Are You Ready to Embrace Structural Credit Assignment?

Rich Sutton

University of Alberta
Alberta Machine Intelligence Institute
Reinforcement Learning and Artificial Intelligence Lab

Tea Time Talk 2023

The problem of learning can be usefully divided into two problems of assigning credit

- Temporal Credit Assignment is determining the *times* that should be credited, i.e. changed
    - This is present in the problem and algorithms of reinforcement learning
- Structural Credit Assignment is determining the *structures*, i.e. *parts*, that should be credited, i.e. changed
    - This is present in both RL and supervised learning
    - We see SCA without TCA in supervised learning

In supervised learning and artificial neural networks, the problem of structural credit assignment is that of determining *which weights* to change

- Any given error can generally be eliminated by changing many weights
- Should the change be distributed equally among all weights?
- Or localized in some weights more than others? Which?
- There are choices to be made!

- This is the problem of structural credit assignment in ANNs
- I don’t think we have given it the thought that it deserves

Backprop is a naive form of structural credit assignment

gradient-proportional structural credit assignment

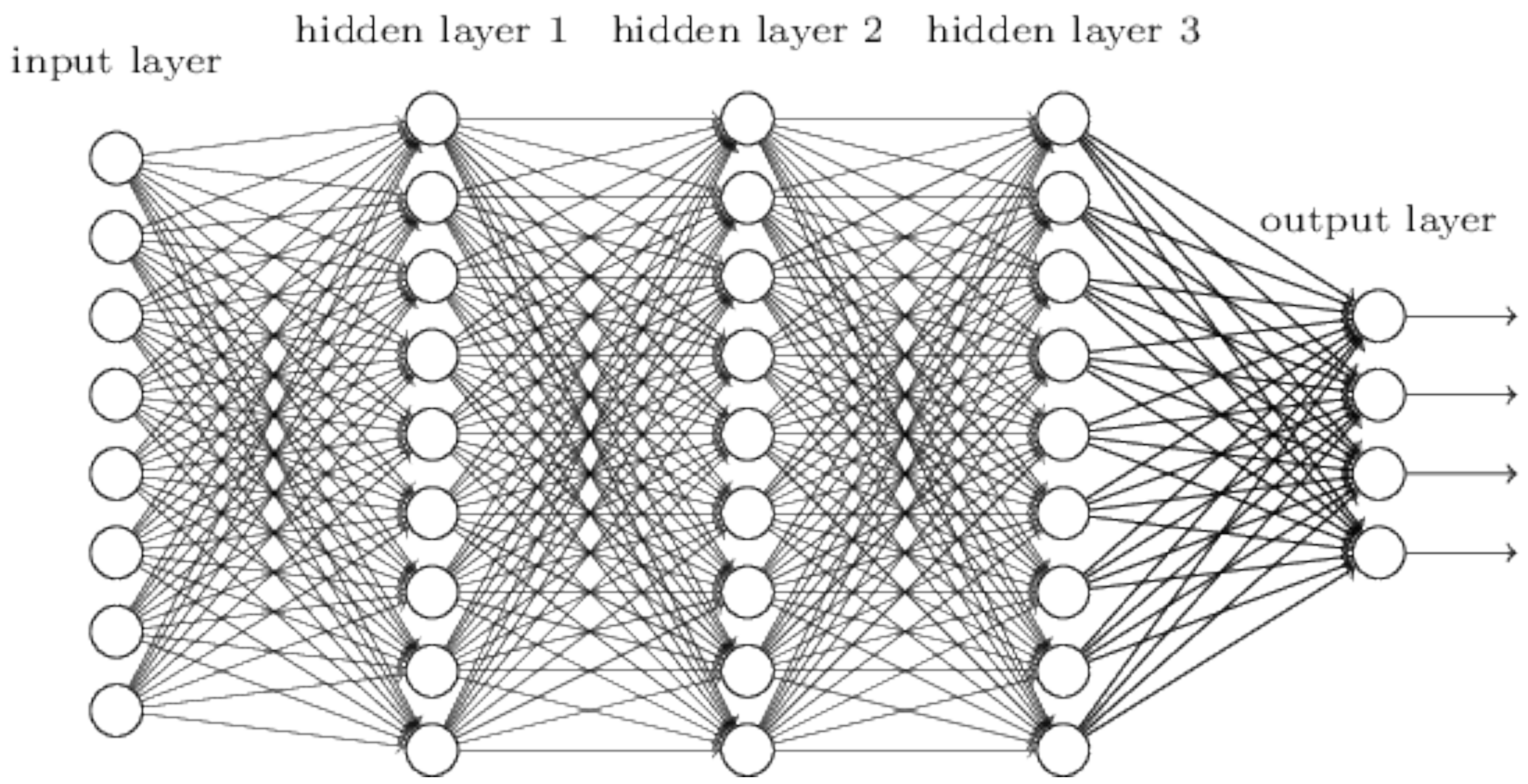
- Backprop is a *steepest descent* method (take the steepest path down the error surface):

$$\Delta w_{i,t} \doteq \alpha_i \frac{\partial \left(y_t - y_t^*\right)^2}{\partial w_{i,t}} \doteq \alpha_i \, g_{i,t} \qquad \text{with all } \alpha_i \text{ equal}$$

