The Optimization Dynamics of Adaptive Gradient Methods

NeurIPS 2023 Heavy Tails Workshop

Jeremy Cohen // Carnegie Mellon University

Based in part on work at Google

Optimization dynamics

- This talk is about the *dynamics of optimization* in deep learning:
  - how the optimizer moves around the weight space,
  - and how this depends on (a) loss landscape, and (b) hyperparameters

Why do we care?

- Consider these goals:
    - Design new optimizers for deep learning
    - Design optimal strategy for setting hyperparameters
    - Understand why different optimizers behave differently
    - Understand what the optimizer hyperparameters *do*
- Need to understand dynamics of optimization first!

Always start simple

- Always understand simple things before more complicated things
- Understand deterministic (full-batch) training, before stochastic training
- Understand gradient descent, before Adam / adaptive methods

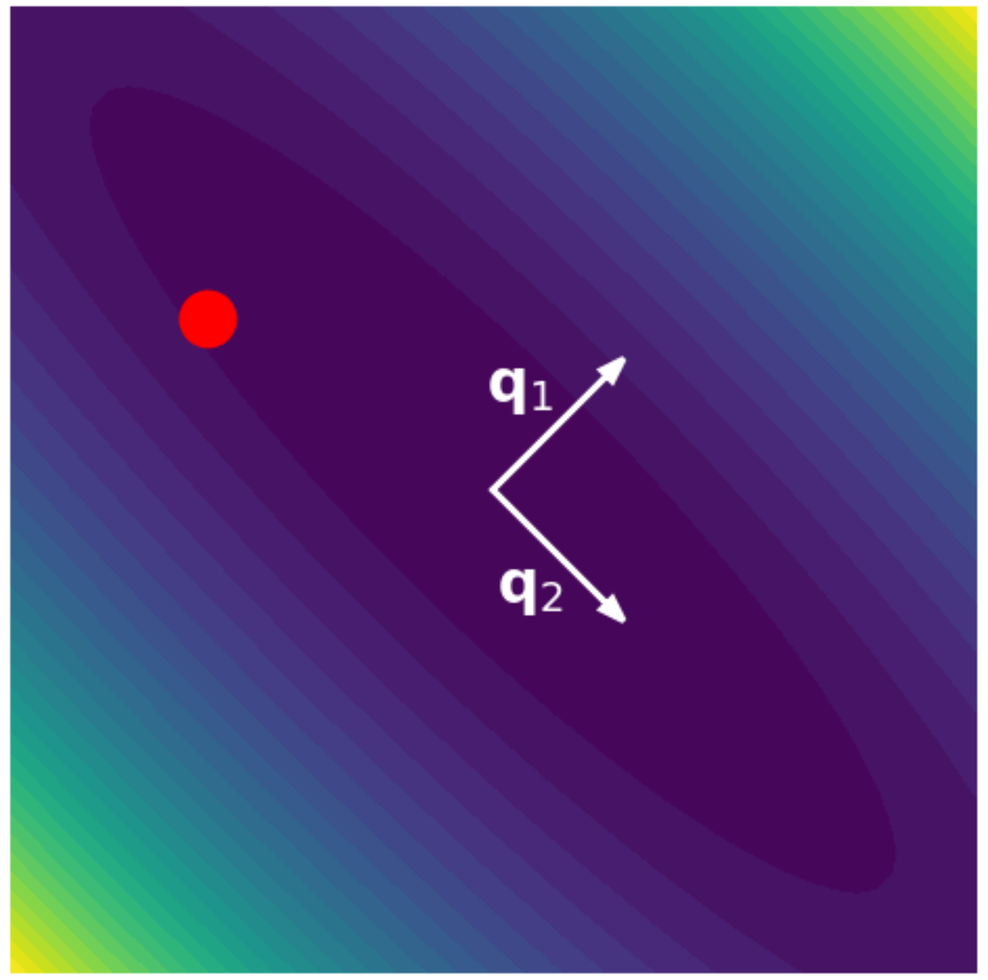
Gradient descent

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \nabla L(\mathbf{w}_t)$$

- What path does gradient descent take?
- What does the learning rate parameter η do?

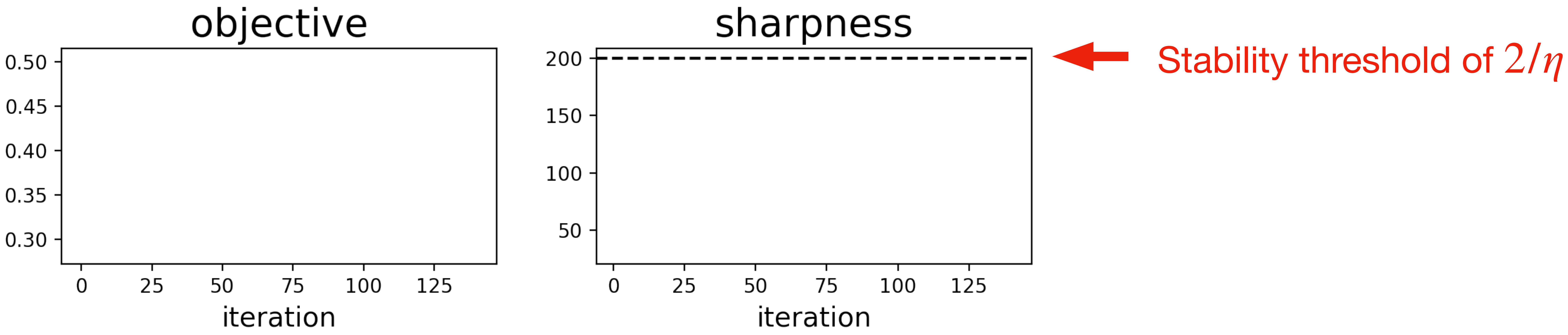
Experiment: train CNN using GD with $\eta = 0.01$

