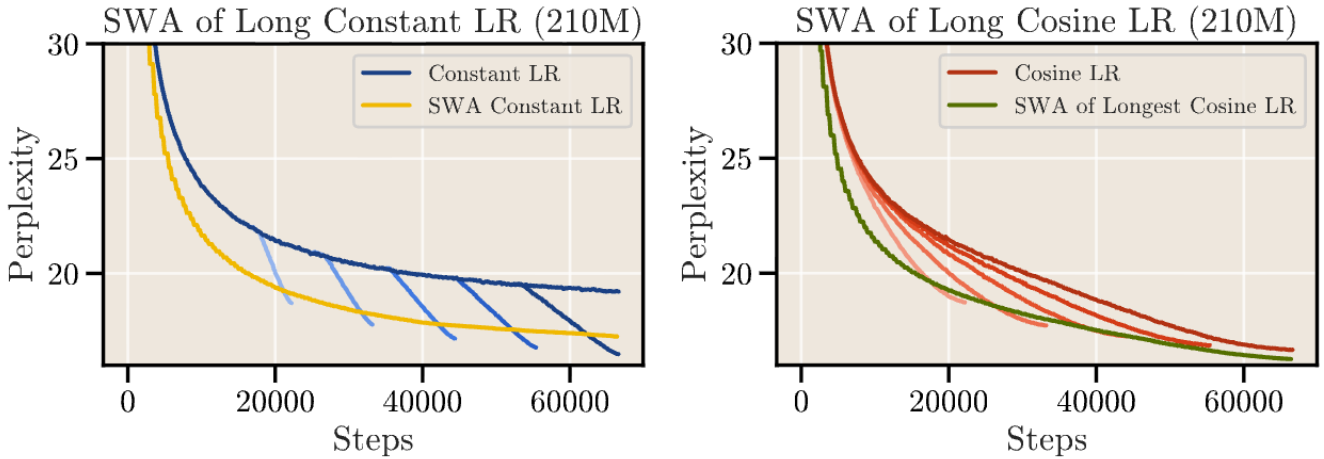


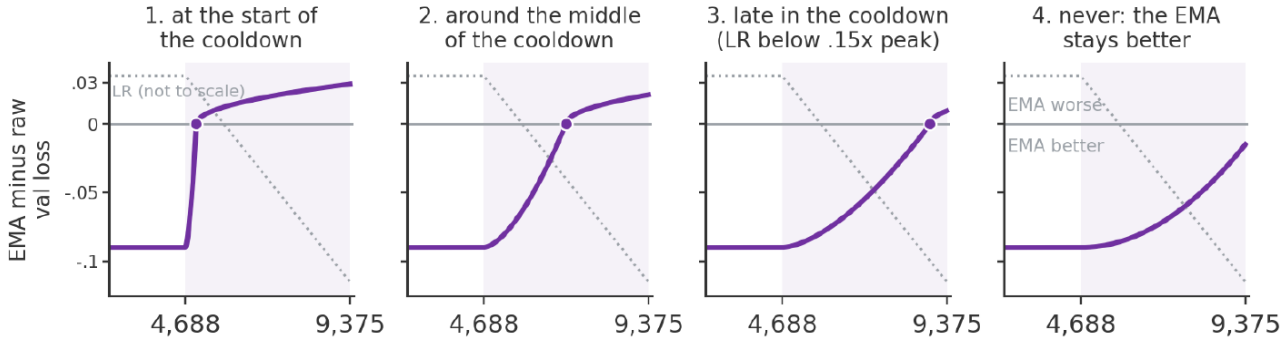
Figure 10: **SWA improves generalization and simulates decayed learning rates.** Using SWA for the constant LR phase (left) strongly boosts the loss, but a gap to the cooldown remains. SWA also improves the generalization of a cosine schedule (right), where the intermediate checkpoints of SWA largely overlap with optimal loss trajectory of shorter cosine runs.

Part b

(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.

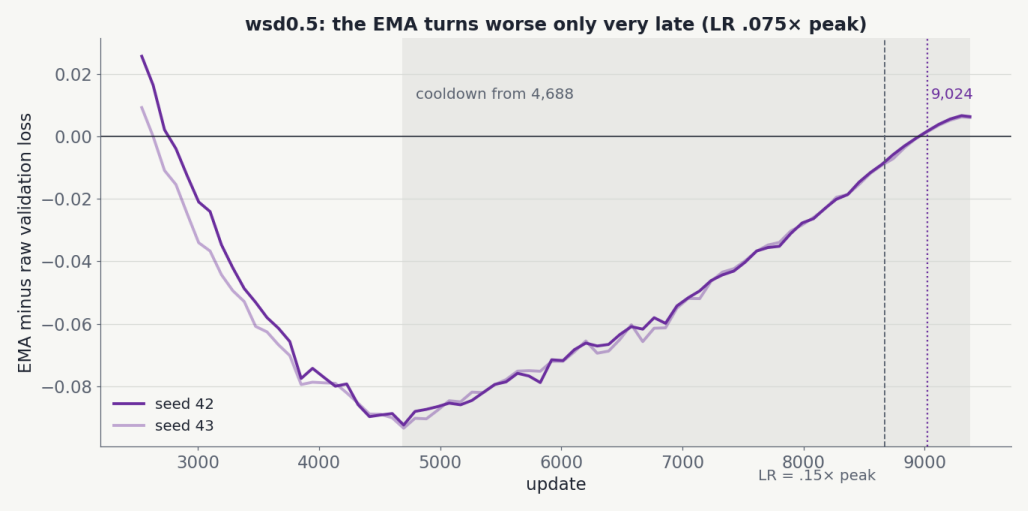
In the `wsd0.5` run, let $\Delta(t)$ be the EMA's validation loss minus the raw weights' at update t. When does Δ become positive (EMA worse) for good?



