GRADIENT BASED HPO OVER LONG HORIZONS

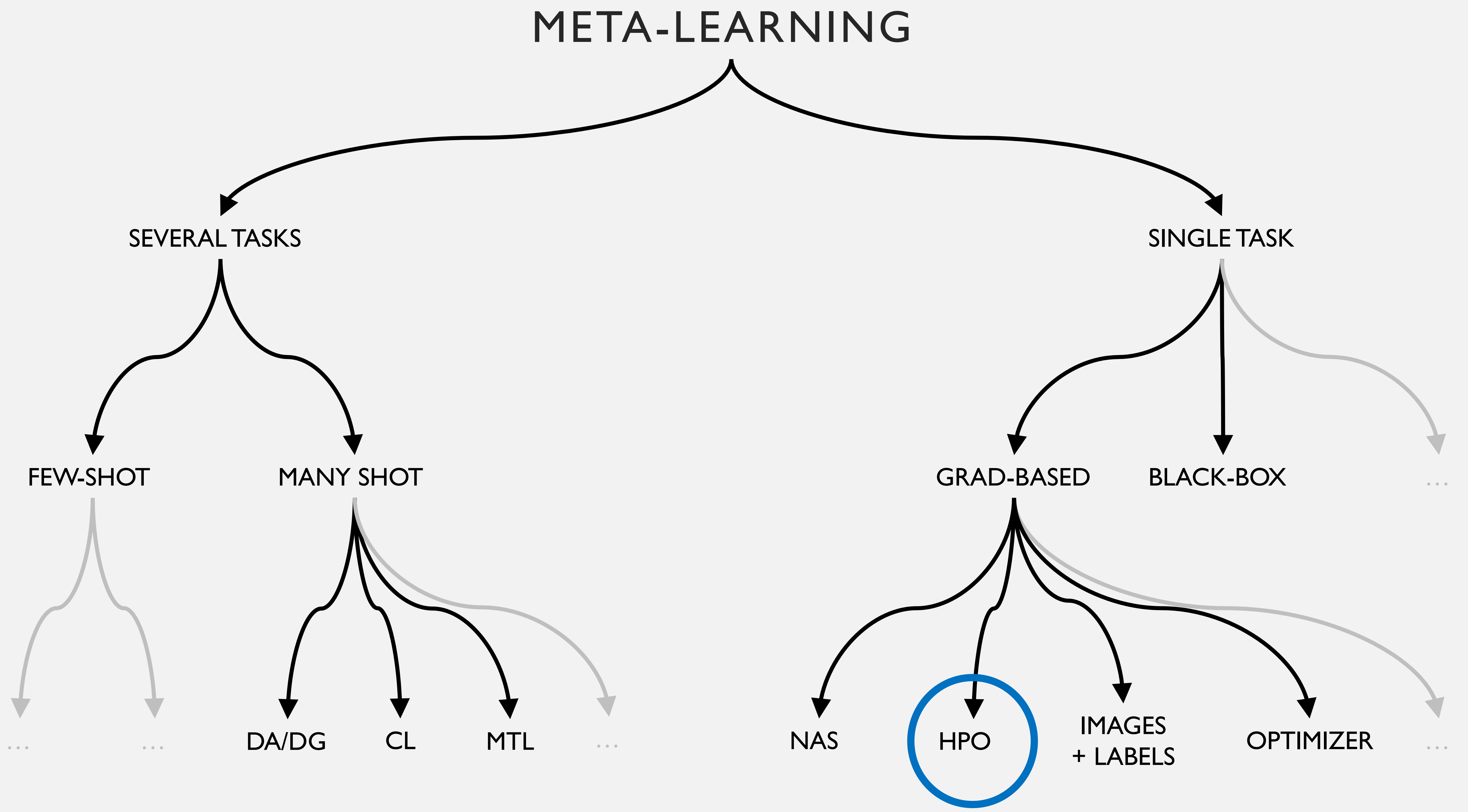
Paul Micaelli and Amos Storkey

OVERVIEW

- Meta-learning & HPO
- Critique of BPTT and greediness for long inner loops
- Forward mode differentiation
- Hyperparameter sharing

META-LEARNING

- Learning to learn
- Reduces human "algorithm engineering"
- *Meta-Learning in Neural Networks: A Survey* Hospedales et al.


META-LEARNING
SEVERAL TASKS
SINGLE TASK
FEW-SHOT
MANY SHOT
GRAD-BASED
BLACK-BOX
...
...
...
DA/DG
CL
MTL
...
NAS
HPO
IMAGES + LABELS
OPTIMIZER
...