- Let sharpness = largest eigenvalue of loss Hessian
- On quadratic functions: if sharpness $> 2/\eta \Rightarrow$ GD blows up



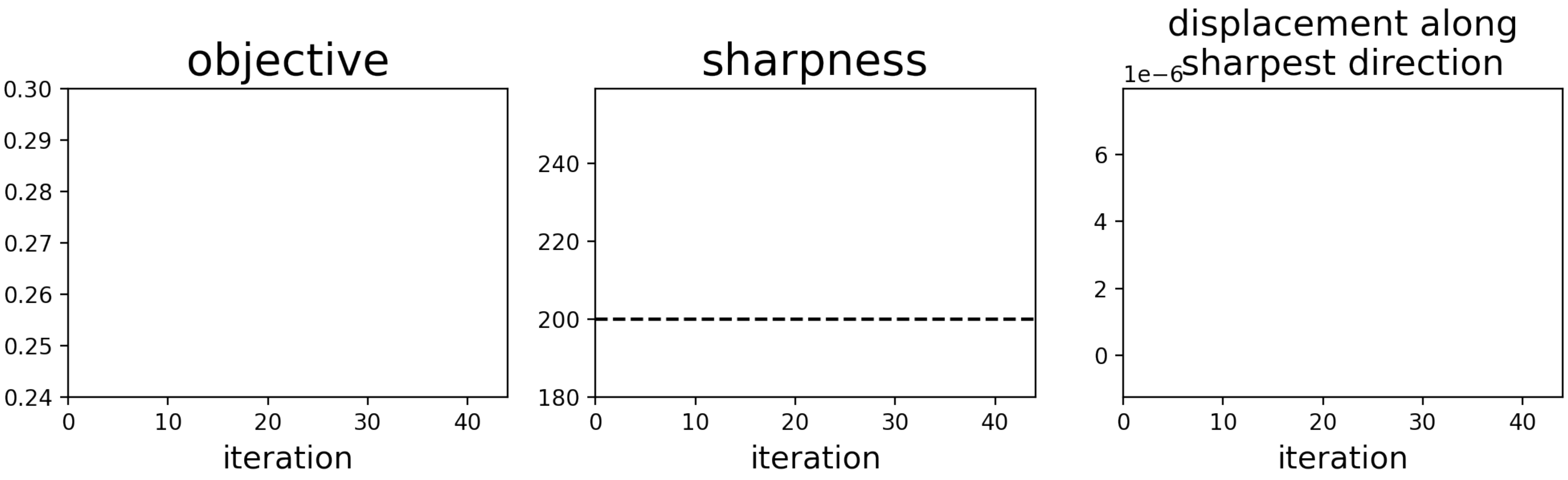
Experiment: train CNN using GD with $\eta = 0.01$

- Let sharpness = largest eigenvalue of loss Hessian
- On quadratic functions: if sharpness > $2/\eta$ $\Rightarrow$ blowup



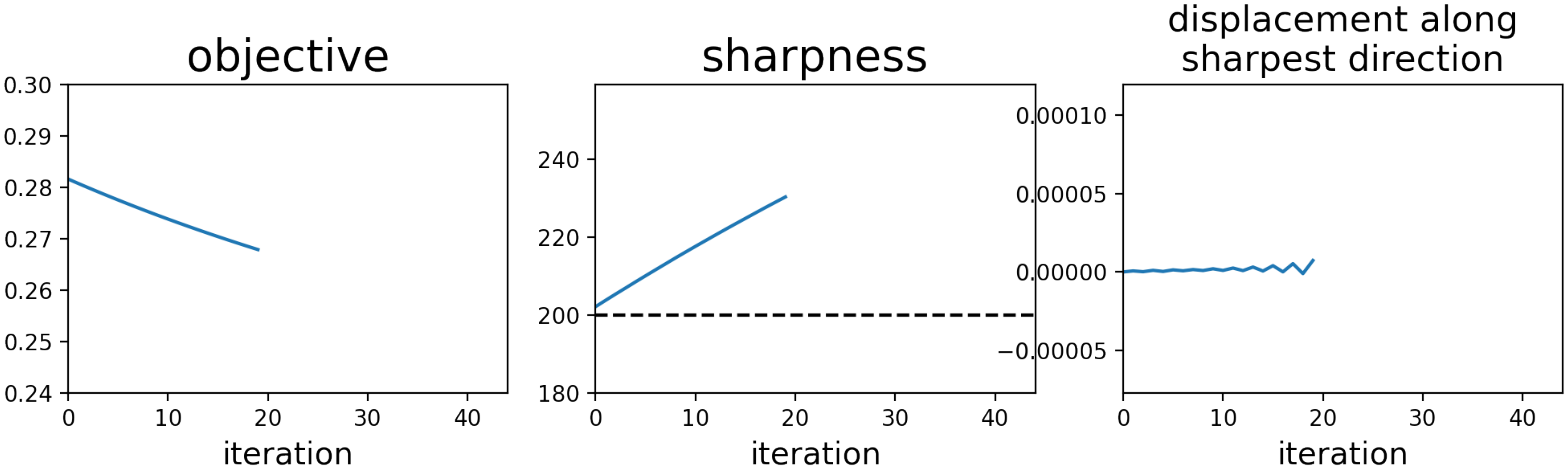
What happens next?