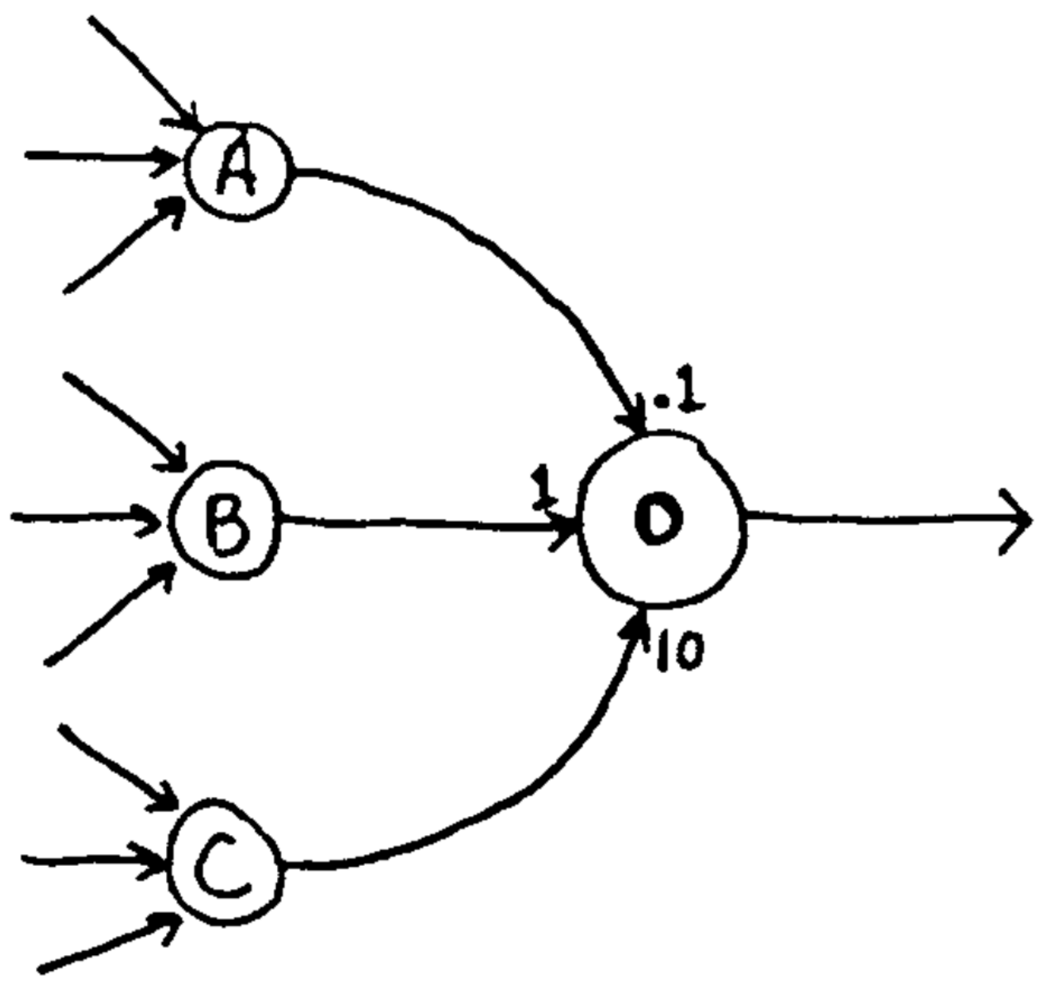
  here we are giving credit to the ith weight in proportion to the ith component of the gradient

- Backprop is a *stochastic* gradient descent method, meaning the above gradient is a <u>sample</u> of the real gradient of interest
  - Because of this, α_i must be roughly <u>constant</u> across examples

In a deep network, the gradients can be very different in different layers



or even in the same layer



rendering steepest descent ineffective (poorly conditioned)

Standard deep learning methods normalize the gradients

Recall: $\Delta w_{i,t} = \alpha_i \, g_{i,t}$ where α_i is constant in backprop

RMSprop instead chooses: $\alpha_i = \dfrac{\eta}{\hat{\sigma}_i}$ (dependence on t is implicit here)

where $\hat{\sigma}_i$ is an estimate of the standard deviation of $g_{i,t}$ across time

Adam is the same, only also adds momentum

The result is that all weights change on average *by about the same amount*

Which is just slightly less naive than steepest descent
It is still an abdication of responsibility for structural credit assignment

All this is a perspective on modern deep networks

- That they are neglecting structural credit assignment
- That they are an undifferentiated mass of learning
- Whereas really some parts of them should be learning rapidly, while others are stable or very slowly learning
- That the networks should be choosing which of their parts are learning
  - this is what it would mean to (meta-)learn representations and to (meta-)learn how to generalize

What would Structural Credit Assignment look like in deep networks?

- For all the complexity of deep learning, almost none of it involves different learning in different parts of the network
- Doing SCA could be as simple as learning different step sizes α_i for different weights i
  - towards the goal of learning better
  - not just toward learning the same amount in all weights
- There is a body of work on this, but mostly not for deep networks