GRADIENT BASED HPO

- Useful to formalize as a constrained optimization

$$\begin{aligned}\boldsymbol{\lambda}^* &= \underset{\boldsymbol{\lambda}}{\operatorname{argmin}}\, \mathcal{L}_{\text{val}}(\boldsymbol{\theta}_T(\boldsymbol{\lambda}), \mathcal{D}_{\text{val}}) \\ &\quad \text{subject to} \quad \boldsymbol{\theta}_{t+1} = \Phi\left(\mathcal{L}_{\text{train}}(\boldsymbol{\theta}_t(\boldsymbol{\lambda}), \mathcal{D}_{\text{train}}), \boldsymbol{\lambda}\right)\end{aligned}$$

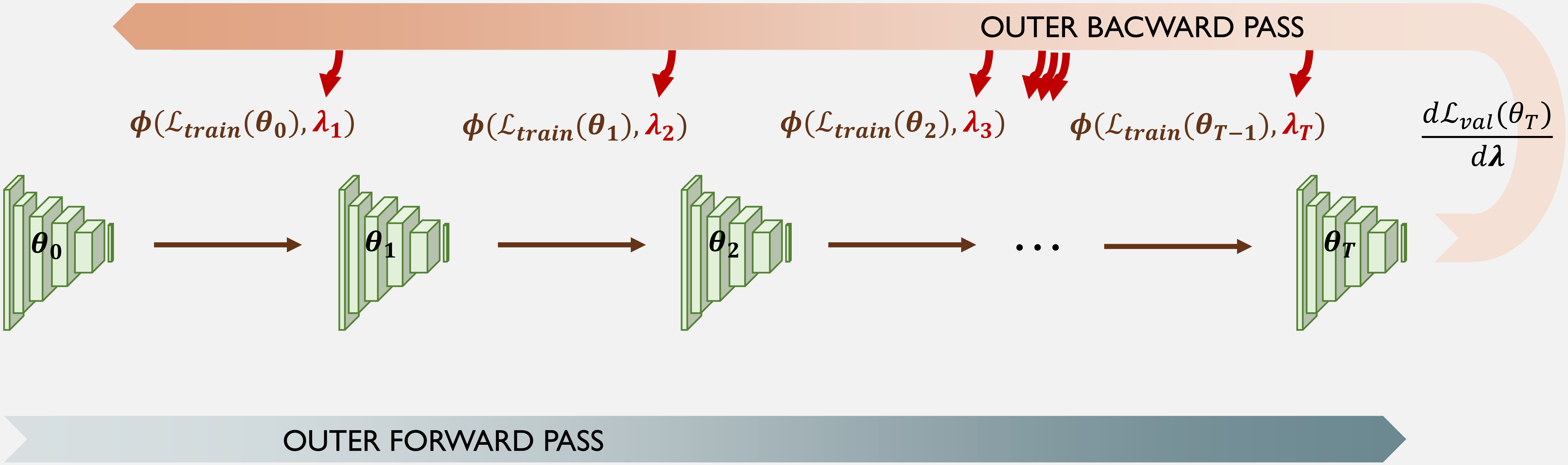
Outer Loop

Inner Loop

Learn **hyperparameters** $\boldsymbol{\lambda}$
… such that the **network weights** $\boldsymbol{\theta_T}$ after T steps of **optimizer** $\boldsymbol{\phi}$ on the train loss
… also minimize the validation loss

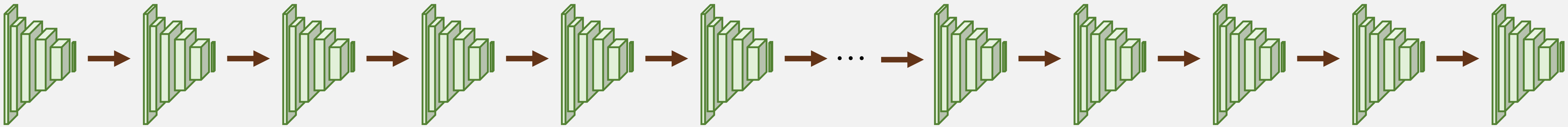
Ultimately what we want is the **hypergradient**: $\dfrac{d\mathcal{L}_{val}(\theta_T)}{d\boldsymbol{\lambda}}$