Run gradient descent for a few more steps



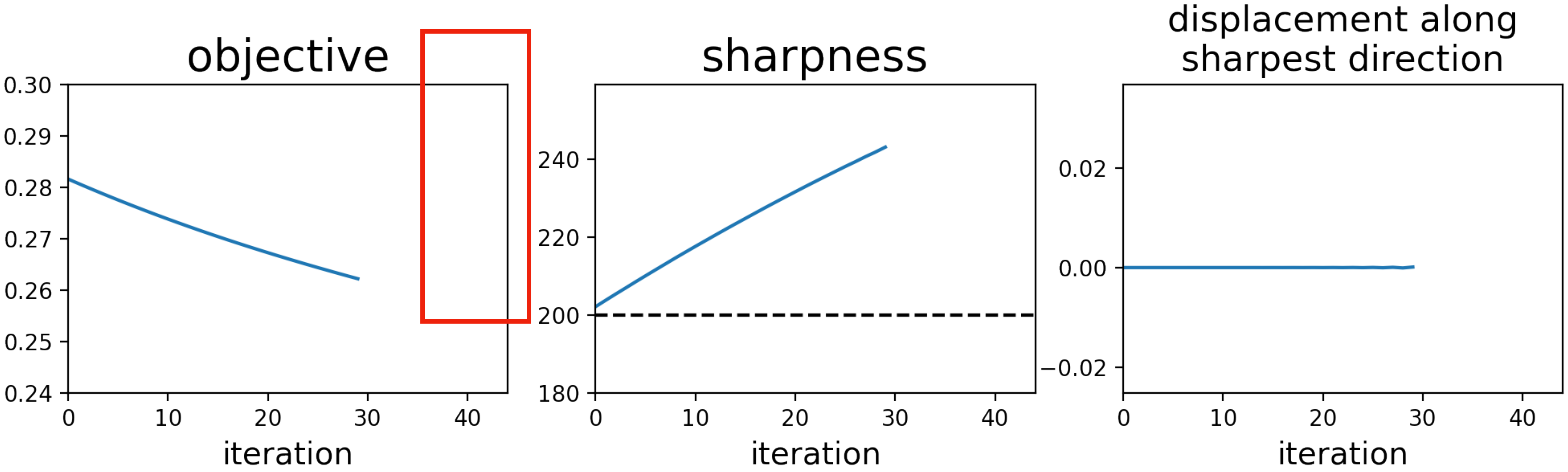
As expected, the optimizer starts to diverge along the sharpest direction

Run gradient descent for a few more steps



The oscillations are growing exponentially in magnitude

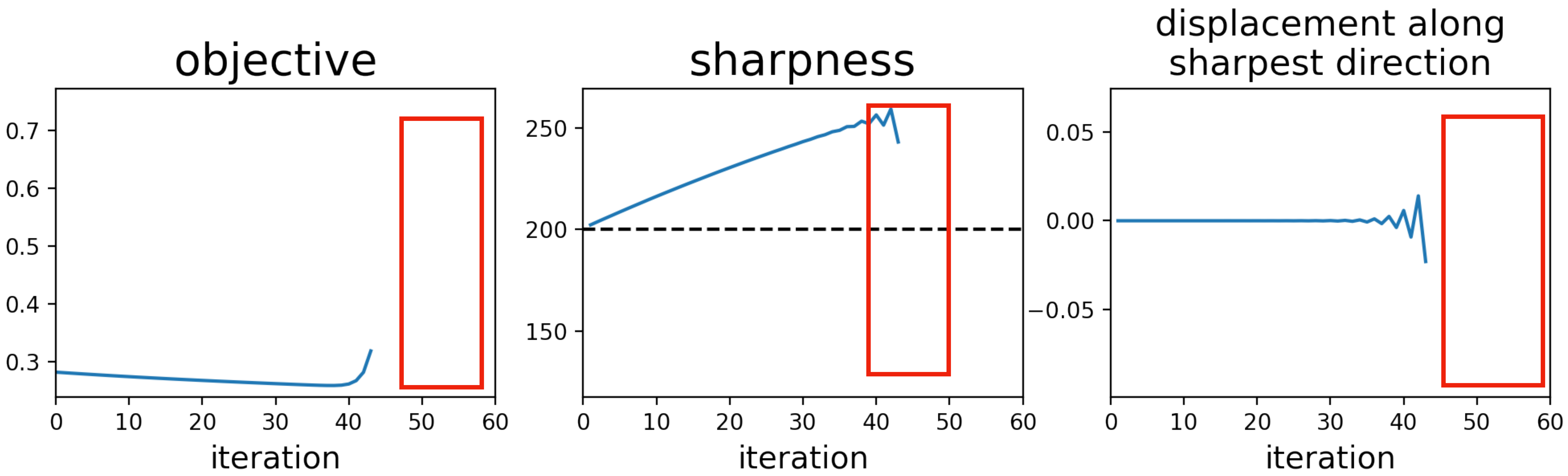
Run gradient descent for a few more steps



Eventually, the oscillations get big enough that the objective goes up 😧

What happens next?

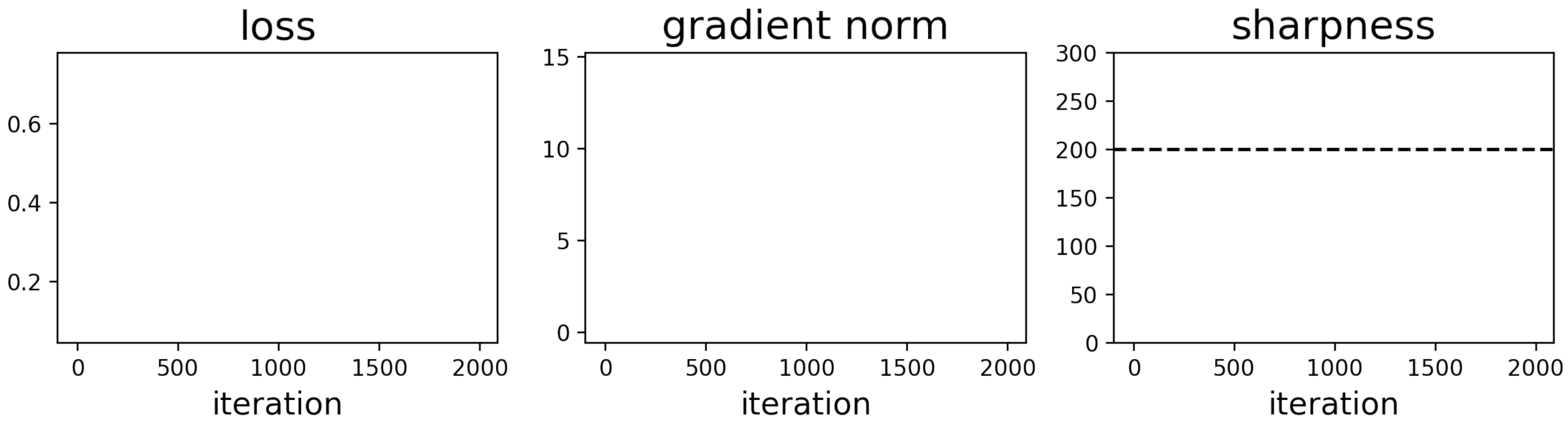
What happens next?