I recently surveyed the history of meta-gradient methods for setting α_is

A History of Meta-gradient: Gradient Methods for Meta-learning

Richard S. Sutton

February 19, 2022

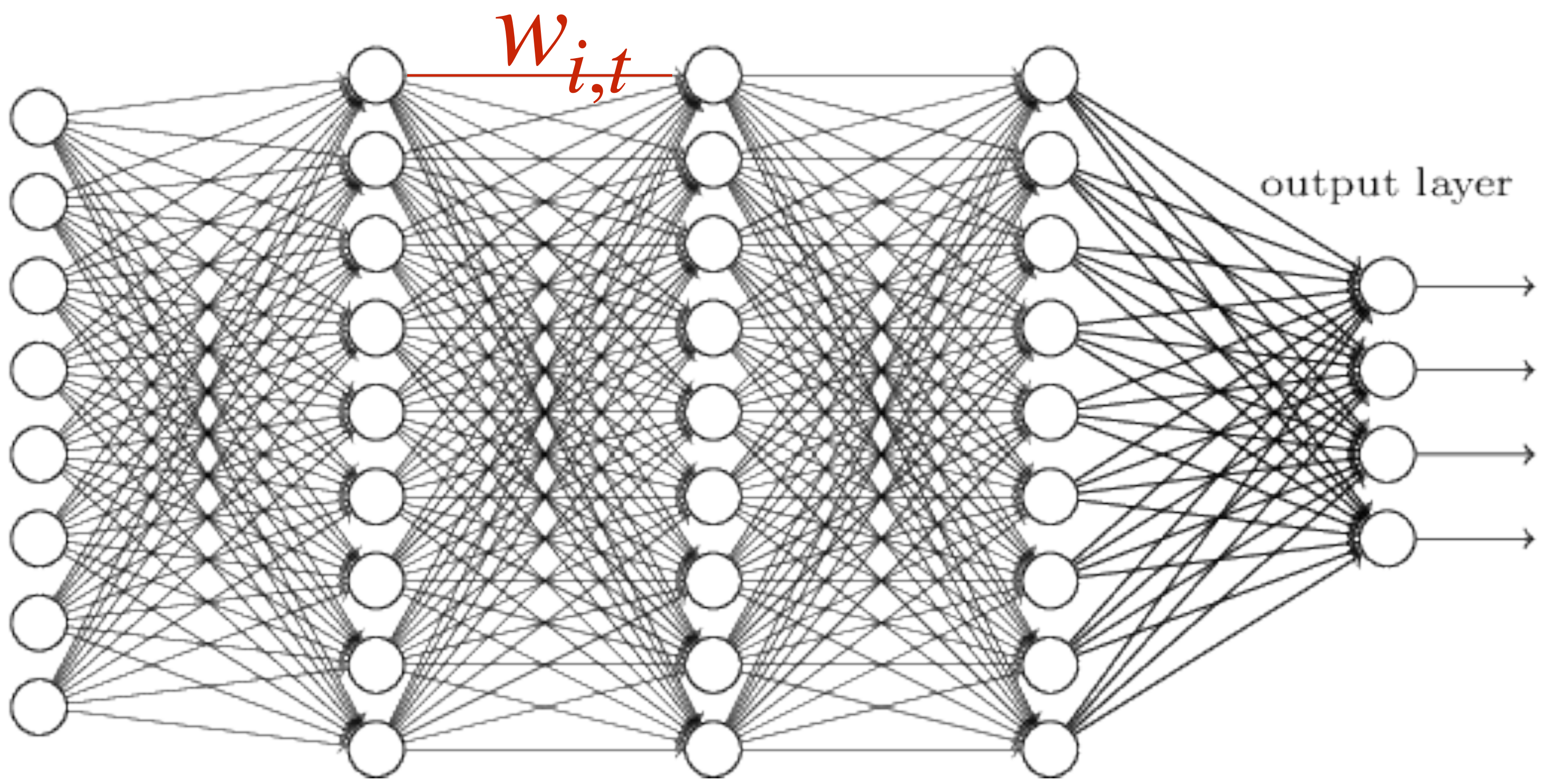
Systems for learning parameters often have meta-parameters such as step sizes, initial weights, or dimensional weightings. With a given setting of the meta-parameters, the learning system is complete and capable of finding parameters that are suited to the task, but its efficiency typically depends on the particular choice of meta-parameters. This has led to interest in learning processes that can find good choices for meta-parameters automatically from experience. These higher-level learning methods are often characterized as "learning to learn" or, as we shall call them here, *meta-learning*. Meta-learning has been explored extensively within machine learning for many years (e.g., see Thrun & Pratt 1998).

ArXiv, 54 references

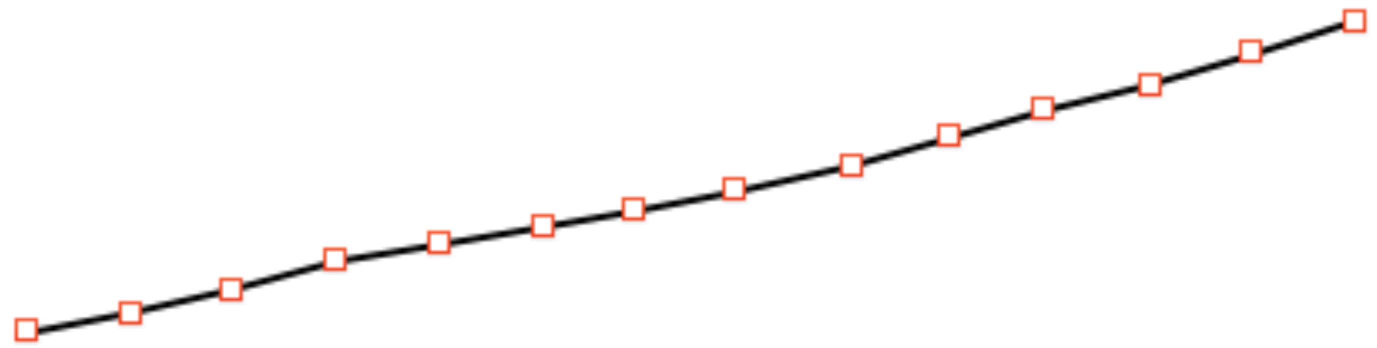
Sutton
Jacobs
Schraudolph
Mahmood
Silver
Koop
Finn
Xu
Kearney
Thill
Young