USING BPTT



OUTER BACWARD PASS
$\phi(\mathcal{L}_{train}(\theta_0), \lambda_1)$
$\phi(\mathcal{L}_{train}(\theta_1), \lambda_2)$
$\phi(\mathcal{L}_{train}(\theta_2), \lambda_3)$
$\phi(\mathcal{L}_{train}(\theta_{T-1}), \lambda_T)$
$\frac{d\mathcal{L}_{val}(\theta_T)}{d\lambda}$
θ_0
θ_1
θ_2
...
θ_T
OUTER FORWARD PASS

CHALLENGES WHEN T IS LARGE: MEMORY



- In BPTT, you need to store each inner step in memory
- Ok for few-shot learning (e.g. MAML where $\mathrm{T} \sim 5$ inner steps)
- But for problems like CIFAR-10, we need $\mathrm{T} \sim 10^4$ inner steps

CHALLENGES WHEN T IS LARGE: GRADIENT DEGRADATION

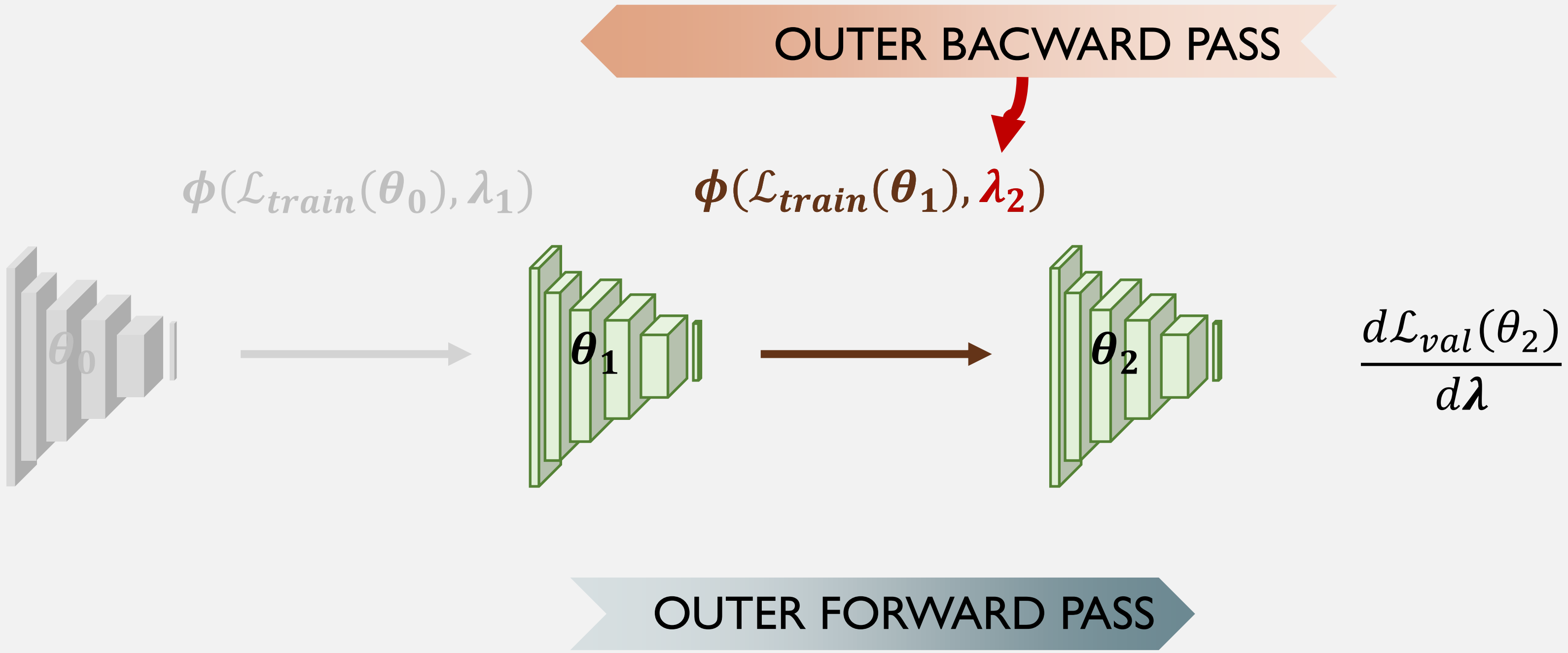
- Broad issue that arises when a parameter influences some scalar in a chaotic fashion (such as a long composition of non-linear ops)
- Vanishing + exploding hypergradients = high variance

SOLUTION = GREEDINESS..?

- We can take H steps before updating hyperparameters, with $H \ll T$
- H is the *horizon*

SOLUTION = GREEDINESS..?