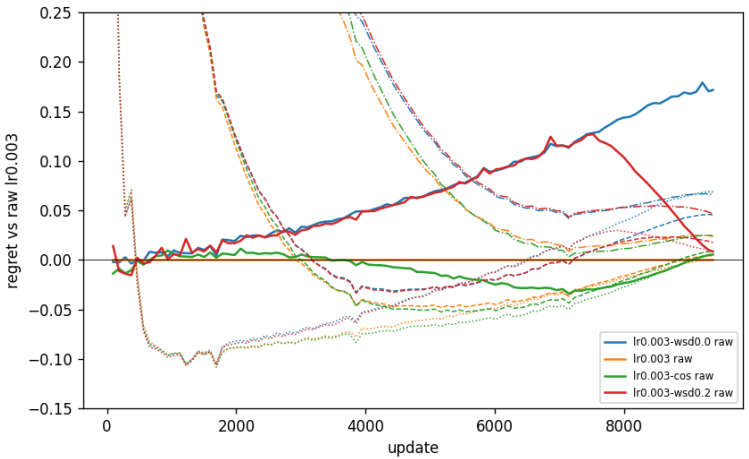
· 1. · 2. · 3. · 4.

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?

Part b answer



of parameters



EMA is helpful mid training run, then regret gets worse only at the end of training

Matches the general A2 results very closely

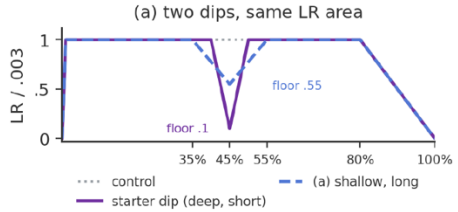
Decay is generally more effective than EMA

Q4

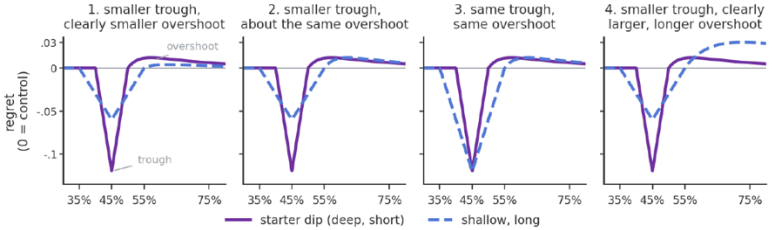
From the assignment (reference only). Recall the dip-and-rewarm experiments of Problem 2(c). A dip multiplies the LR of the `wsd0.2` control (peak .003) by a factor that falls linearly from 1 to a floor and then returns linearly to 1. The starter dip runs from 40% to 50% of training with a floor of 0.1×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

(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).