As if by magic, the sharpness drops below $2/\eta = 200$

Subsequently, the optimizer oscillates back inward

Full training trajectory of the CNN on CIFAR-10



Whenever the sharpness rises above $2/\eta$, it somehow gets pushed back down

Training happens at the "edge of stability"

Relevant publications

C, Simran Kaur, Yuanzhi Li, J. Zico Kolter, Ameet Talwalkar. “Gradient descent on neural networks typically occurs at the edge of stability.” ICLR 2021.

Stanislaw Jastrzébski, Maciej Szymczak, Stanislav Fort, Devansh Arpit, Jack Tabor, Kyunghyun Cho, and Krzysztof Geras. “The break-even point on optimization trajectories of deep neural networks.” ICLR 2020.

Aitor Lewkowicz, Yasaman Bahri, Ethan Dyer, Jascha Sohl-Dickstein, and Guy Gur-Ari. “The large learning rate phase of deep learning: the catapult mechanism.” arXiv 2020.

Lei Wu, Chao Ma, Weinan E. “How SGD selects the global minima in over-parameterized learning: a dynamical stability perspective.” NeurIPS, 2018.

“Self-stabilization”

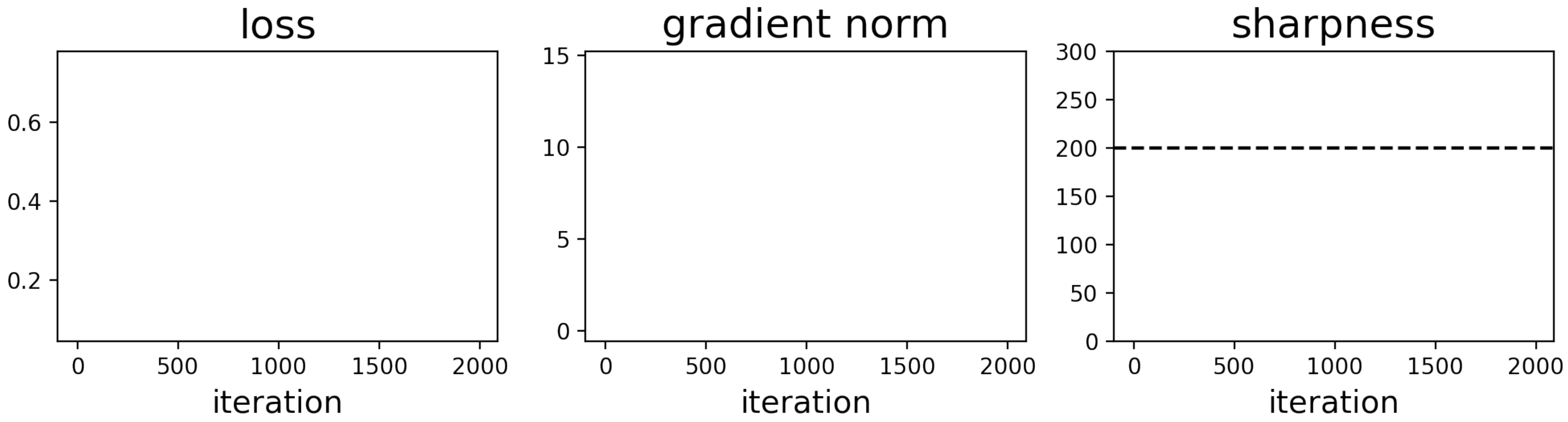
- Damian et al [2022] explained why the sharpness goes down.
- They showed that this behavior is *not* specific to the structure of neural net objectives, but is a generic property of gradient descent.

Alex Damian*, Eshaan Nichani*, Jason Lee. “Self-stabilization: the implicit bias of gradient descent at the edge of stability.” ICLR 2022.

The main idea

- Damian et al [2022] showed that after starting to diverge, the loss gradient always aligns with *the gradient of the sharpness itself*.
  - To see this, need to take a local *cubic* Taylor expansion.
- Thus, following the negative loss gradient automatically reduces sharpness!
- Gradient descent has an inbuilt self-regulatory mechanism: if the sharpness is too high, gradient descent oscillates … but these oscillations automatically push the sharpness back down!