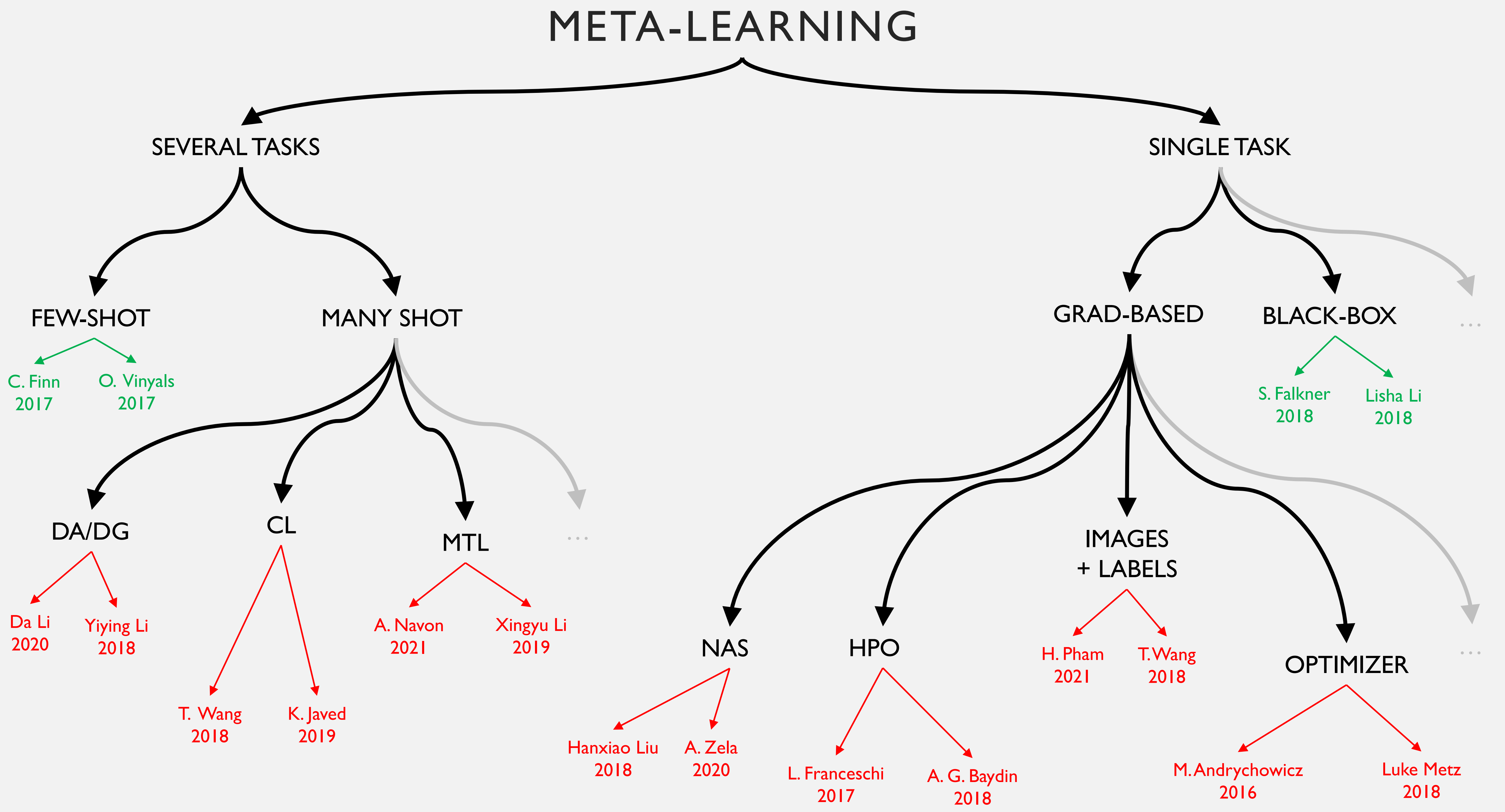
- For instance if $H = 1$:



SOLUTION = GREEDINESS..?

- Solves memory issue of BPTT
- Solves gradient degradation
- Improves computational cost
- Solves for the wrong objective

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


META-LEARNING
SEVERAL TASKS
SINGLE TASK
FEW-SHOT
MANY SHOT
GRAD-BASED
BLACK-BOX
...
C. Finn 2017
O. Vinyals 2017
S. Falkner 2018
Lisha Li 2018
DA/DG
CL
MTL
...
IMAGES + LABELS
NAS
HPO
OPTIMIZER
...
Da Li 2020
Yiying Li 2018
T. Wang 2018
K. Javed 2019
A. Navon 2021
Xingyu Li 2019
H. Pham 2021
T. Wang 2018
Hanxiao Liu 2018
A. Zela 2020
L. Franceschi 2017
A. G. Baydin 2018
M. Andrychowicz 2016
Luke Metz 2018

SOLUTION = FORWARD MODE DIFFERENTIATION…?

