CSC 2541: Neural Net Training Dynamics

Lecture 12 - Closing Thoughts

Roger Grosse

University of Toronto, Winter 2021

Empirical Deep Learning

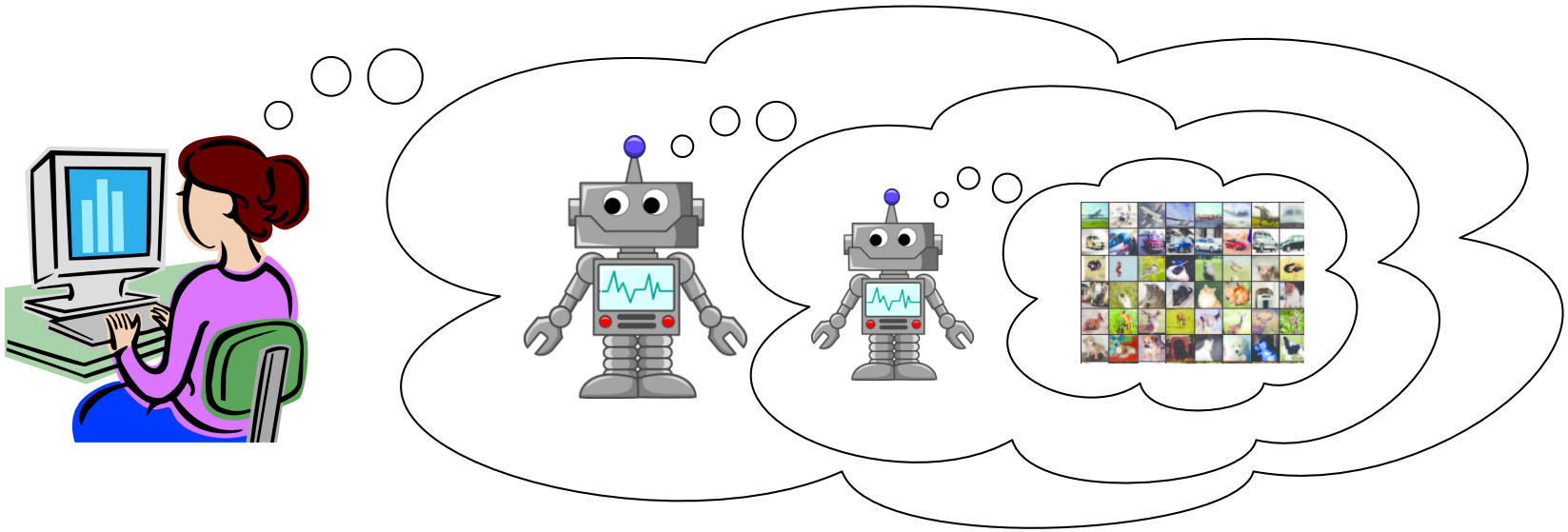
- Studying neural nets presents an unusual set of scientific challenges
- AI used to feel like engineering
  - Start with a goal (optimization, prediction, etc.), ask how a rational agent would solve it, and figure out how to implement that solution
- Now we're doing a lot more reverse engineering
  - The neural net somehow (apparently) solves a problem, and we have to figure out how
- This course didn't give the answers, but it did cover some conceptual tools we need to look for the answers
  - Linearization, metrics, implicit regularization, stochasticity, infinite limits, dynamical systems, etc.

Bilevel Optimization

- Much of the progress of AI has been about automating aspects of AI engineering
  - Hand-coded knowledge $\Rightarrow$ statistical learning
  - Hand-coded reasoning $\Rightarrow$ SAT solvers, probabilistic inference
  - Feature engineering $\Rightarrow$ deep learning
- What's next?
  - Hyperparameters, optimizers, architectures, regularizers, curricula, data augmentation strategies, self-supervised learning objectives, search algorithms, debiasing
- In principle, much of this can be formulated as bilevel optimization
- Our understanding of bilevel optimization is comparable to deep learning circa 2008. Things sometimes work if we get lucky