Suppose a weight is being learned…

$$\Delta w_{i,t} = \alpha_i \, g_{i,t}$$

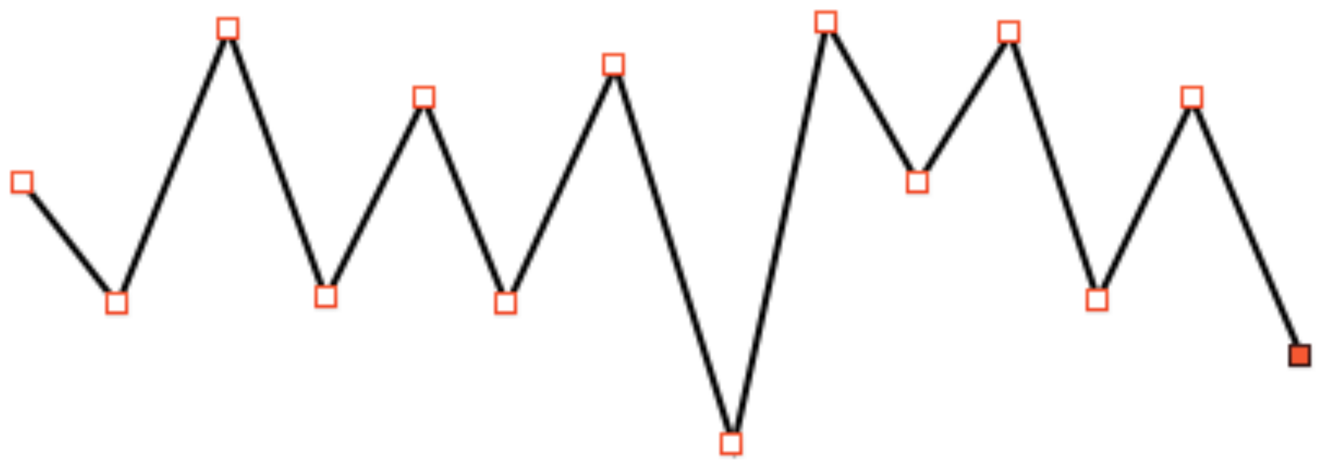


$w_{i,t}$



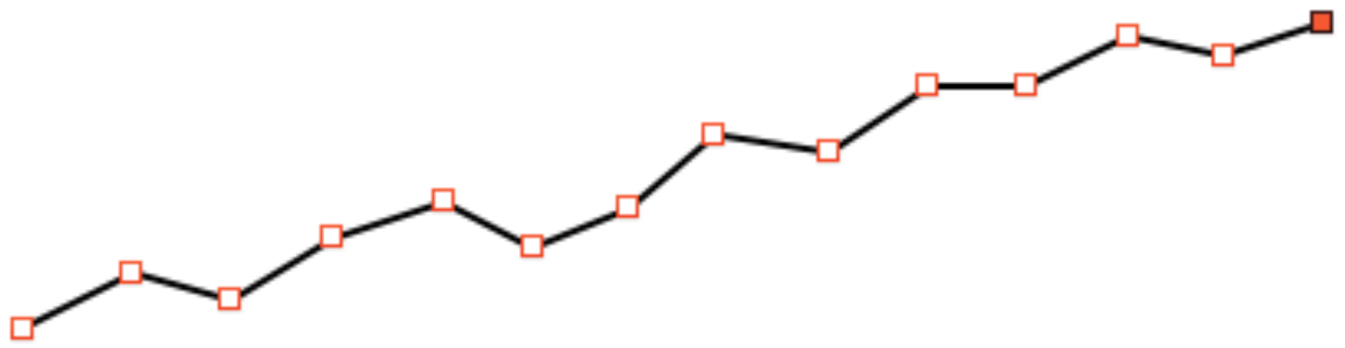
α_i too small

$w_{i,t}$



α_i too big

$w_{i,t}$

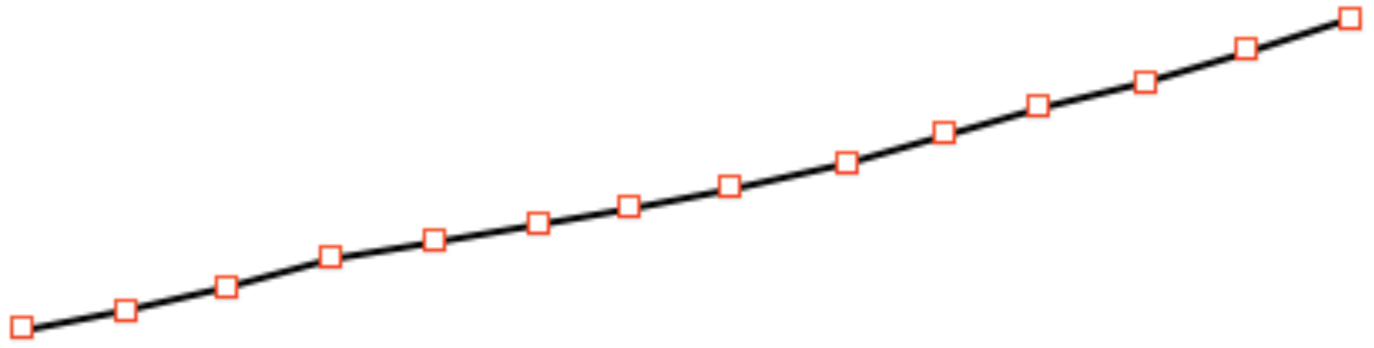


α_i just right?

Define: $\Delta_t \doteq w_{i,t} - w_{i,t-1}$

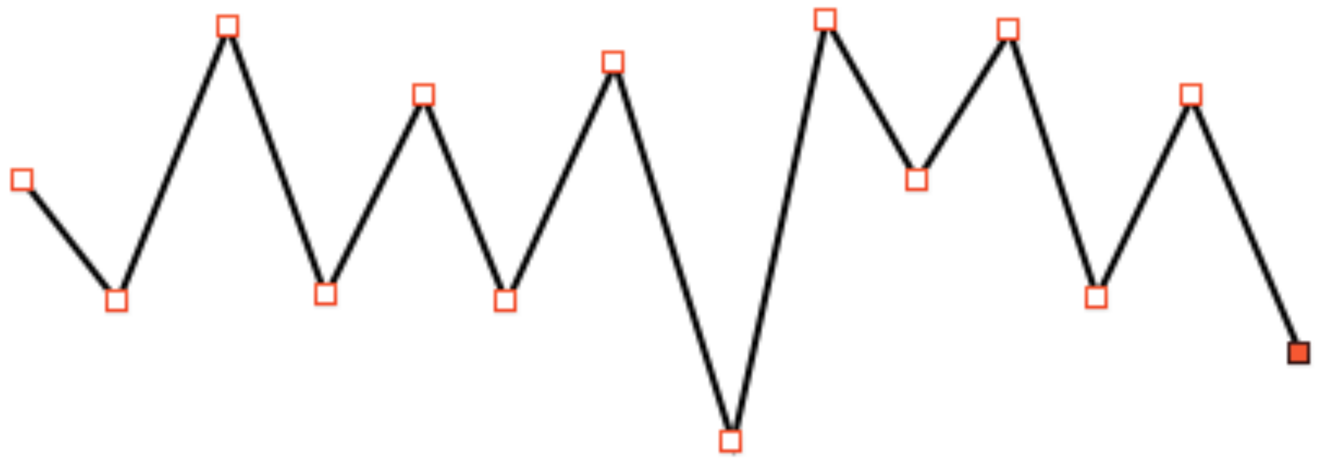
Signature

$w_{i,t}$



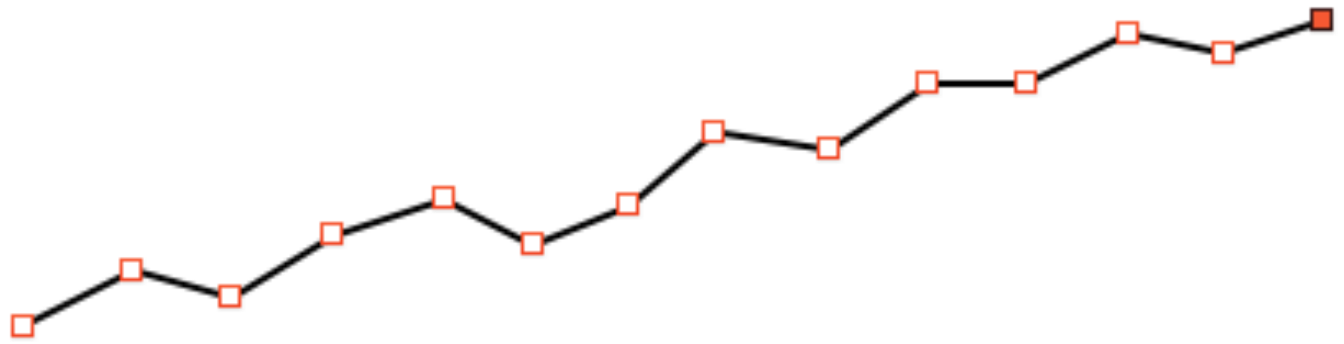
α_i too small — Δ_ts all positive, or same sign