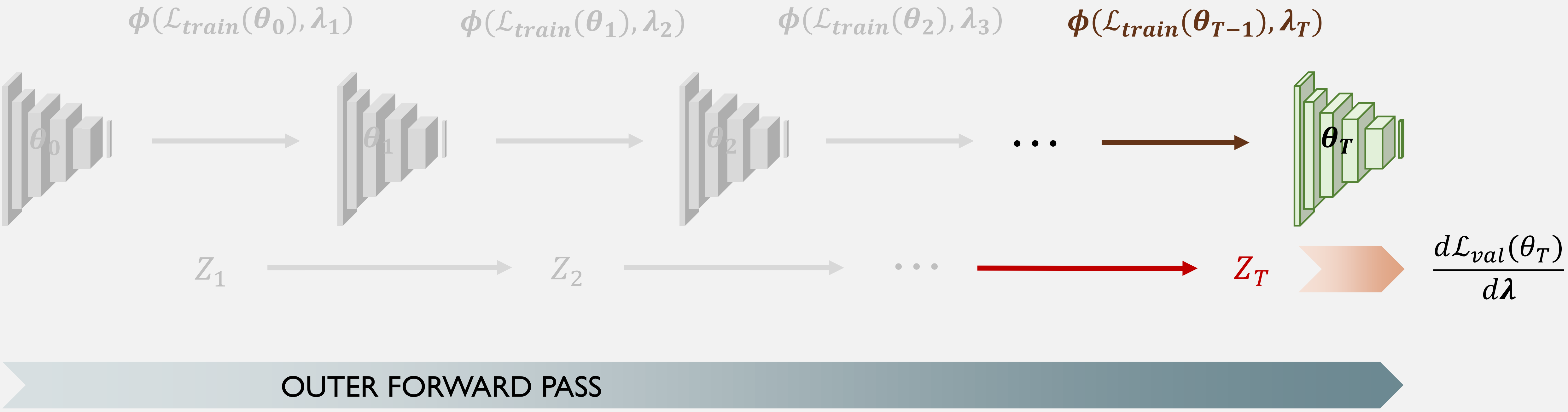
- Calculate gradient components during forward pass
- Applying it to HPO:

$$\frac{d\mathcal{L}_{\text{val}}}{d\boldsymbol{\lambda}} = \overbrace{\frac{\partial \mathcal{L}_{\text{val}}}{\partial \boldsymbol{\theta}_T}}^{\text{reverse mode}} \underbrace{\frac{d\boldsymbol{\theta}_T}{d\boldsymbol{\lambda}}}_{\text{forward mode}}$$

$$\frac{d\boldsymbol{\theta}_t}{d\boldsymbol{\lambda}} = \left.\frac{\partial \boldsymbol{\theta}_t}{\partial \boldsymbol{\theta}_{t-1}}\right|_{\boldsymbol{\lambda}} \frac{d\boldsymbol{\theta}_{t-1}}{d\boldsymbol{\lambda}} + \left.\frac{\partial \boldsymbol{\theta}_t}{\partial \boldsymbol{\lambda}}\right|_{\boldsymbol{\theta}_{t-1}}$$