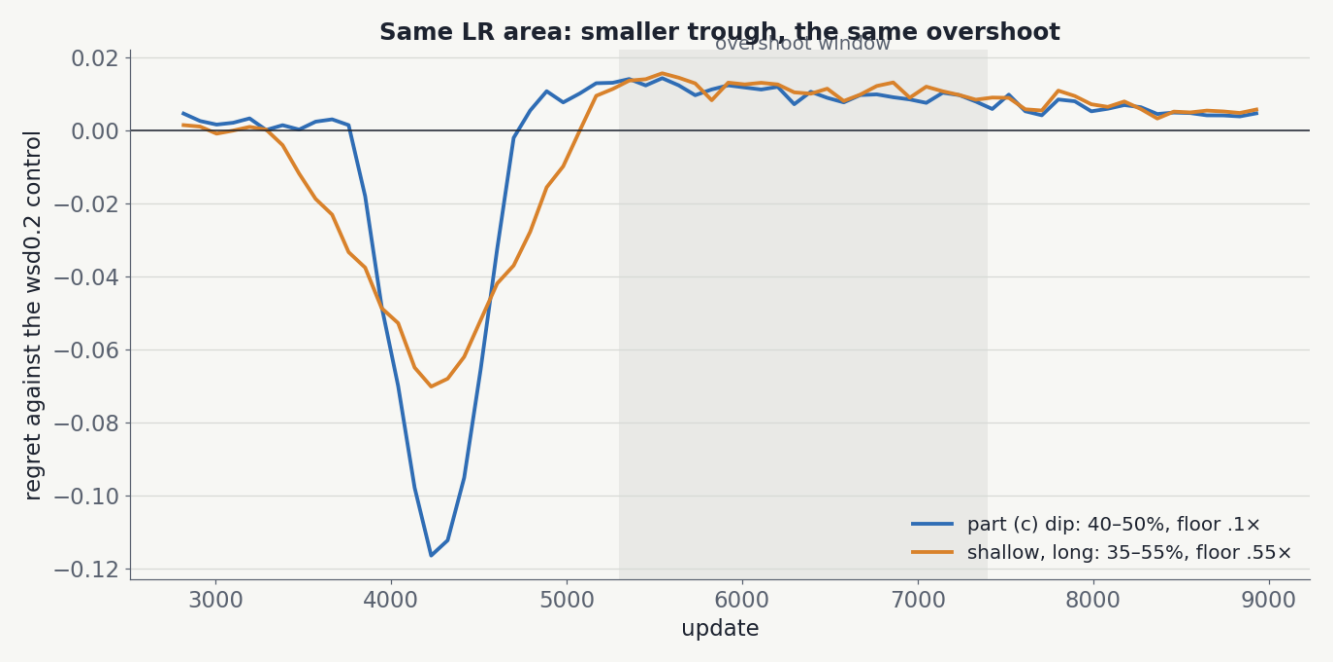


Which sketch matches the two dips' regret curves? ("About the same overshoot": within a factor of 1.5.)



· 1. · 2. · 3. · 4.

Q4 answer



Option 2: smaller trough, similar overshoot

Observe – deep + shallow have the same area. In A2, we saw the area way key to overshoot

Valley itself is controlled by the transient, not the area.

Some interesting stuff from the literature

SCALING LAW WITH LEARNING RATE ANNEALING

Howe Tissue ✉
h-sun20@tsinghua.org.cn

Venus Wang
wangxxing12@gmail.com

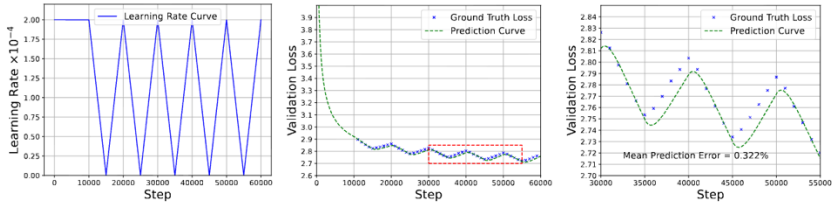
Lu Wang
wangluloveslezhi@gmail.com

Overall, we discover that the validation loss of a language model at any step is determined by two factors: the forward area S_1 under the LR curve and the degree of LR annealing S_2 at that step. Formally, the expectation of loss L at step s of a language model follows::

$$L(s) = L_0 + A \cdot S_1^{-\alpha} - C \cdot S_2,$$
$$S_1 = \sum_{i=1}^{s} \eta_i, \tag{1}$$
$$S_2 = \sum_{i=1}^{s} \sum_{k=1}^{i} (\eta_{k-1} - \eta_k) \cdot \lambda^{i-k},$$

where η_i is the learning rate at step i, and λ is a hyper-parameter representing the decay factor for LR annealing momentum (see Sec. 3 in detail),which typically ranges from 0.99 to 0.999. L_0, A, C, α are undetermined positive constants. S_1 is also known as the summed learning rate (Kaplan et al., 2020), and S_2 represents the LR annealing area. A visualization of S_1 and S_2 is provided as Fig. 1.

(d) Full Loss curve prediction of the WSD LRS (20% cosine annealing to $\eta_{min} = 0$) (Hu et al., 2024).



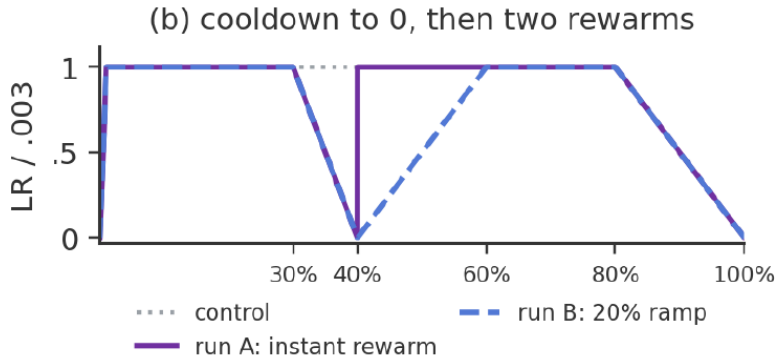
(e) Full Loss curve prediction of the Cyclic LRS (Smith, 2017) including multiple unseen LR re-warmup stages.

