Low-Variance Gradient Estimation in Unrolled Computation Graphs with ES-Single

Paul Vicol, Zico Kolter, Kevin Swersky

Motivation

RNN Training	**Unrolled Computation Graph**	**Hyperparameter Optimization**
Cross-entropy loss per token	$L_1 \quad L_2 \quad L_3$	Training or validation loss
RNN hidden state	$\mathbf{s}_1 \rightarrow \mathbf{s}_2 \rightarrow \mathbf{s}_3$	Inner model parameters + optimizer state (e.g., momentum buffers)
RNN parameters	$\boldsymbol{\theta} \quad \boldsymbol{\theta} \quad \boldsymbol{\theta}$	Hyperparameters

$$\begin{tikzcd}
L_1 & L_2 & L_3 \\
\mathbf{s}_1 \arrow[u] \arrow[r] & \mathbf{s}_2 \arrow[u] \arrow[r] & \mathbf{s}_3 \arrow[u] \\
\boldsymbol{\theta} \arrow[u] & \boldsymbol{\theta} \arrow[u] & \boldsymbol{\theta} \arrow[u]
\end{tikzcd}$$

- For all these tasks, the objective is: $L(\boldsymbol{\theta}) = \sum_{t=1}^{T} L_t(\mathbf{s}_t, \boldsymbol{\theta})$

$\Rightarrow$ We need the gradient $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$

Existing Approaches

- Challenges with both short and long unrolls of the inner problem
  - **Short unrolls** $\rightarrow$ *truncation bias*
  - **Long unrolls** $\rightarrow$ *chaotic outer loss landscapes*

- *Evolution Strategies (ES)* computes an estimate of the gradient of a *smoothed loss*

$$g^{ES} = \frac{1}{\sigma^2}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,\sigma^2 I)}[\epsilon L(\boldsymbol{\theta} + \epsilon)]$$

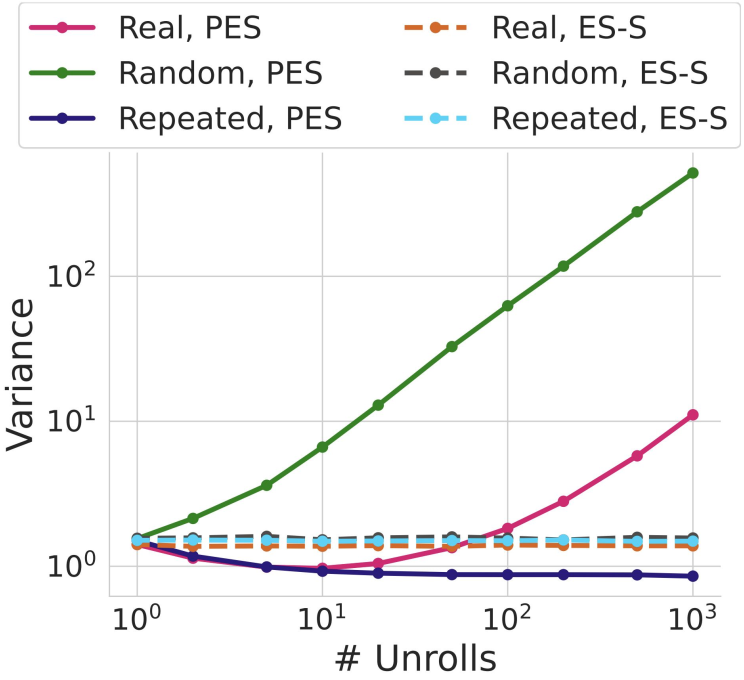
  - Smoothing overcomes chaos
  - But applying ES to full unrolls of the inner problem is expensive — slow updates
  - Naively applying ES to truncated unrolls leads to bias

- *Persistent Evolution Strategies (PES)* computes unbiased gradient estimates using truncated unrolls