The full training trajectory



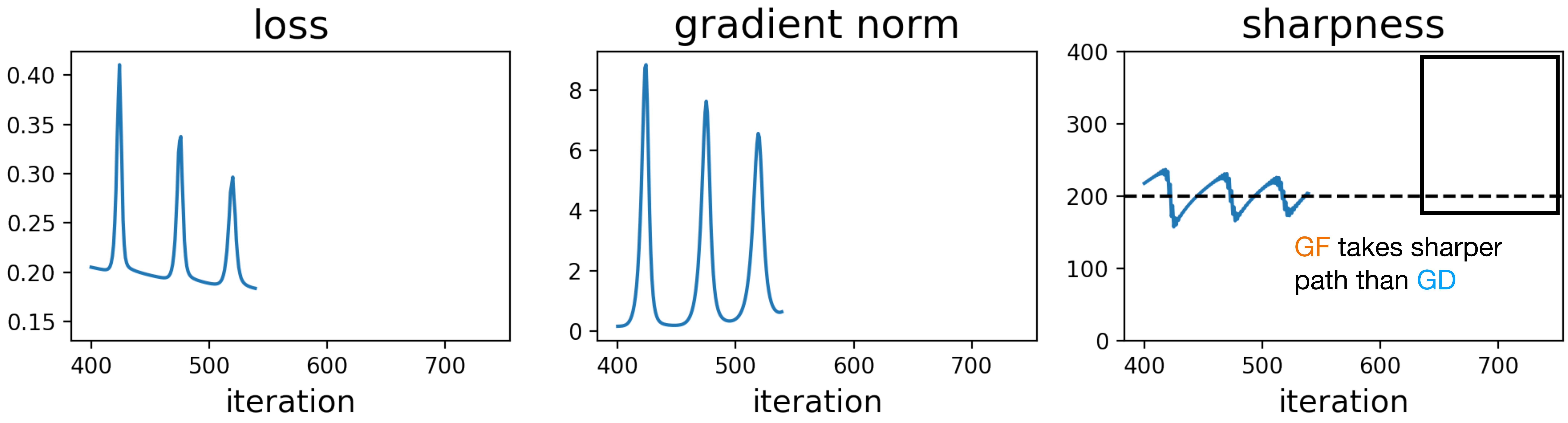
Whenever sharpness goes above $2/\eta$, gradient descent oscillates … but these oscillations automatically push the sharpness back down!

What path does gradient descent take?

- People often conceive of gradient descent as approximating the gradient flow trajectory $\dot{\mathbf{w}}(t) = -\eta \nabla L(\mathbf{w})$
- But this does not hold in the EOS regime!

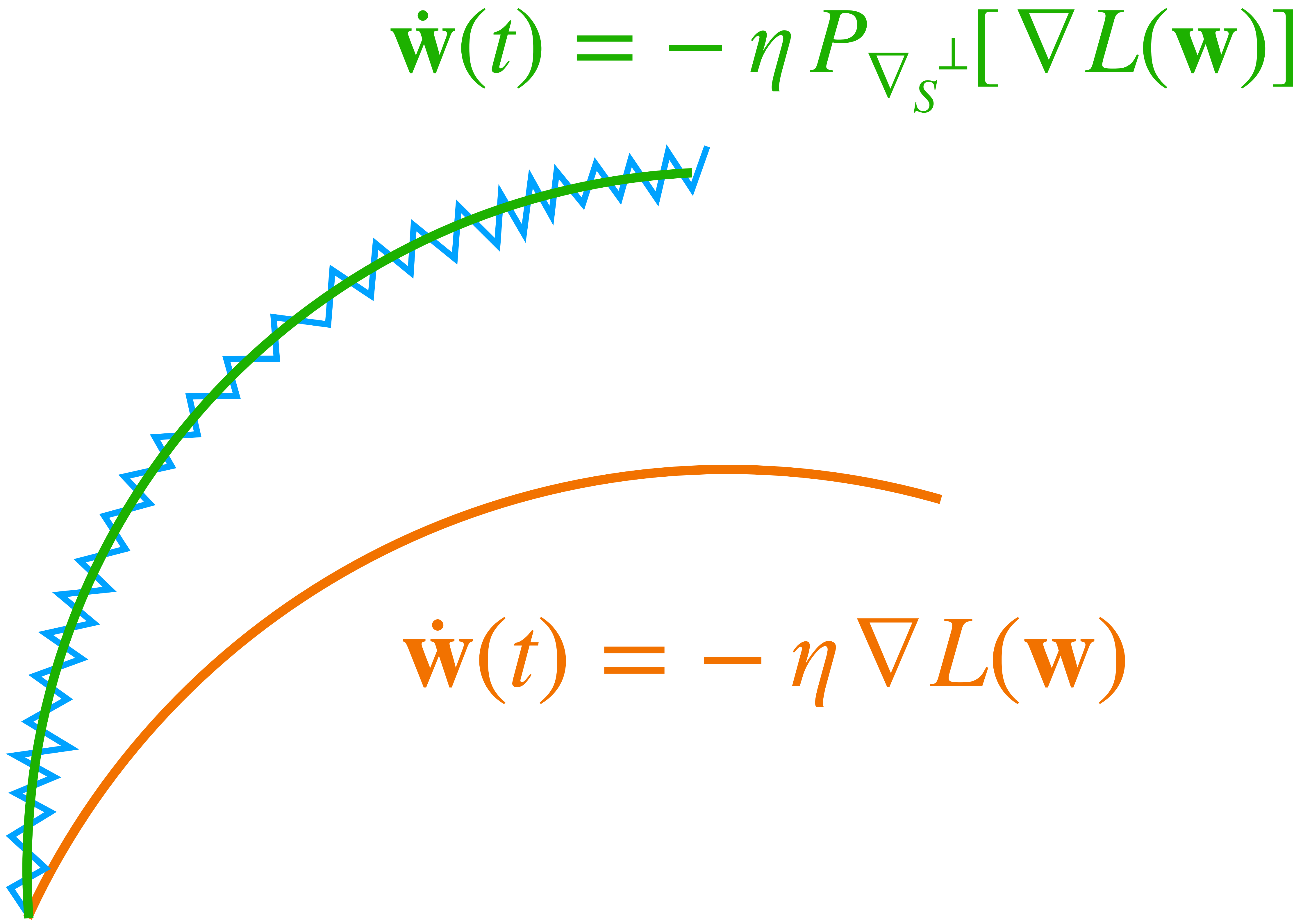
Gradient descent takes a different path than gradient flow

- Experiment: compare gradient flow (in orange) to gradient descent (in blue)