Bilevel Optimization and NNTD

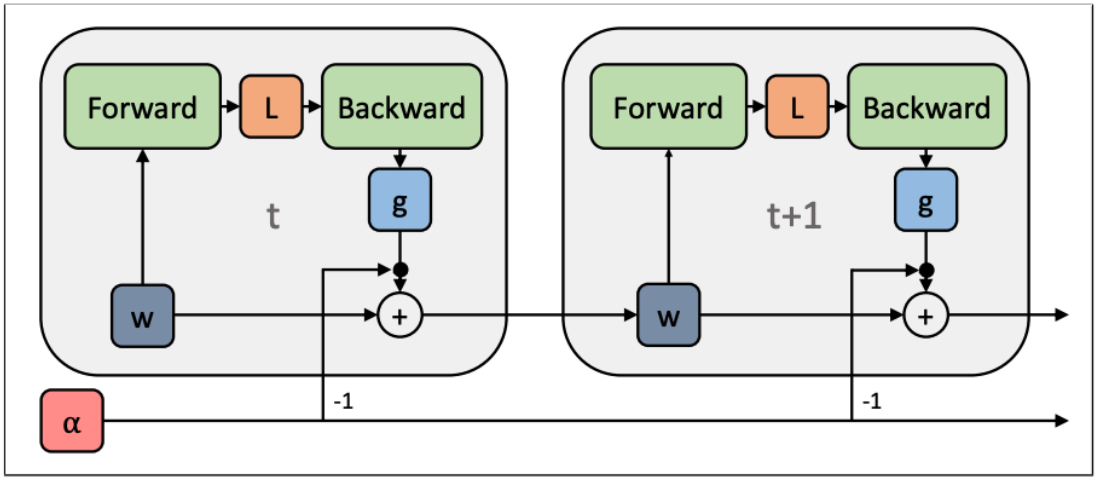
- Understanding bilevel optimization, meta-learning, etc. requires thinking about NNTD in both the inner and outer levels



Short-Horizon Bias in Stochastic Meta-Optimization

Short-Horizon Bias

- We saw that it's possible (in principle) to learn an optimizer using meta-descent
- Can we solve the much easier problem of adapting the learning rate? If we can't even do this, then meta-optimization is hopeless!
- We'll even ignore the computational cost of the meta-optimization itself and just ask if it gives a reasonable solution.



What actually happens?