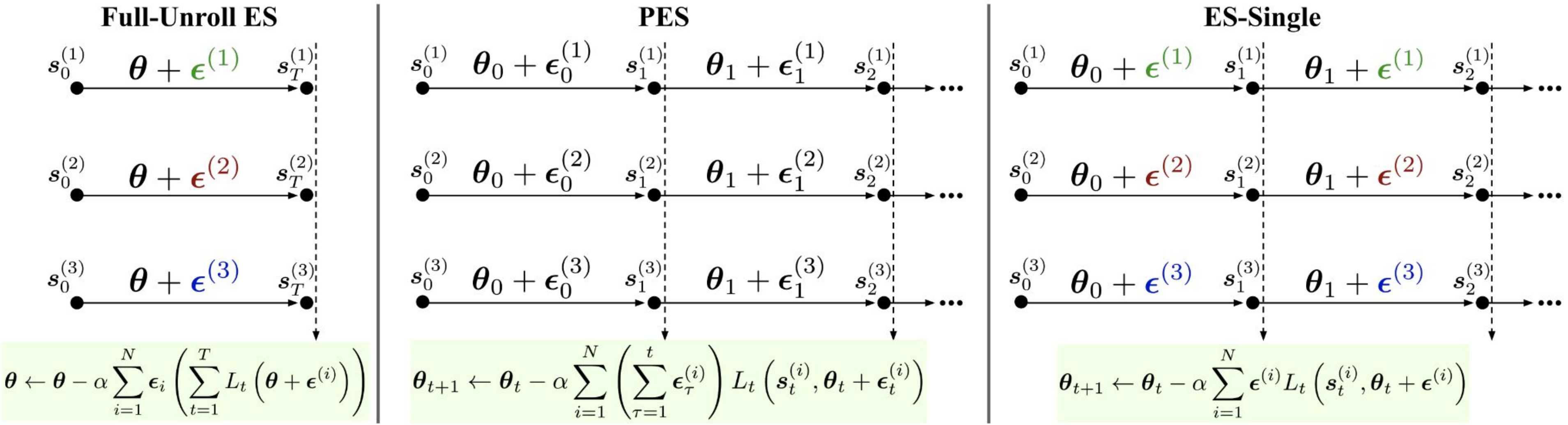
Variance Comparison

- PES is unbiased, but its *variance increases with the number of partial unrolls per inner problem*

- *We introduce ES-Single:* an algorithm for unbiased gradient estimation using partial unrolls
- Simpler and easier to implement than PES
- Has *constant variance* w.r.t. the number of partial unrolls per inner problem



ES-Single Computation Graph



Full-Unroll ES samples a perturbation for each particle, and *runs a full unroll for T steps using perturbed outer parameters*

PES samples a new perturbation for each particle in each unroll, and sums the perturbations experienced by each particle up to the current point in the inner problem

ES-Single samples a single perturbation per particle at the start of each inner problem—keeping it fixed for the duration of the problem—and does not sum perturbations over time.

ES-Single Algorithm

Algorithm 1 Truncated Evolution Strategies (ES) applied to partial unrolls of a computation graph.

Input: $\boldsymbol{s}_0$, initial state
K, truncation length for partial unrolls
N, number of particles
σ, standard deviation of perturbations
α, learning rate for outer optimization

Initialize $\boldsymbol{s} = \boldsymbol{s}_0$
while inner problem not finished **do**
  $\hat{\boldsymbol{g}}^{\text{ES}} \leftarrow \mathbf{0}$
  for $i = 1, \ldots, N$ **do**
    $\boldsymbol{\epsilon}^{(i)} = \begin{cases} \text{draw from } \mathcal{N}(0, \sigma^2 I) & i \text{ odd} \\ -\boldsymbol{\epsilon}^{(i-1)} & i \text{ even} \end{cases}$
    $\hat{L}_K^{(i)} \leftarrow \text{unroll}(\boldsymbol{s}, \boldsymbol{\theta} + \boldsymbol{\epsilon}^{(i)}, K)$
    $\hat{\boldsymbol{g}}^{\text{ES}} \leftarrow \hat{\boldsymbol{g}}^{\text{ES}} + \boldsymbol{\epsilon}^{(i)} \hat{L}_K^{(i)}$
  end for
  $\hat{\boldsymbol{g}}^{\text{ES}} \leftarrow \frac{1}{N\sigma^2} \hat{\boldsymbol{g}}^{\text{ES}}$
  $\boldsymbol{s} \leftarrow \text{unroll}(\boldsymbol{s}, \boldsymbol{\theta}, K)$
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \hat{\boldsymbol{g}}^{\text{ES}}$
end while

Algorithm 2 ES with a single perturbation per particle re-applied in each truncated unroll (ES-Single).