$$\boldsymbol{Z}_t = \boldsymbol{A}_t \boldsymbol{Z}_{t-1} + \boldsymbol{B}_t$$

FORWARD MODE DIFFERENTIATION

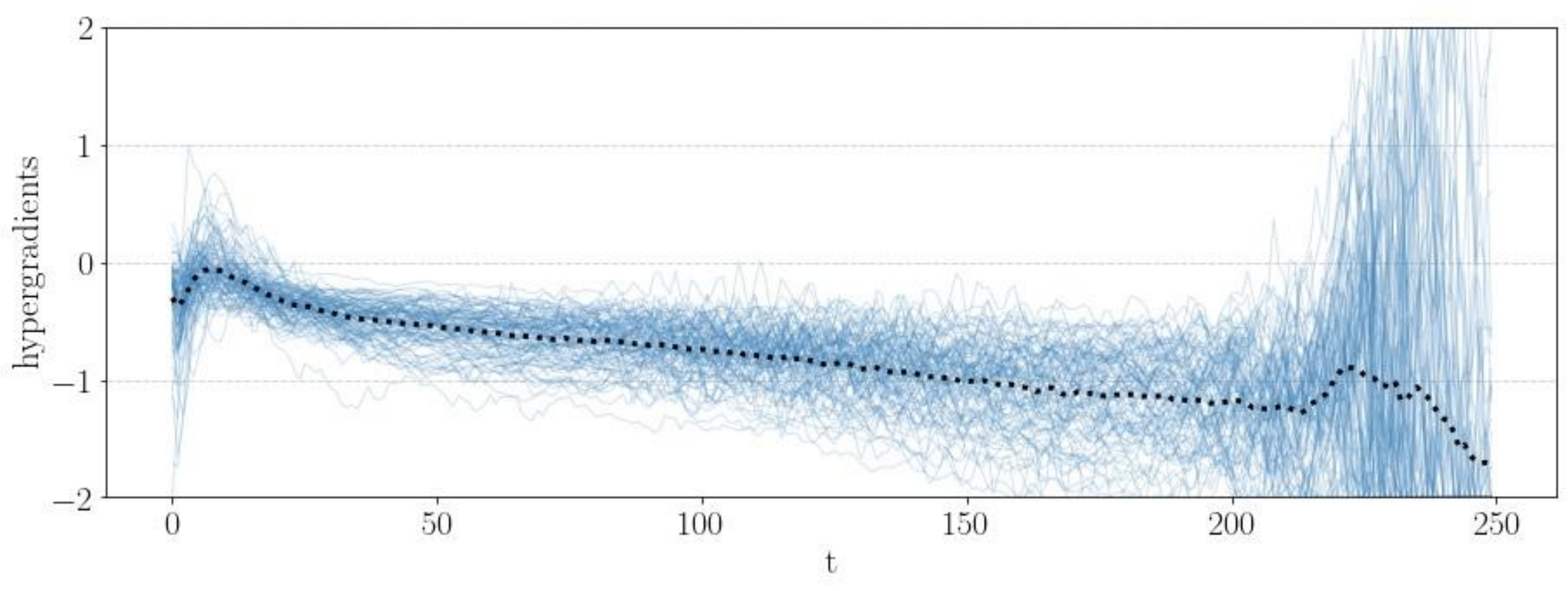


FORWARD MODE DIFFERENTIATION

- Memory cost constant with horizon size
- Solves for correct objective
- Memory cost scales poorly with size of λ
- Doesn't solve gradient degradation

SOLVING GRADIENT DEGRADATION WITH ENSEMBLE AVERAGING

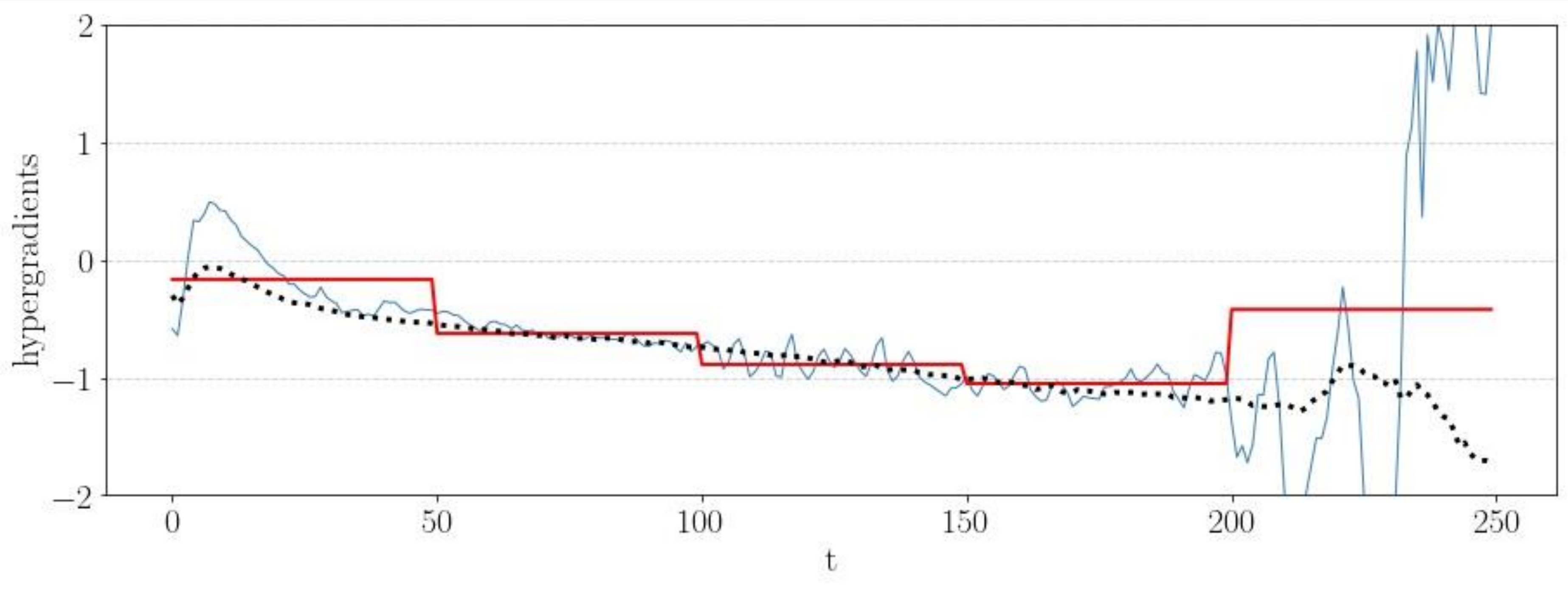
- Each seed gives different hypergradients