Tissue et al, https://arxiv.org/abs/2408.11029

Part b

(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%.



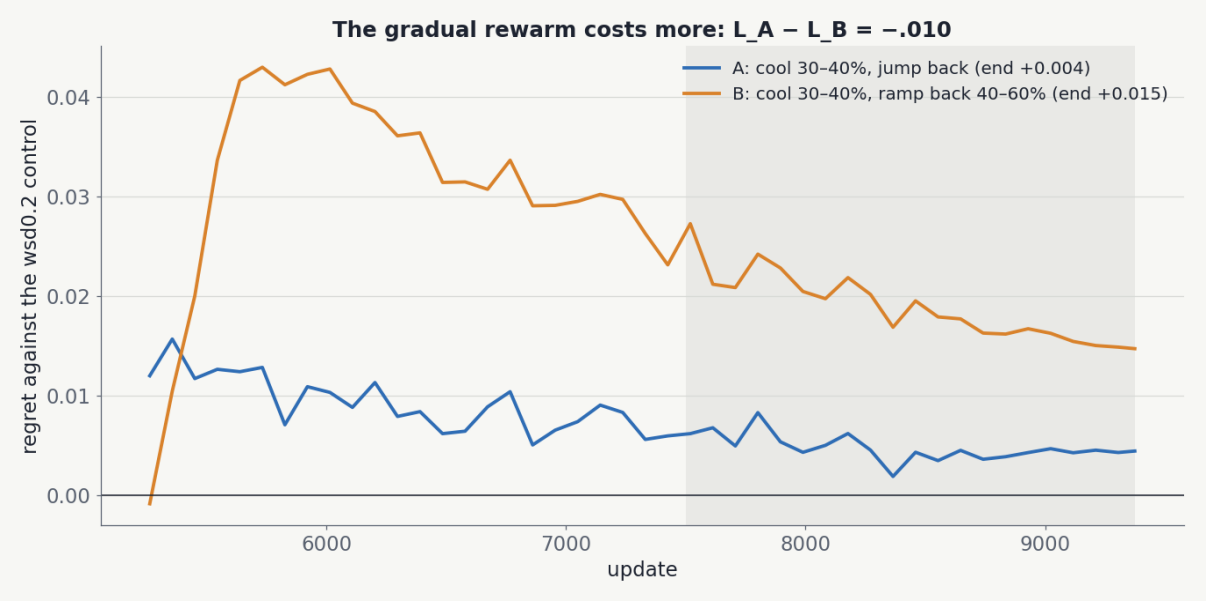
Let L_A and L_B be the final validation losses of runs A and B. Which interval contains $L_A - L_B$?

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

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. $[.005, \infty)$

Part b answer



Gradual rewarm is worse here

Why? Loss of useful high-LR steps.

We're undertrained so we want to spend time at high LRs

Q5

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.

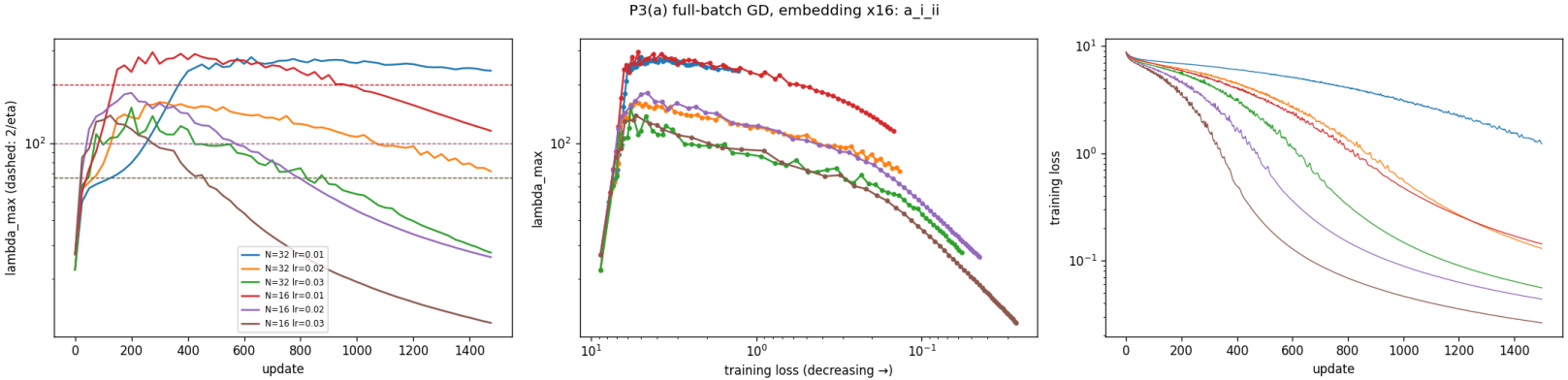
(a) How far above 2 at a large step size (10 points).