$w_{i,t}$



α_i too big — Δ_ts changing sign

$w_{i,t}$



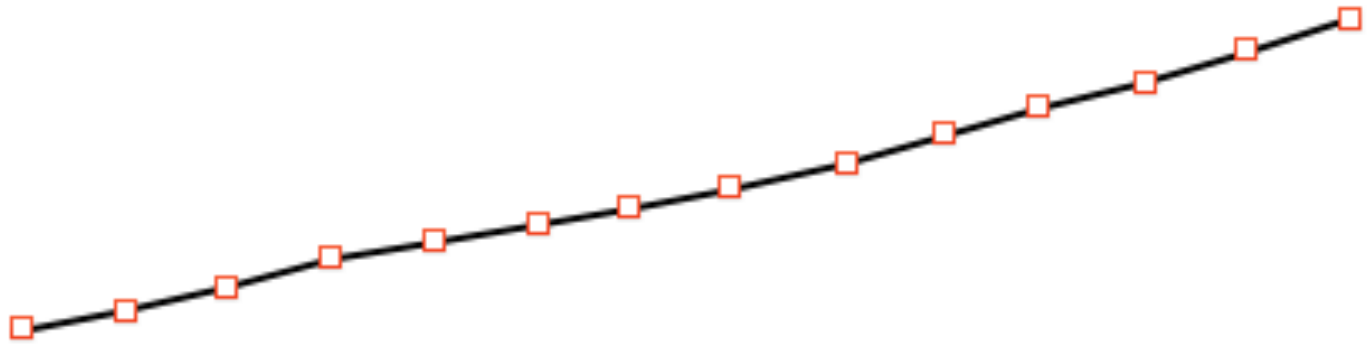
α_i just right? — Δ_ts sometimes same sign, sometimes opposite the $\Delta\Delta$ rule

Suggests a meta learning rule: $\alpha_{i,t} = \alpha_{i,t} + \beta \Delta_{i,t-1} \Delta_{i,t}$

Define: $\Delta_t \doteq w_{i,t} - w_{i,t-1}$

Signature

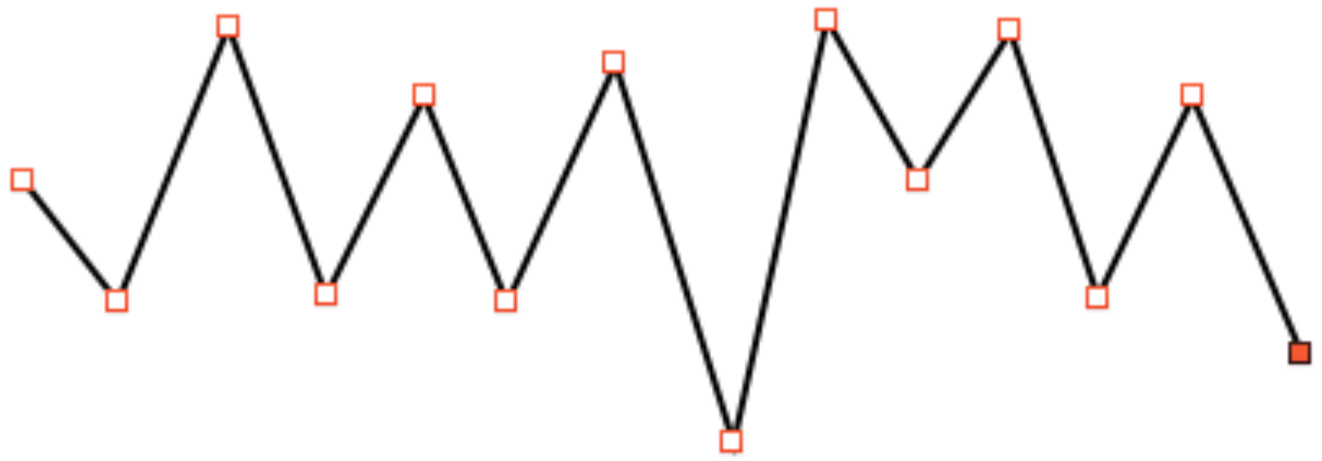
$w_{i,t}$



α_i too small

Δ_ts all positive, or same sign

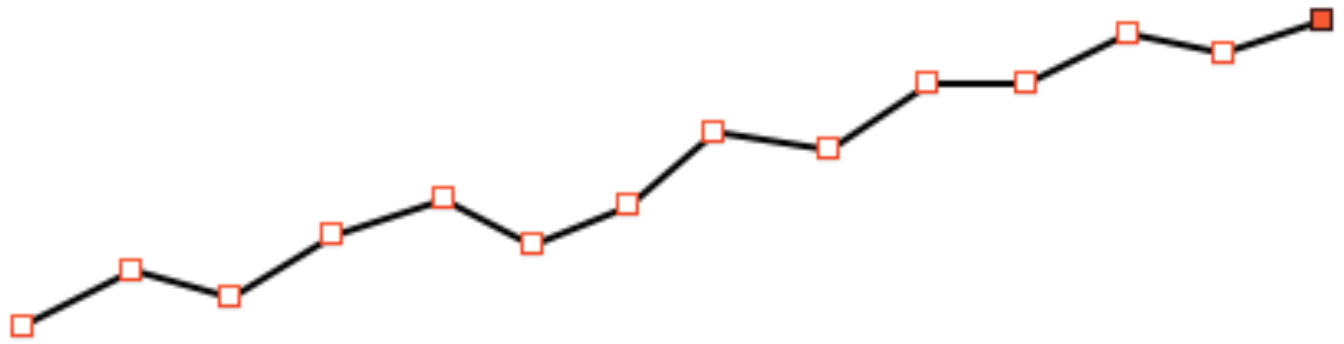
$w_{i,t}$



α_i too big

Δ_ts changing sign

$w_{i,t}$



α_i just right?