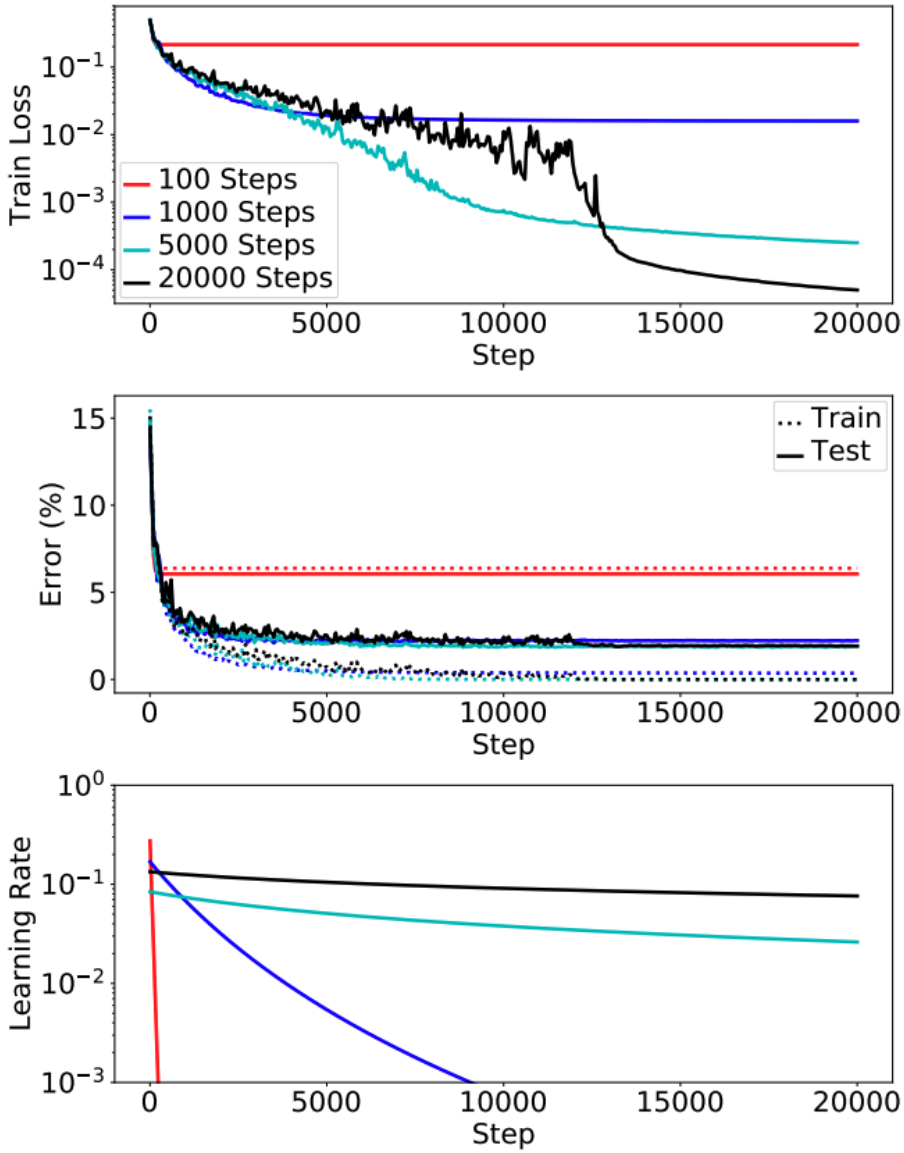
Short-Horizon Bias

Wu et al., 2018, "Understanding short-horizon bias in stochastic meta-optimization"

- Offline adaptation of a parametric learning rate schedule for SGD

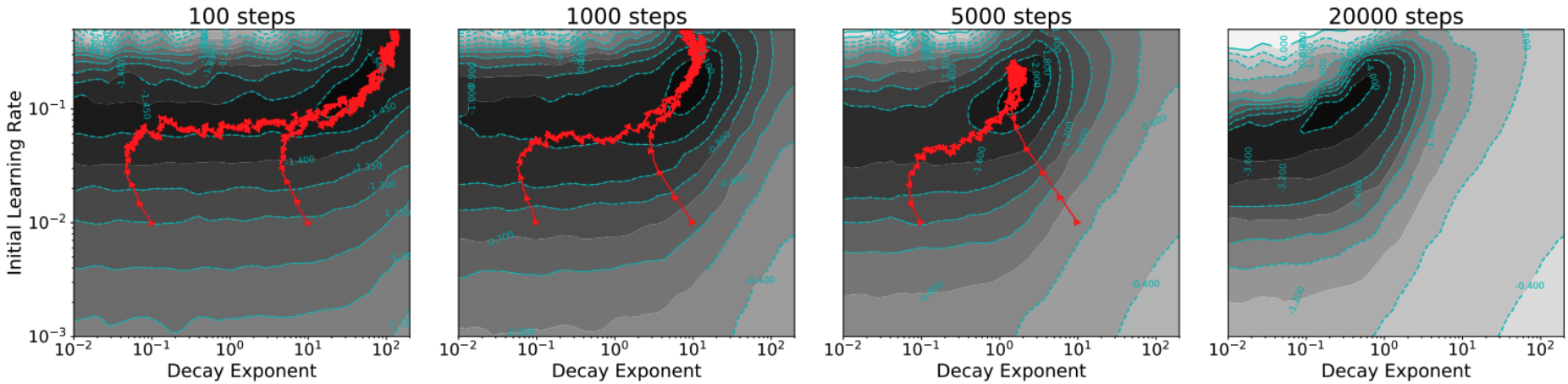
$$\alpha_k = \frac{\alpha_0}{(1 + k/K)^\beta}$$

- Hyperparameters α_0, K, β
- Estimate hypergradient with unrolling
- Evaluate the validation loss after $\{100, 1000, 5000, 20000\}$ steps (the horizon)
- MNIST dataset