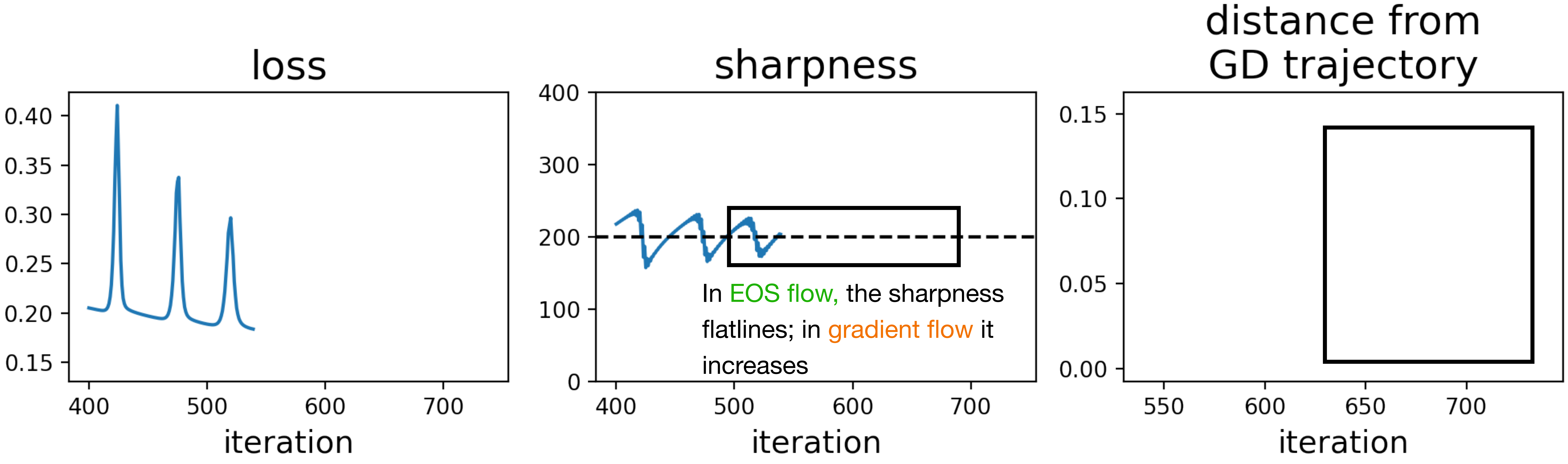
A cartoon

- Gradient flow takes one path
- Gradient descent at EOS takes a different, oscillatory path
- It oscillates around the EOS flow
- The EOS flow removes the oscillations <u>while retaining their effect</u>



The EOS flow

- The EOS flow matches the trajectory of gradient descent, whereas gradient flow takes a different path.



The distance between gradient flow and GD grows over time; the distance between EOS flow and GD doesn't.

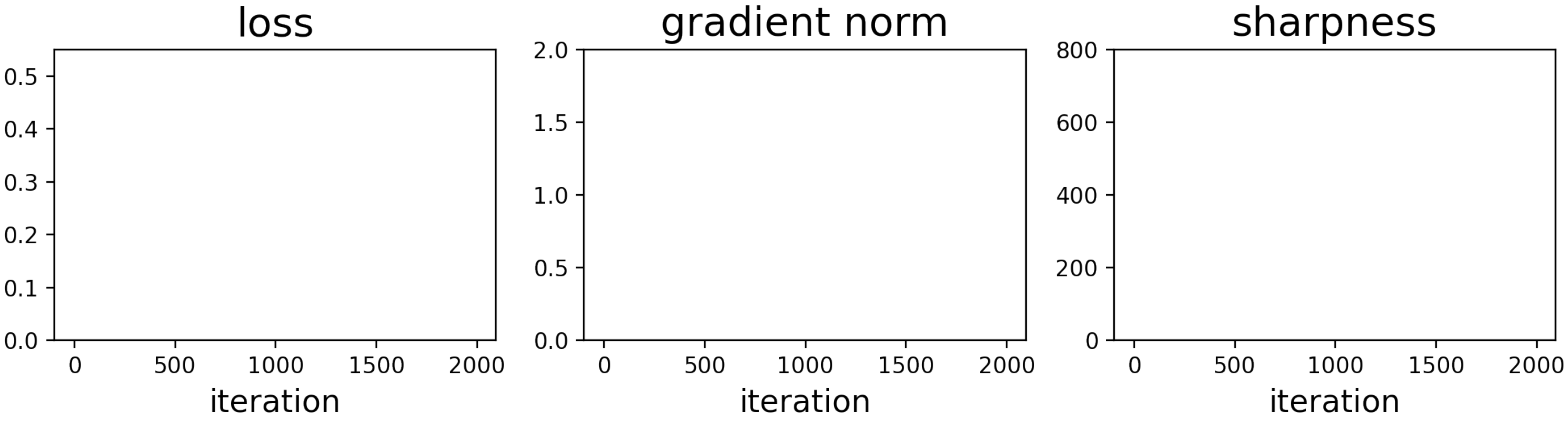
What about adaptive optimizers?

- In practice, many neural nets are trained using adaptive methods (like Adam)
- Let's start with RMSProp (Adam with no momentum or bias correction):

$$\boldsymbol{\nu}_{t+1} = \beta_2 \boldsymbol{\nu}_t + (1 - \beta_2) \nabla L(\mathbf{w}_t)^{\circ 2}$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \, \text{diag}(\boldsymbol{\nu}_{t+1})^{-1/2} \nabla L(\mathbf{w}_t)$$

An RMSProp trajectory ($\eta = $ 1e-5)



- Sharpness shows no clear pattern, but might a different quantity?

The adaptive edge of stability

- The key is to interpret RMSProp as preconditioned gradient descent with a changing preconditioner.