- Ensemble average = too expensive in compute / memory

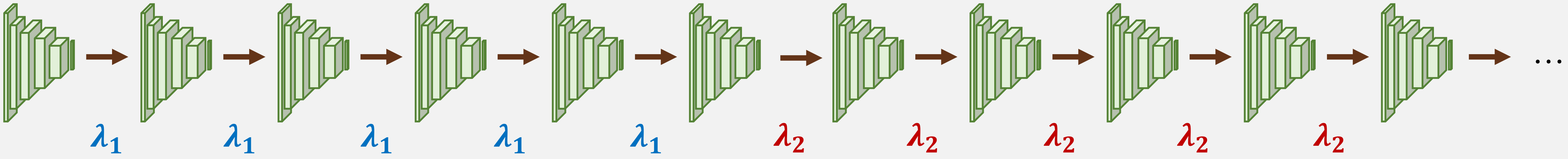
SOLVING GRADIENT DEGRADATION WITH TIME AVERAGING

- Time averaging averages hypergradients in window of size W



- Cheap, and easily achieved by sharing hyperparameters

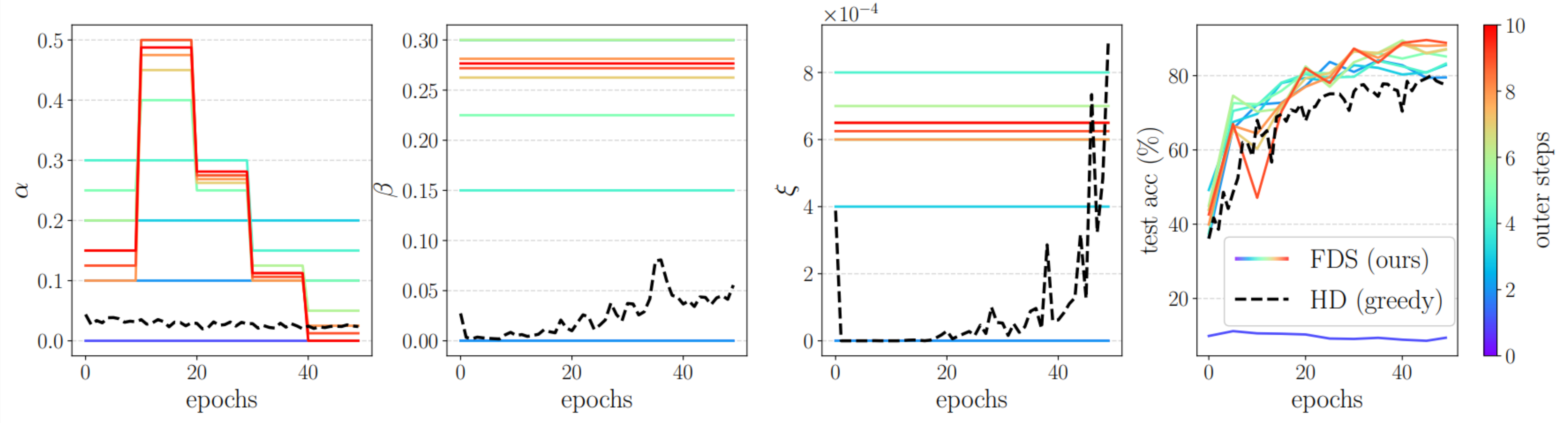
SOLVING GRADIENT DEGRADATION WITH TIME AVERAGING