Short-Horizon Bias

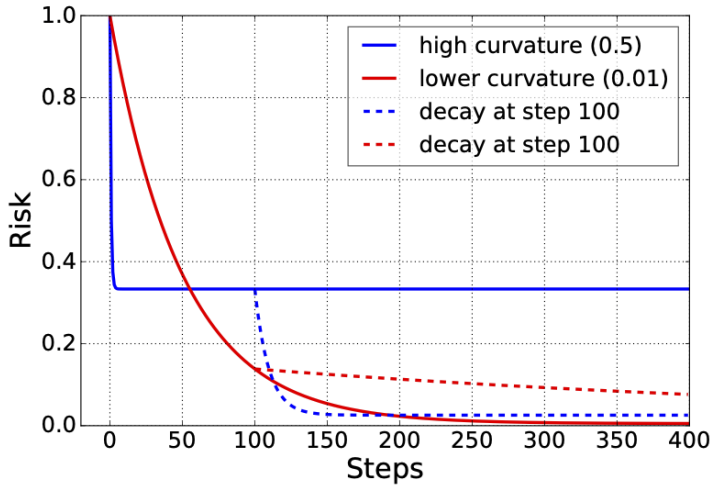
Hyperparameter trajectories for different horizons

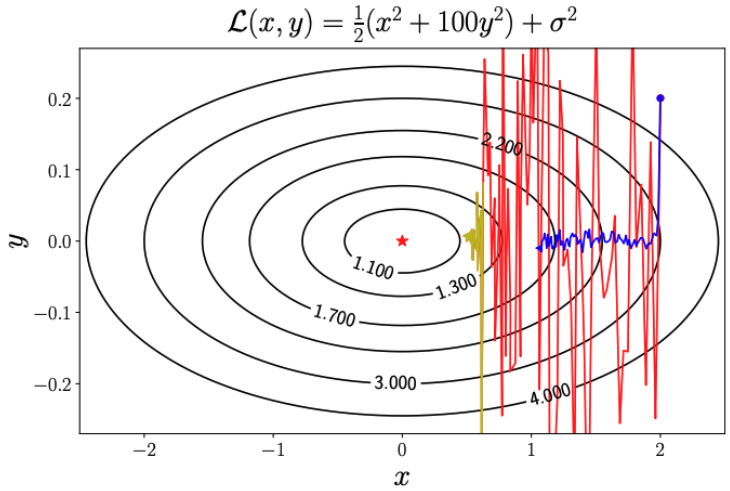


Why does this happen?