- Preconditioned gradient descent:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \, \mathbf{P}^{-1} \, \nabla L(\mathbf{w}_t)$$

- This algorithm diverges on quadratics if the "preconditioned sharpness" is too high.

$$\lambda_1(\mathbf{P}^{-1}\,\mathbf{H}\,) > 2/\eta$$

largest eigenvalue of *preconditioned* Hessian

The adaptive edge of stability

- The key: interpret RMSProp as preconditioned gradient descent with a changing preconditioner.

$$\boldsymbol{\nu}_{t+1} = \beta_2 \boldsymbol{\nu}_t + (1 - \beta_2) \nabla L(\mathbf{w}_t)^{\circ 2}$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \, \text{diag}(\boldsymbol{\nu}_{t+1})^{-1/2} \nabla L(\mathbf{w}_t)$$

The adaptive edge of stability

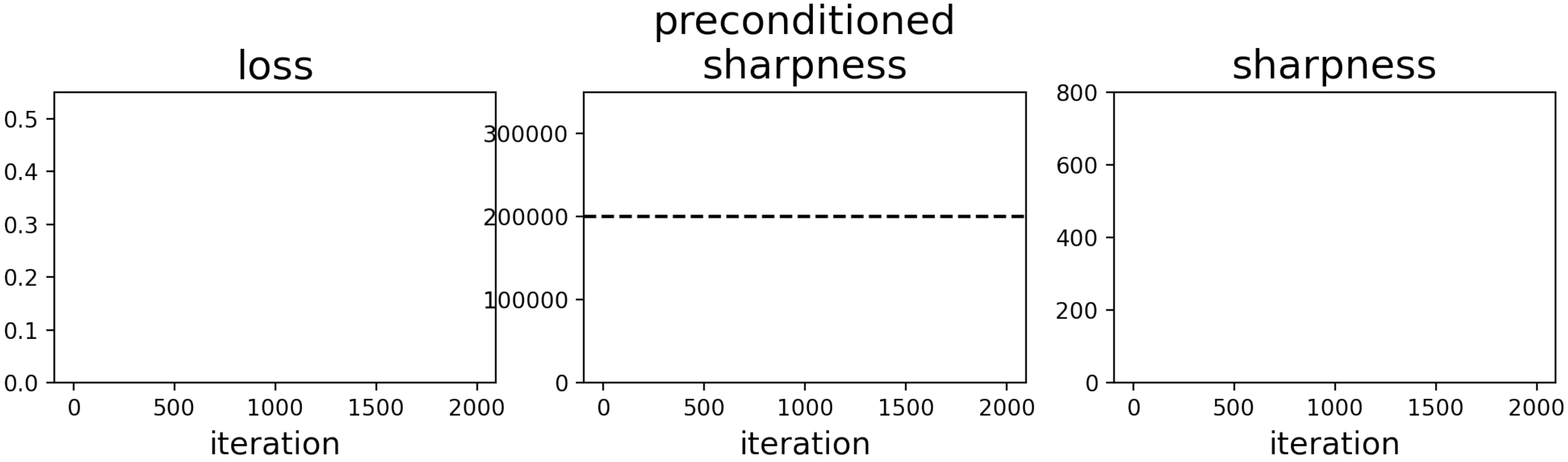
- The key: interpret RMSProp as preconditioned gradient descent with a changing preconditioner.

$$\boldsymbol{\nu}_{t+1} = \beta_2 \boldsymbol{\nu}_t + (1 - \beta_2) \nabla L(\mathbf{w}_t)^{\circ 2}$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \, \underbrace{\text{diag}(\boldsymbol{\nu}_{t+1})^{-1/2}}_{= \mathbf{P}_{t+1}} \nabla L(\mathbf{w}_t)$$

RMSProp with η = 1e-5

- Now plot the *preconditioned sharpness* $\lambda_1(\mathrm{diag}(\boldsymbol{\nu}_t)^{-1/2}\,\nabla^2 L(\mathbf{w}_t))$



- The preconditioned sharpness equilibrates right at $2/\eta$

What about Adam?

$$\boldsymbol{\nu}_{t+1} = \beta_2 \boldsymbol{\nu}_t + (1 - \beta_2) \nabla L(\mathbf{w}_t)^{\circ 2}$$

$$\mathbf{m}_{t+1} = \beta_1 \mathbf{m}_t + (1 - \beta_1) \nabla L(\mathbf{w}_t)$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \, \mathsf{diag}(\boldsymbol{\nu}_{t+1})^{-1/2} \mathbf{m}_{t+1}$$

What about Adam?

- Adam can be viewed as preconditioned *momentum* GD with a changing preconditioner.

$$\boldsymbol{\nu}_{t+1} = \beta_2 \boldsymbol{\nu}_t + (1 - \beta_2) \nabla L(\mathbf{w}_t)^{\circ 2}$$

$$\mathbf{m}_{t+1} = \beta_1 \mathbf{m}_t + (1 - \beta_1) \nabla L(\mathbf{w}_t)$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \, \text{diag}(\boldsymbol{\nu}_{t+1})^{-1/2} \mathbf{m}_{t+1}$$

What about Adam?

- Adam can be viewed as preconditioned *momentum* GD with a changing preconditioner.
- The analogous momentum algorithm diverges on the local quadratic Taylor approximation whenever:

$$\boldsymbol{\nu}_{t+1} = \beta_2 \boldsymbol{\nu}_t + (1-\beta_2)\nabla L(\mathbf{w}_t)^{\circ 2}$$