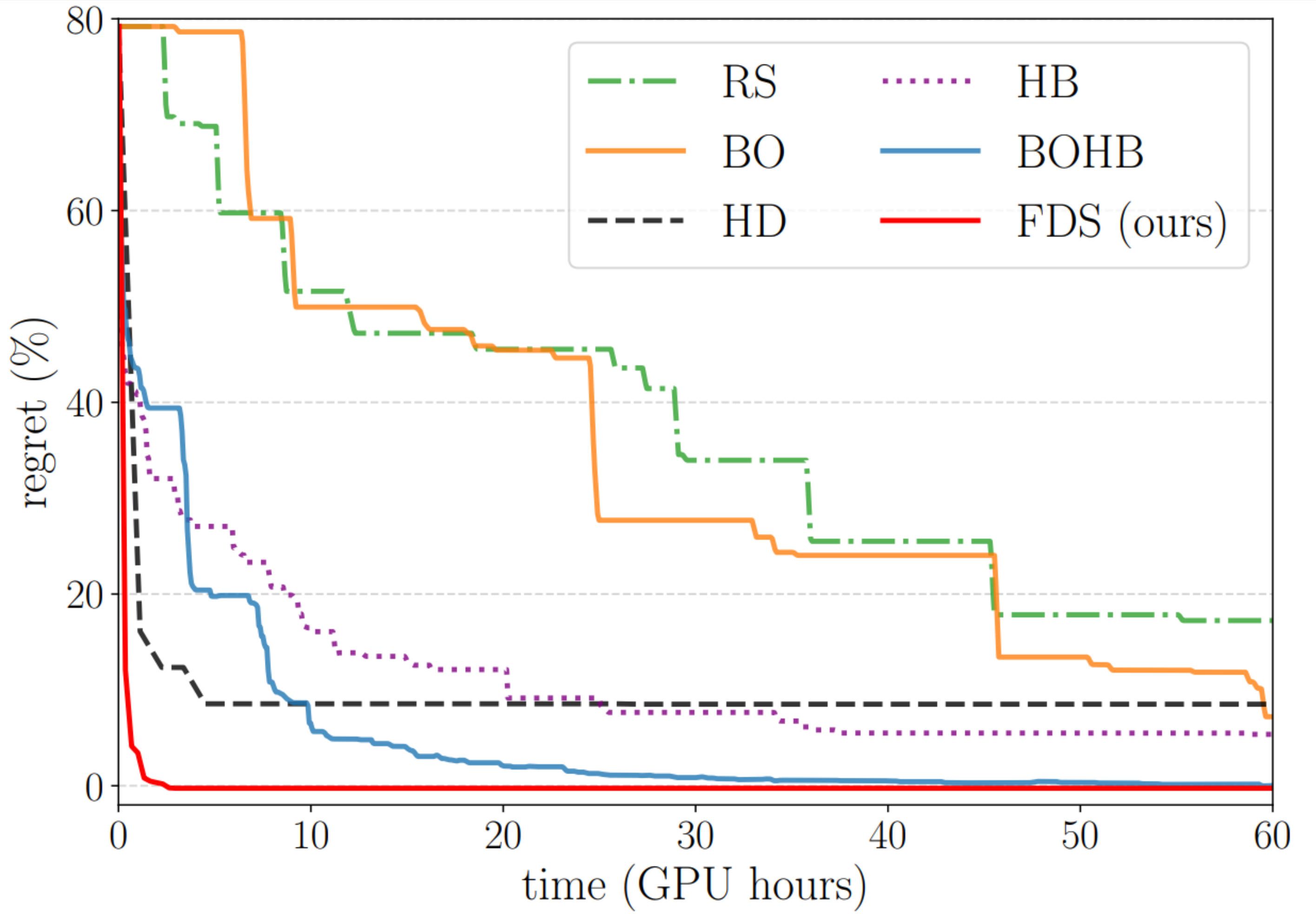
- What window size should we use?
- Optimal W = tradeoff between **variance reduction** and **bias increase**

$$\mathrm{MSE}_W \leq \frac{\mathrm{MSE}_1}{W} + \frac{\epsilon^2(W^2-1)}{12}$$

RESULTS: LEARNING SCHEDULES ON CIFAR10



RESULTS: REGRET



CONCLUSION

- You can differentiation through $\sim 10^4$ gradient steps by sharing contiguous hyperparameters in forward mode differentiation
- Future work:
  - Use automatic forward mode differentiation
  - Tackle harder problems/datasets

Contact: paul.micaelli@ed.ac.uk