Δ_ts sometimes same sign, sometimes opposite

Step-size Optimization for Continual Learning

Thomas Degris[⋆], Khurram Javed[†], Arsalan Sharifnassab[†], Yuxin Liu[†], Rich Sutton[†]

[⋆]DeepMind, London
degris@deepmind.com

[†]Department of Computing Science
University of Alberta
kjaved@ualberta.ca
sharifna@ualberta.ca
yliu17@ualberta.ca
rsutton@ualberta.ca

Submitted to TMLR

2-dim non-stationary linear supervised learning

Target: $y_t^* \doteq w_1^* x_{1,t} + w_2^* x_{2,t}$

where $w_1^* \doteq 0$

and w_2^* may switch between ± 1

every 20 examples

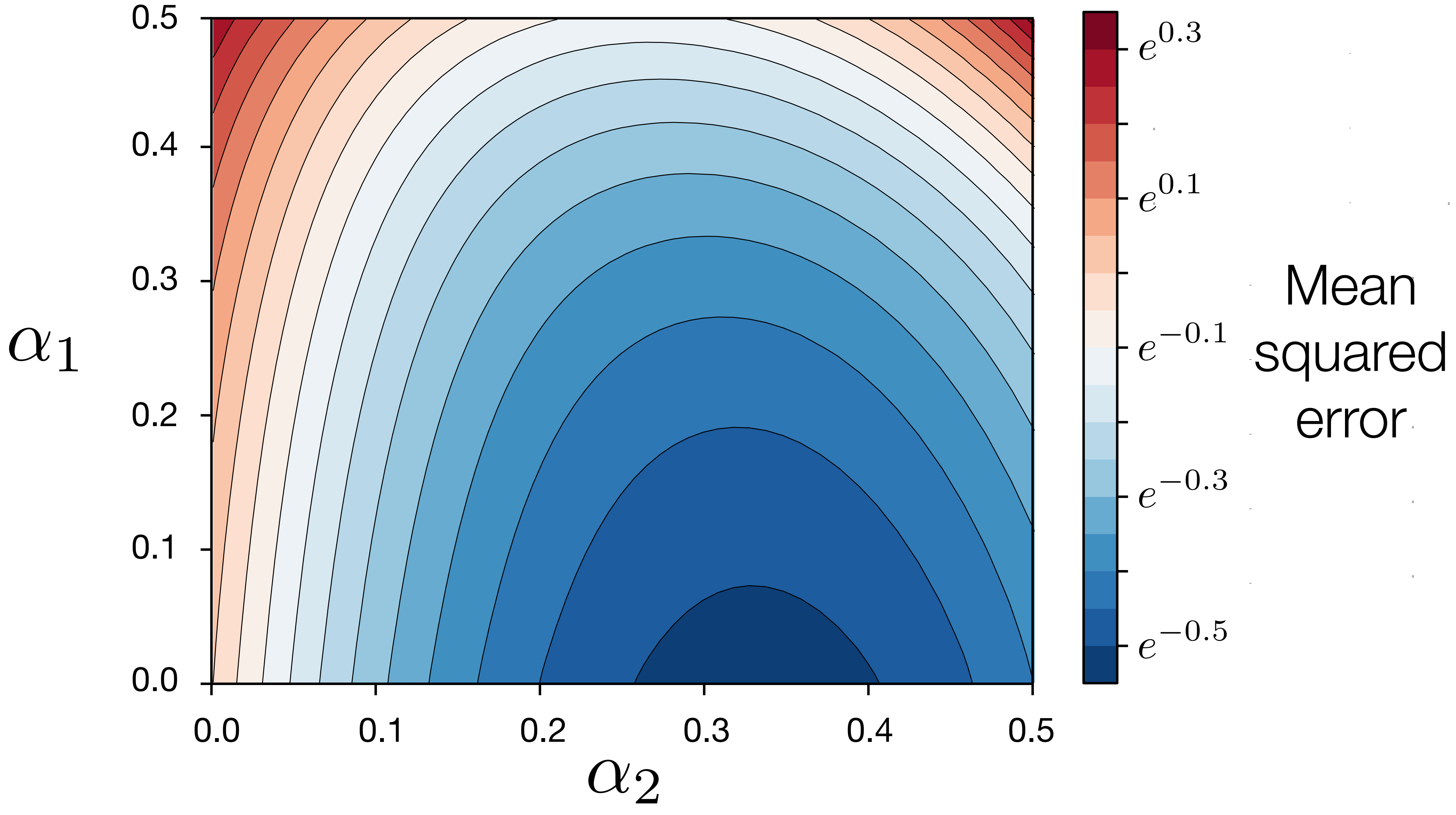
Inputs: $x_{i,t} \sim \mathcal{N}(0,1)$

Learning:

$$y_t \doteq w_{1,t} x_{1,t} + w_{2,t} x_{2,t}$$

$$w_{i,t+1} \doteq w_{i,t} + \alpha_i \left(y_t^* - y_t \right) x_{i,t}$$

The Error Surface
as a function of the *step sizes*



2-dim non-stationary linear supervised learning

Target: $y_t^* \doteq w_1^* x_{1,t} + w_2^* x_{2,t}$

where $w_1^* \doteq 0$

and w_2^* may switch between ± 1

every 20 examples

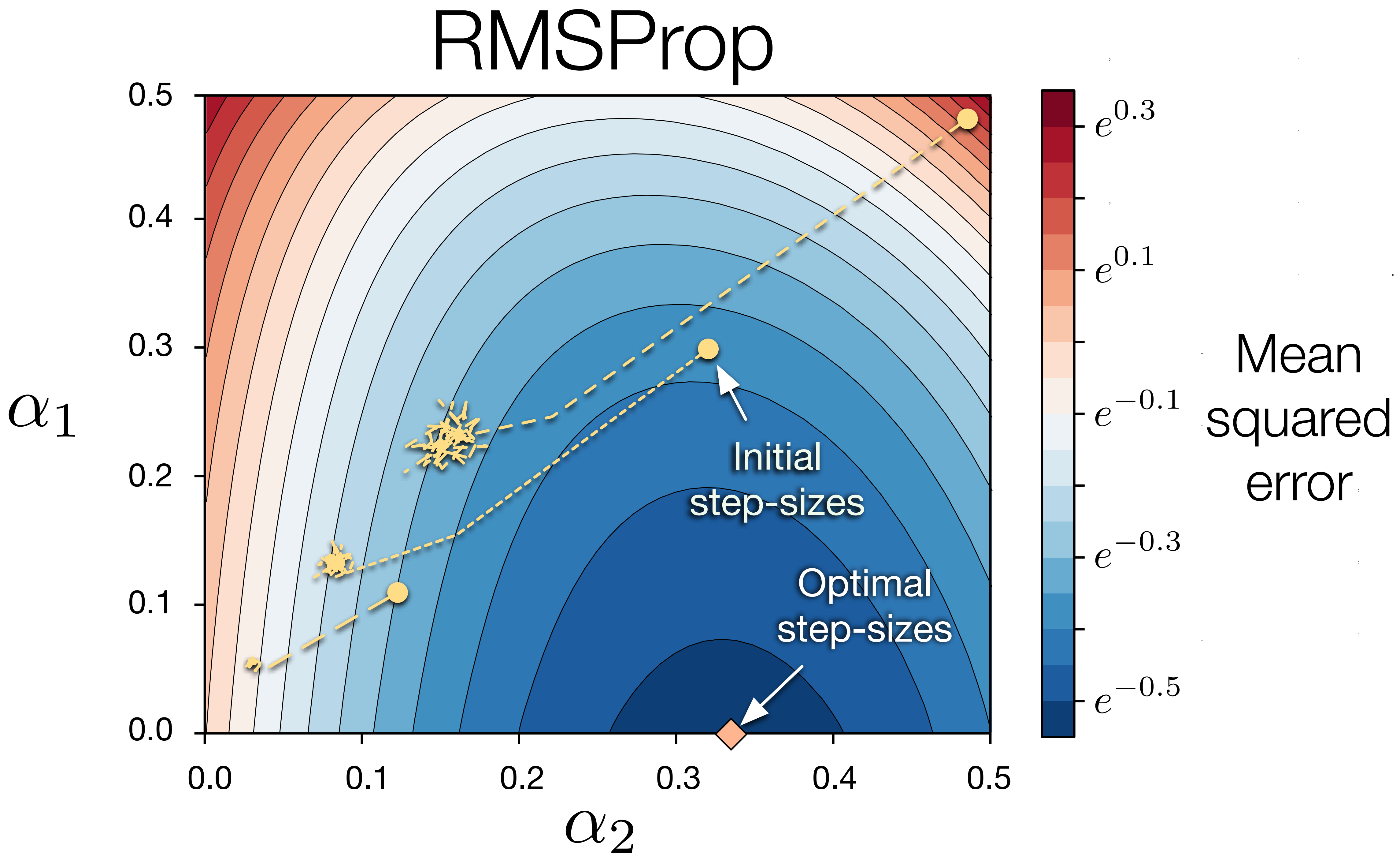
Inputs: $x_{i,t} \sim \mathcal{N}(0,1)$

Learning:

$$y_t \doteq w_{1,t} x_{1,t} + w_{2,t} x_{2,t}$$

$$w_{i,t+1} \doteq w_{i,t} + \alpha_i \left(y_t^* - y_t\right) x_{i,t}$$

The Error Surface
as a function of the *step sizes*



2-dim non-stationary linear supervised learning

Target: $y_t^* \doteq w_1^* x_{1,t} + w_2^* x_{2,t}$

where $w_1^* \doteq 0$

and w_2^* may switch between ± 1

every 20 examples

Inputs: $x_{i,t} \sim \mathcal{N}(0,1)$

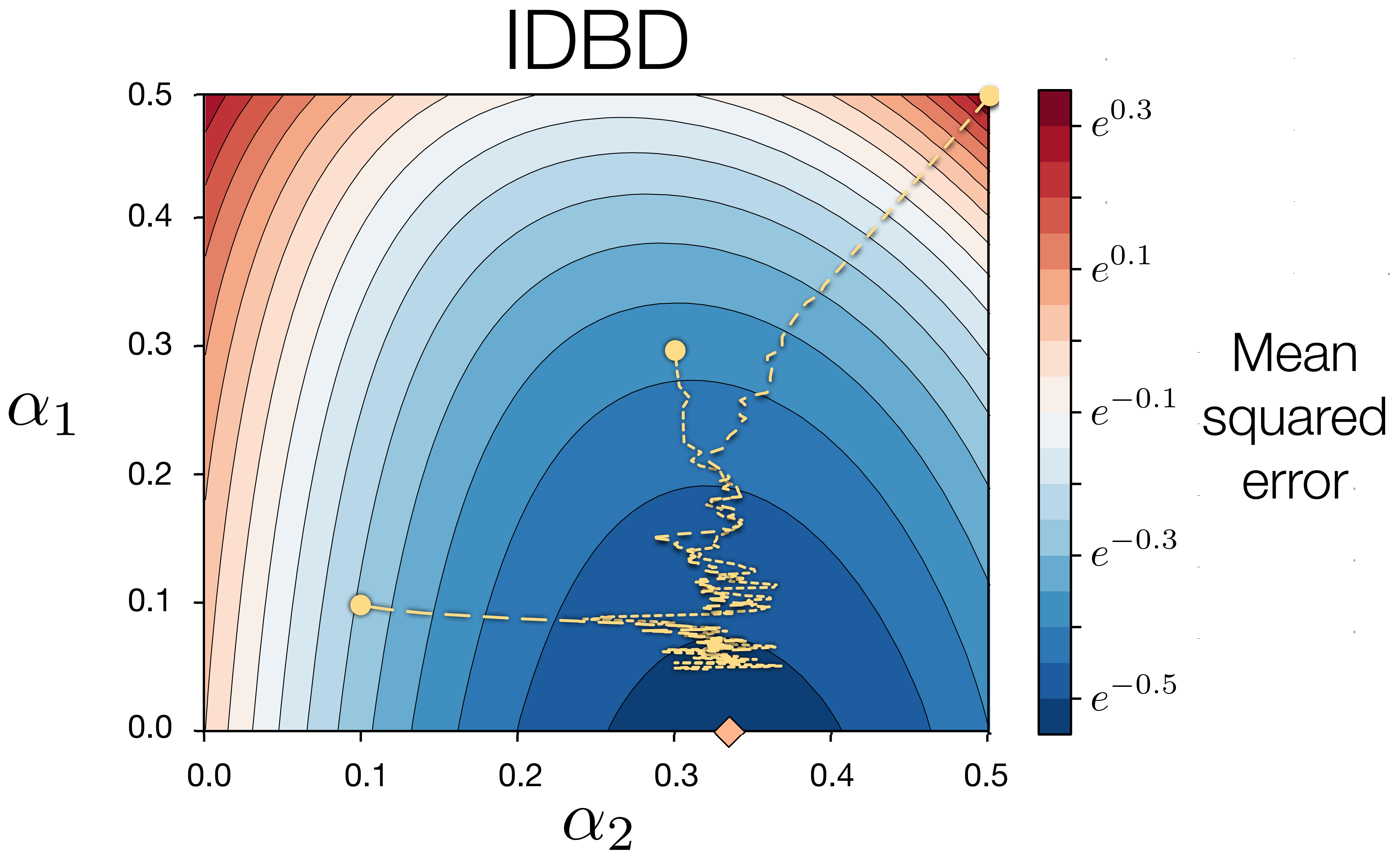
Learning:

$$y_t \doteq w_{1,t} x_{1,t} + w_{2,t} x_{2,t}$$

$$w_{i,t+1} \doteq w_{i,t} + \alpha_i \left(y_t^* - y_t\right) x_{i,t}$$

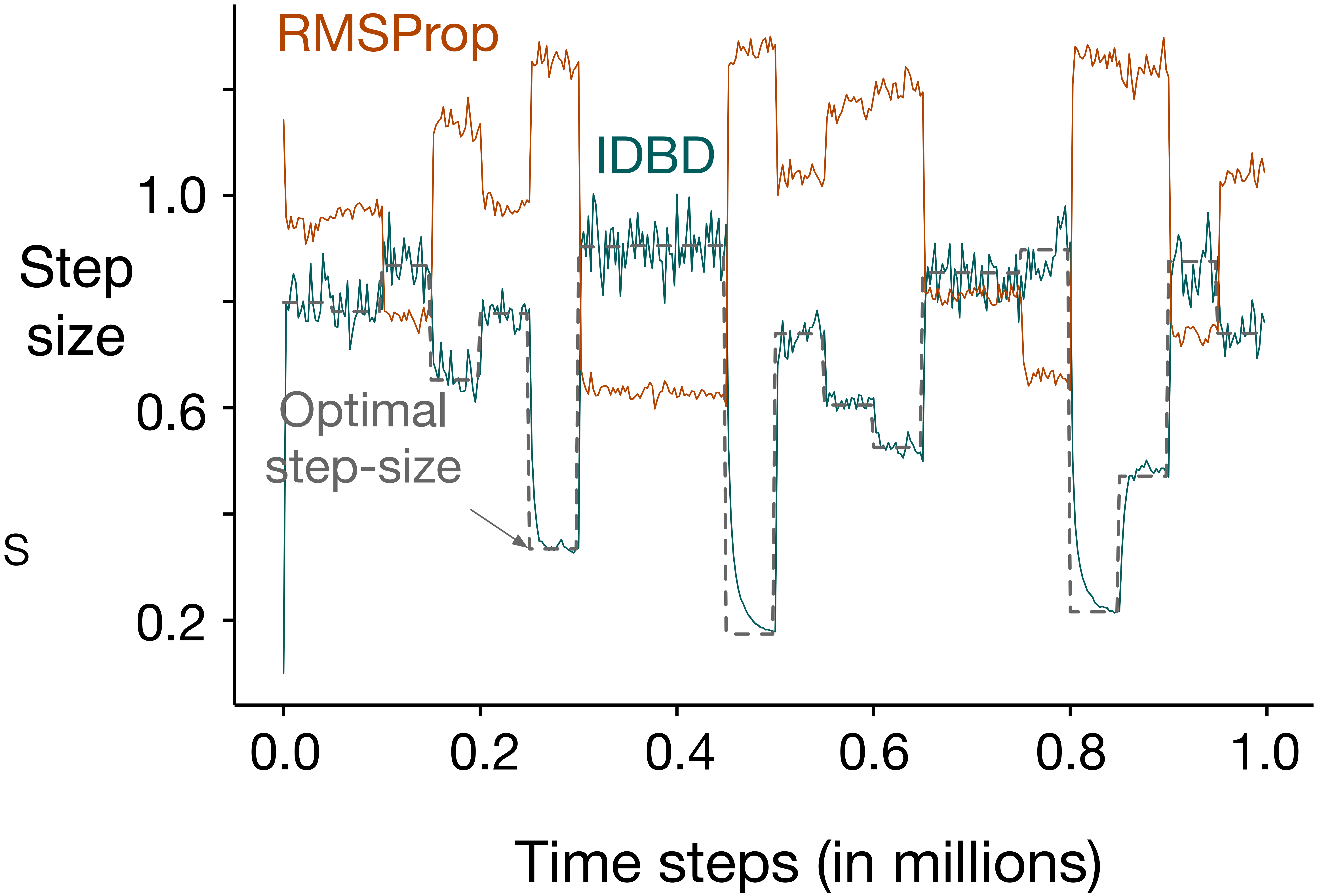
Only IDBD finds optimal step sizes

The Error Surface
as a function of the *step sizes*



1-dim noisy non-stationary tracking problem

- Tracking a number that is taking a random walk
  - incr by $\mathcal{N}(0,1)$ each step
- Number observed with noise $\mathcal{N}(0,\sigma)$
  - $\sigma \in [0,3]$ re-selected randomly every 50,000 steps



Conclusions

- I presented a perspective on modern deep learning
  - that it still not embracing structural credit assignment
  - that it is still not choosing which parts of the network should learn
  - and that until it does that, it will fail at continual learning and representation learning
- There is a line of research—meta-gradient step-size optimization—that does address structural credit assignment
  - well developed for *linear* supervised learning
  - but seemingly ready to be extended to deep networks
- It is high time to embrace the challenge of structural credit assignment

Thank you for your attention