This question changes. $N =$ **256** sequences and $\eta =$ **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

Evidence from assignment



Eos-like, but not quite EoS plots

Differences?

- LR seems to affect the target point and plateau
  - Generally overshoots relative to the EoS predictions
- Sample size (and LR) affect entry to the edge and the ‘time on plateau’
- Not perfectly EoS – eventually falls off the edge.

Related stuff

GRADIENT DESCENT ON NEURAL NETWORKS TYPICALLY OCCURS AT THE EDGE OF STABILITY

Jeremy Cohen Simran Kaur Yuanzhi Li J. Zico Kolter[1] and Ameet Talwalkar[2]
Carnegie Mellon University and: [1]Bosch AI [2] Determined AI
Correspondence to: jeremycohen@cmu.edu

Cross entropy losses show deviation from the ‘precise edge’

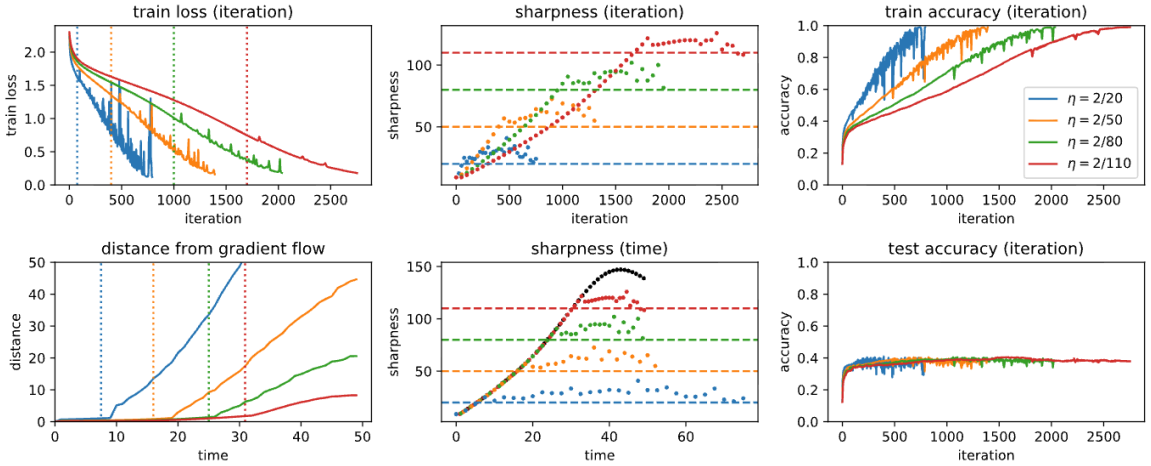


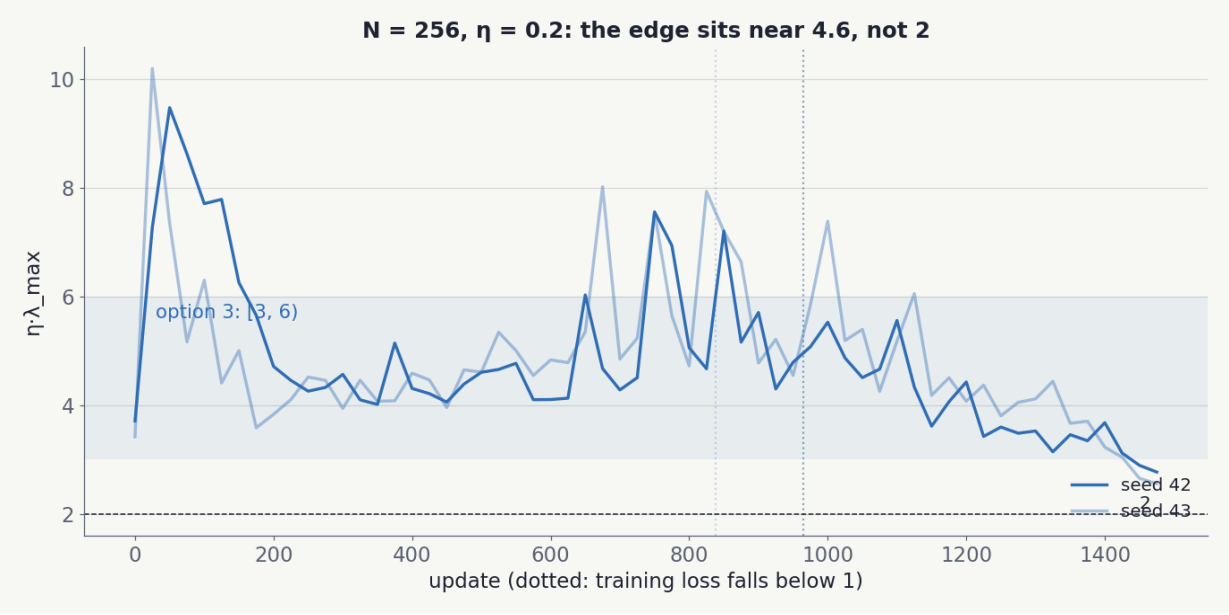
Figure 31: **Gradient descent.** Refer to the Figure 29 caption for more information.