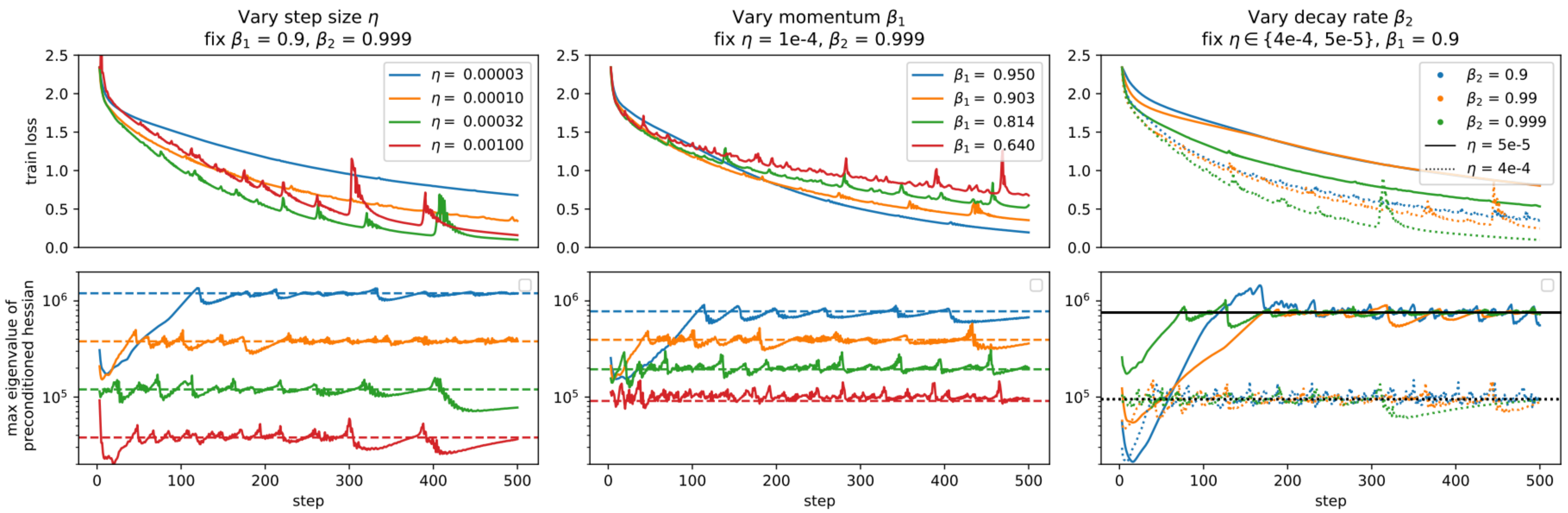
$$\mathbf{m}_{t+1} = \beta_1 \mathbf{m}_t + (1-\beta_1)\nabla L(\mathbf{w}_t)$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta\, \mathrm{diag}(\boldsymbol{\nu}_{t+1})^{-1/2}\, \mathbf{m}_{t+1}$$

$$\lambda_1(\mathrm{diag}(\boldsymbol{\nu}_t)^{-1/2}\, \nabla^2 L(\mathbf{w}_t)) > \frac{2+2\beta_1}{1-\beta_1}\frac{1}{\eta}$$

Adam at the edge of stability

As we train using Adam, the largest eigenvalue of the preconditioned Hessian $\mathrm{diag}(\boldsymbol{\nu}_t)^{-1/2}\,\nabla^2 L(\mathbf{w}_t)$ equilibrates at $\frac{2(1+\beta_1)}{1-\beta_1}\frac{1}{\eta}$, drawn below in dashed lines.

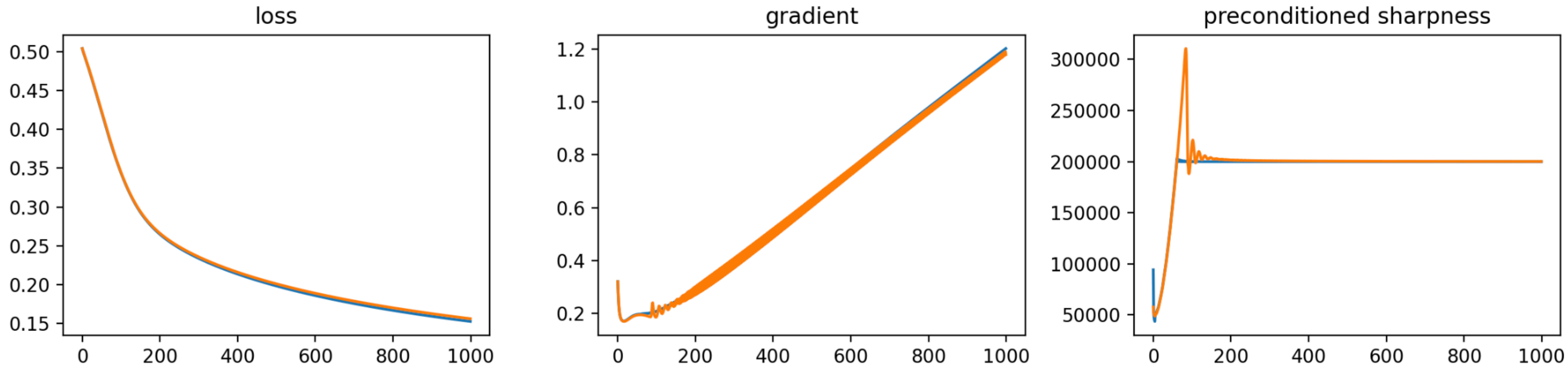


Understanding the adaptive EOS

- Whereas gradient descent maintains stability only by regularizing sharpness…
- … adaptive optimizers maintain stability *both* by regularizing sharpness *and* by adapting the preconditioner
- The tradeoff between these two is controlled by the hyperparamers of the algorithm
- In ongoing work (coming soon!), Alex Damian and I are making this precise.
  - We are deriving an EOS flow for RMSProp which runs through the oscillatory trajectory taken by the optimizer.

Preview of ongoing work (coming soon)

- orange = real RMSProp trajectory, blue = EOS flow



Some relevant papers

C, Simran Kaur, Yuanzhi Li, J. Zico Kolter, Ameet Talwalkar. “Gradient descent on neural networks typically occurs at the edge of stability.” ICLR 2021.

Alex Damian*, Eshaan Nichani*, Jason Lee. “Self-stabilization: the implicit bias of gradient descent at the edge of stability.” ICLR 2022.