Short-Horizon Bias

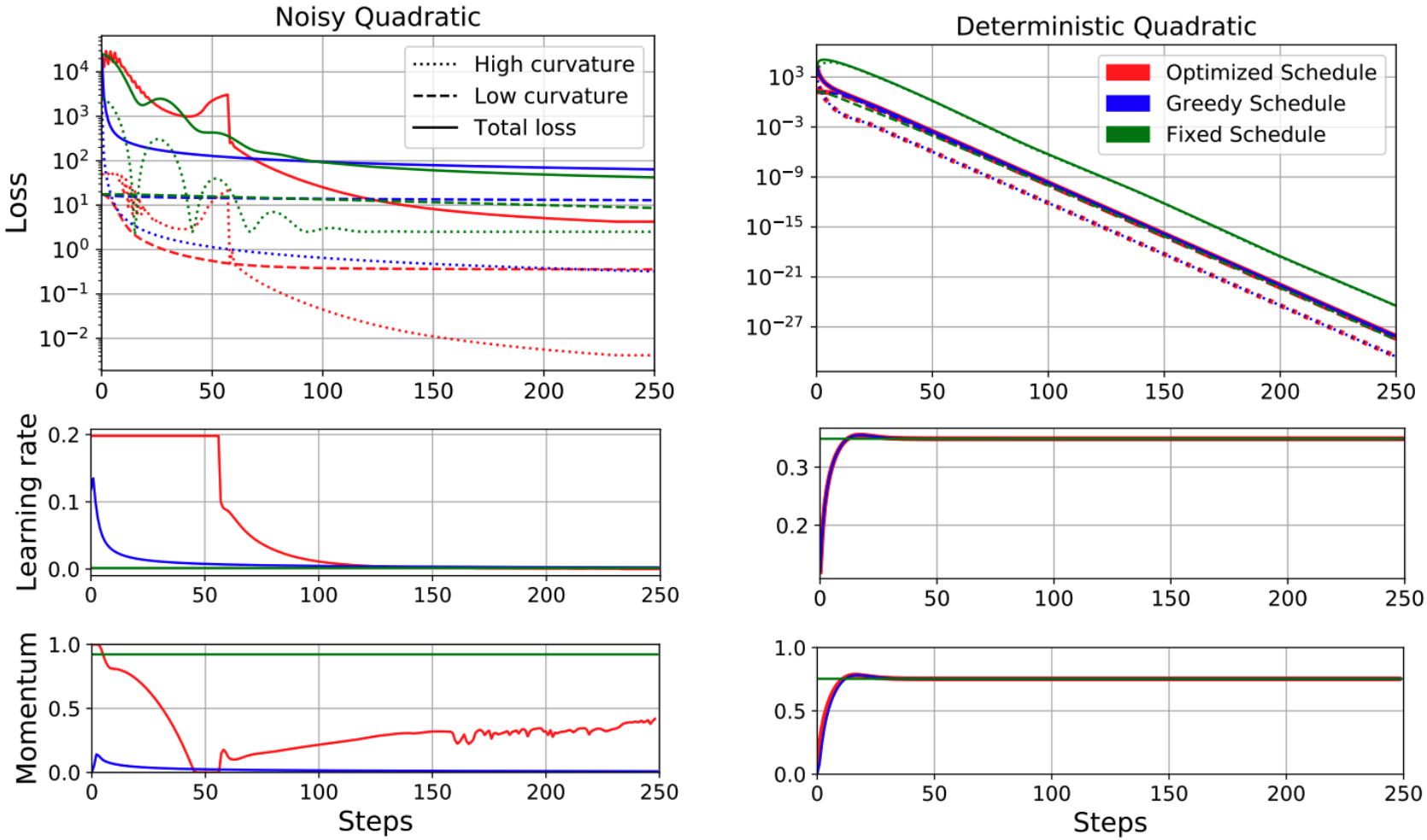
- Remember the Noisy Quadratic Model? (Lecture 7)
- This model captures the phenomenon
  - Over a short horizon, you want to use a small learning rate to reduce the effects of gradient noise
  - Over a long horizon, you want to keep a high learning rate (to make more progress in low-curvature directions) and then decay at the end (to eliminate noise)
- For a deterministic quadratic, a greedy (1-step) choice of learning rate and momentum decay is optimal, since it's equivalent to conjugate gradient (Lecture 9)





Short-Horizon Bias

- Using dynamic programming, we can determine the expected loss under any learning rate and momentum schdule (Lecture 7)
- We can optimize the schedule using dynamic programming



Short-Horizon Bias

- The NQM gives a clear model for why short horizons bias us towards small learning rates
  - This is a tough problem to get around, since large learning rates help by making progress in low-curvature directions, which is invisible if you only measure the loss over the short term
  - Maybe measuring more information would make meta-descent work? But what information?
- I believe this is a fundamental problem not just for meta-descent on learning rate (schedules), but also for any meta-optimizer that can express a learning rate (schedule), e.g.
  - Rescaling a preconditioner is equivalent to changing the learning rate
  - ϵ in RMSprop/Adam, damping parameter in K-FAC
  - Batch norm implicit decay effect

Closing Thoughts

Closing Thoughts

Some (mostly) open questions about bilevel optimization dynamics:

- When is the outer objective smooth or chaotic?
- When should we use a simultaneous vs. a Stackelberg game?
- What are the effects of various ways of approximating $\mathbf{H}^{-1}$?
- We assumed the inner and outer objectives had unique optima. What if one (or both) is overparameterized?
  - What implicit regularization is encoded in bilevel optimization, and how does it depend on the inner and outer optimizers?
- How is it affected by stochastic gradients?
  - Are we in the noise-dominated or curvature-dominated regime, and are these even the right concepts to consider?
- Can we understand and improve the game dynamics for STNs and other approaches?
  - What do $\mathbf{H}$, $\mathbf{G}$, etc. for the inner and outer objectives tell us about the game dynamics? (E.g., how to understand the centering effect in Δ-STNs?)