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

Answer/rationale

Overshoots, as we expect
Also, larger sample size means we 'don't fall off the edge'



Part b

(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.

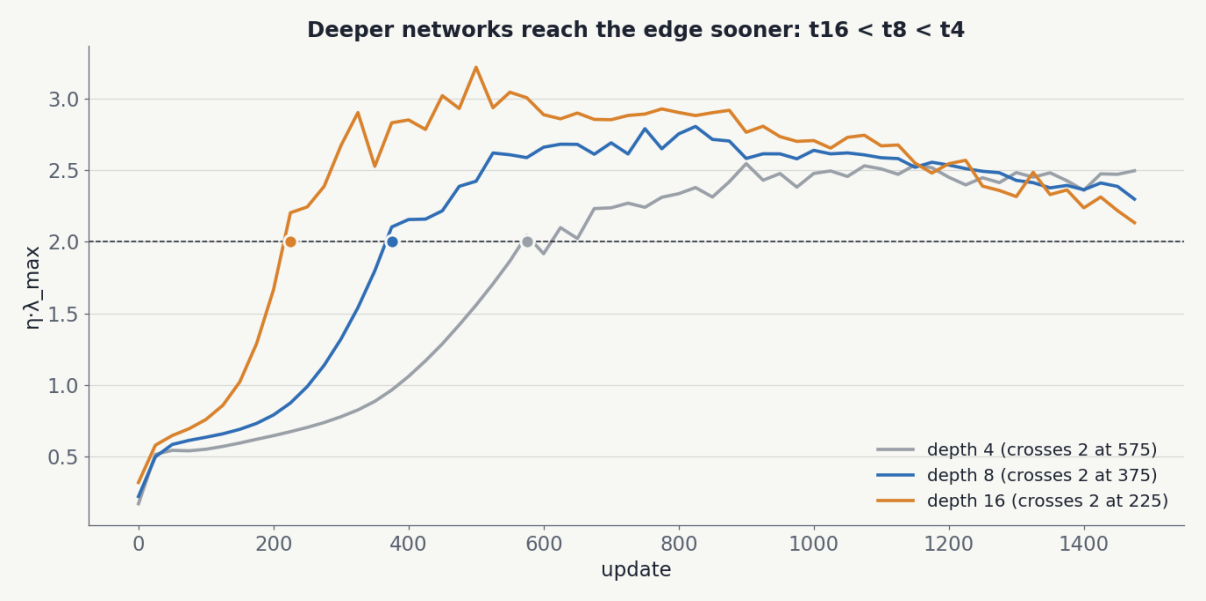
Let t_L be the first update at which $\eta\lambda_{\max} > 2$ at depth L. Which ranking do you predict?

- 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

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

Part b answer



Deeper networks reach the edge sooner.

Also partly associated with the fact that they generally train faster too.

Observation from the EoS paper

D EMPIRICAL STUDY OF PROGRESSIVE SHARPENING

In this appendix, we empirically study how problem parameters such as network width, network depth, and dataset size affect the degree to which progressive sharpening occurs. To study progressive sharpening on its own, without the confounding factor of instability, we train neural networks using gradient flow (Runge-Kutta) rather than gradient descent. Informally speaking, gradient flow does "what gradient descent would do if gradient descent didn't have to worry about instability."

We observe that progressive sharpening occurs to a greater degree: (1) for narrower networks than for wider networks (which is consistent with infinite-width NTK theory), (2) for deeper networks than for shallower networks, and (3) for larger datasets than for smaller datasets.

Q6

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.

(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. $[1.2, \infty)$

Q6 Assignment data

From the assignment: messing with sharpness by messing with weight decay

Training change	Final λ_max	Top eigenvector
weight decay 0	2.55	RMSNorm gains 70%, head 24%
weight decay .1 (default)	5.95	LM head
weight decay .3	14.9	LM head 94%

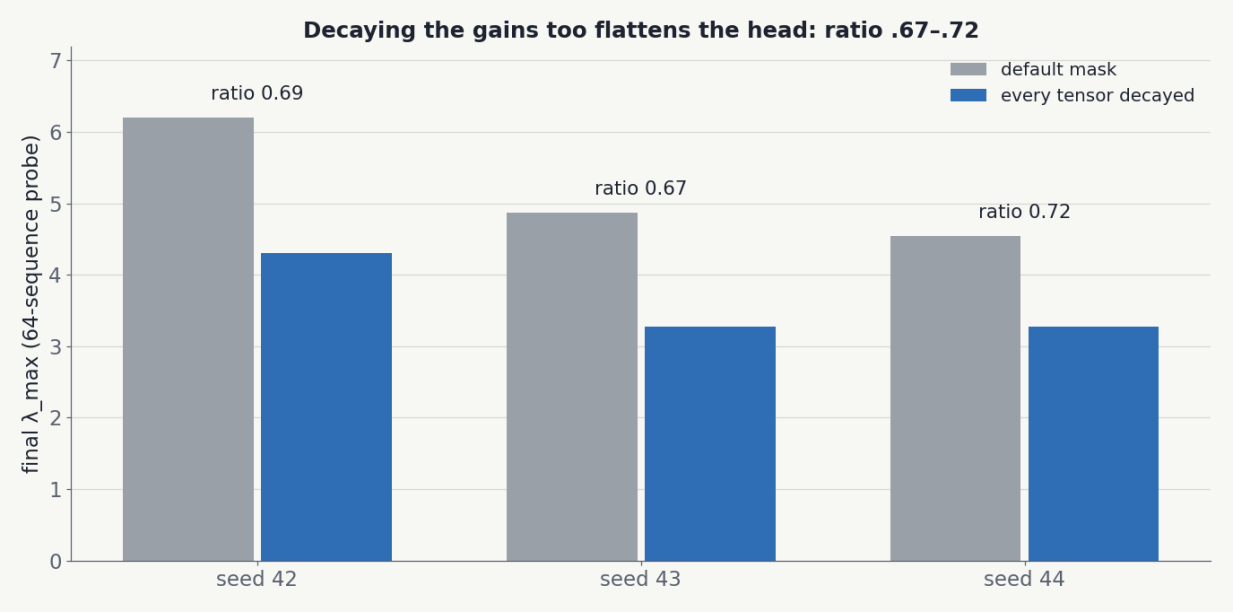
Higher weight decay → higher lambda max

Weight decay shrinks weights, compensated by RMS norm 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. $[1.2, \infty)$

Q6 answer and rationale



Key – if we decay the RMSnorm gains, renormalization is no longer trival! Sharpness goes back down, since we can no longer scale down the matrices

Part b

(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}

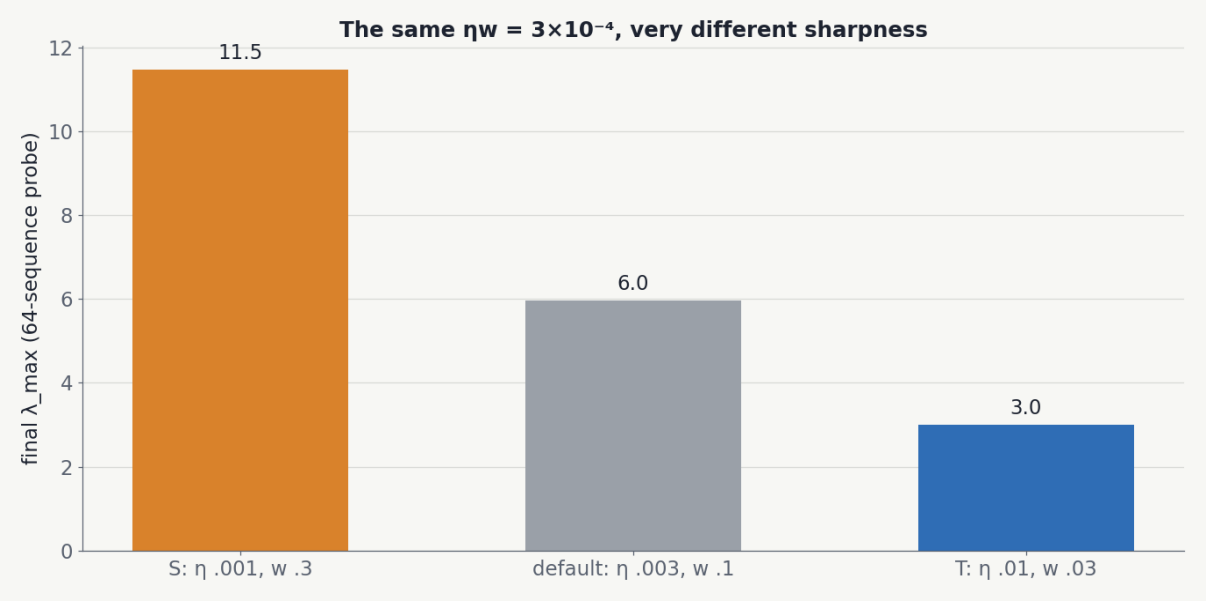
Let λ_S, λ_{def} and λ_T be the final λ_{max} of runs S, Default and T. Which ranking do you predict?

- 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

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

Part b answer



Not a trick question – LR*wd is a useful optimization variable

But, it doesn't control sharpness. Higher WD → higher sharpness.

Q7

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.

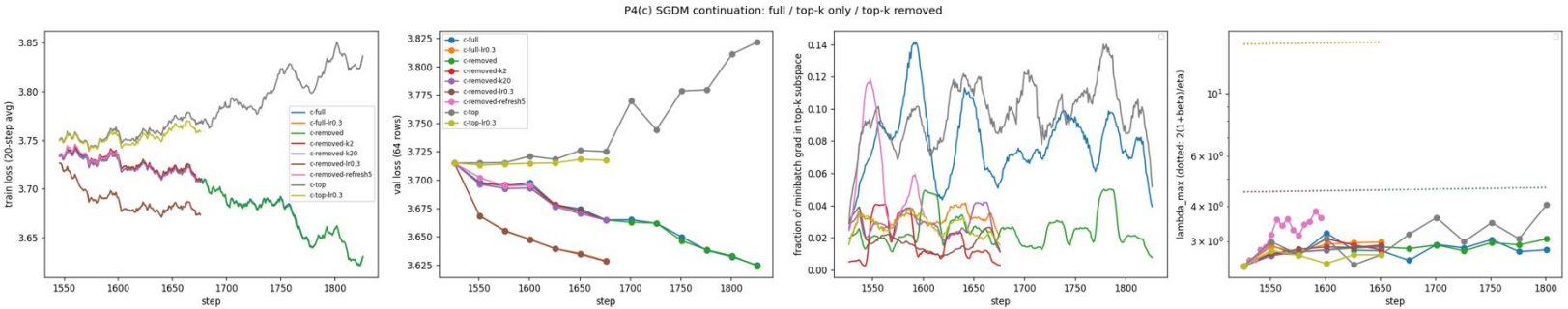
Adam fed only the top subspace (10 points).

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. $[.3, \infty)$

Assignment data

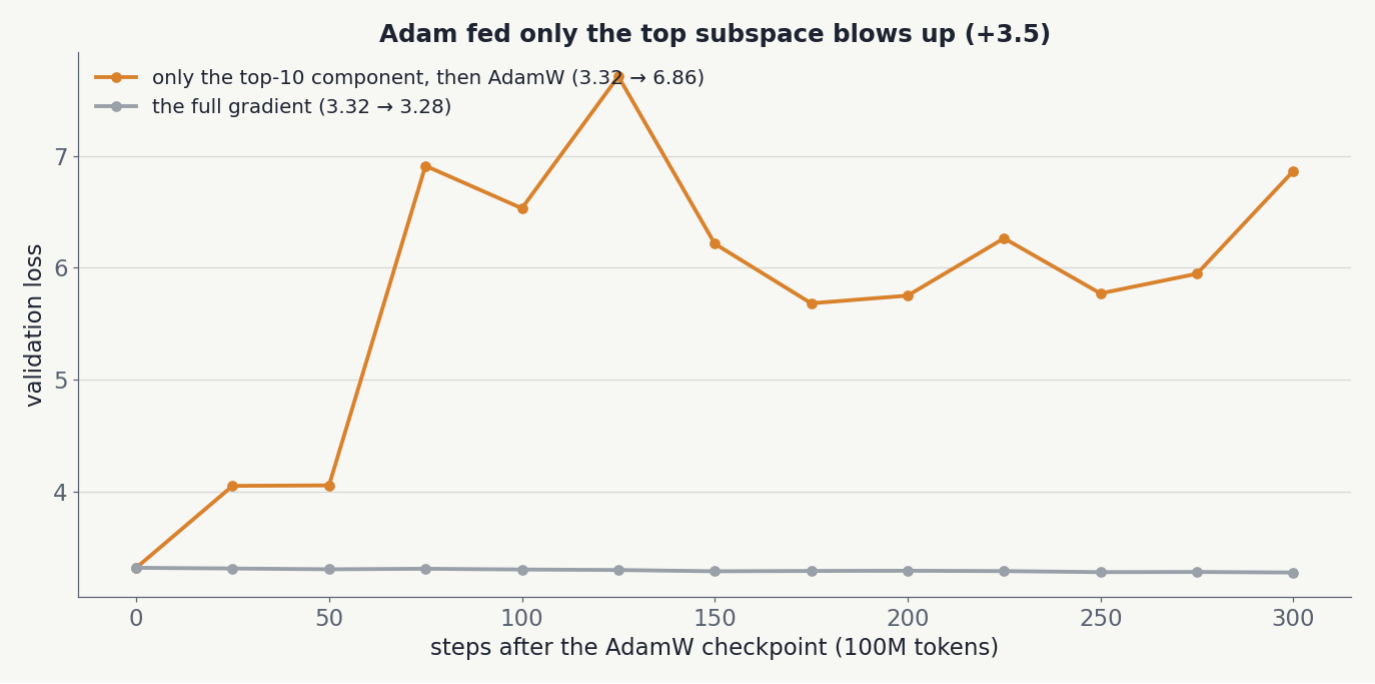


From the assignment: top 10 directions are very small, so using just them does nothing.

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. $[.3, \infty)$

Answer and rationale



Correct option (4): blowup.

Why? Reason about Adam's second moment behavior. Small gradient norm + normalization = blowup