Input: $\boldsymbol{s}_0$, initial state
K, truncation length for partial unrolls
N, number of particles
σ, standard deviation of perturbations
α, learning rate for outer optimization

Initialize $\boldsymbol{s}^{(i)} = \boldsymbol{s}_0$ for $i \in \{1, \ldots, N\}$
for $i = 1, \ldots, N$ **do**
  $\boldsymbol{\epsilon}^{(i)} = \begin{cases} \text{draw from } \mathcal{N}(0, \sigma^2 I) & i \text{ odd} \\ -\boldsymbol{\epsilon}^{(i-1)} & i \text{ even} \end{cases}$
end for
while inner problem not finished **do**
  $\hat{\boldsymbol{g}}^{\text{ES-Single}} \leftarrow \mathbf{0}$
  for $i = 1, \ldots, N$ **do**
    $\boldsymbol{s}^{(i)}, \hat{L}_K^{(i)} \leftarrow \text{unroll}(\boldsymbol{s}^{(i)}, \boldsymbol{\theta} + \boldsymbol{\epsilon}^{(i)}, K)$
    $\hat{\boldsymbol{g}}^{\text{ES-Single}} \leftarrow \hat{\boldsymbol{g}}^{\text{ES-Single}} + \boldsymbol{\epsilon}^{(i)} \hat{L}_K^{(i)}$
  end for
  $\hat{\boldsymbol{g}}^{\text{ES-Single}} \leftarrow \frac{1}{N\sigma^2} \hat{\boldsymbol{g}}^{\text{ES-Single}}$

  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \hat{\boldsymbol{g}}^{\text{ES-Single}}$
end while

ES-Single Properties

- ES-Single is *mathematically equivalent* to Full-Unroll ES, but *differs algorithmically*
  - ES-Single has the *same bias and variance characteristics as Full-Unroll ES*

Bias

Proposition 3.1 (ES-Single is unbiased). *Assume that $L(\boldsymbol{\theta})$ is quadratic and $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$ exists. Then, the ES-Single gradient estimator is unbiased, that is, $bias(\hat{\boldsymbol{g}}^{\text{ES-Single}}) = \mathbb{E}_{\epsilon}\left[\hat{\boldsymbol{g}}^{\text{ES-Single}}\right] - \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) = 0.$*

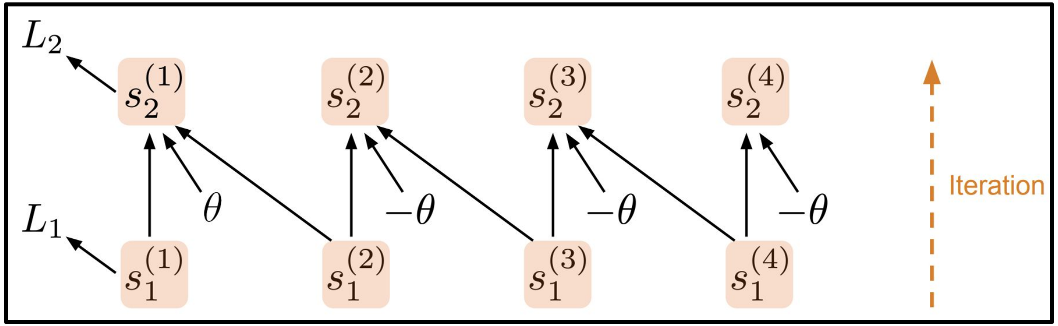
Proof. The proof is provided in Appendix D.1. □

Variance

Proposition 3.2 (ES-Single Variance). *The total variance of ES-Single is $tr(Var(\hat{\boldsymbol{g}}^{\text{ES-Single}})) = (P+1)\|\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})\|^2$, where P is the dimensionality of $\boldsymbol{\theta}$.*

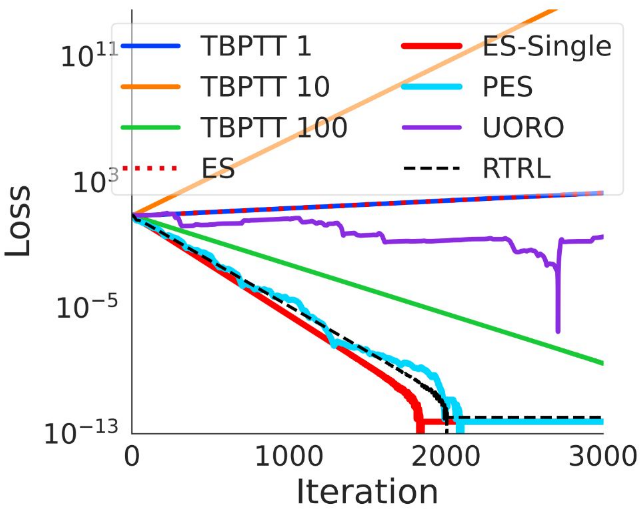
Proof. The proof is provided in Appendix D.2. □

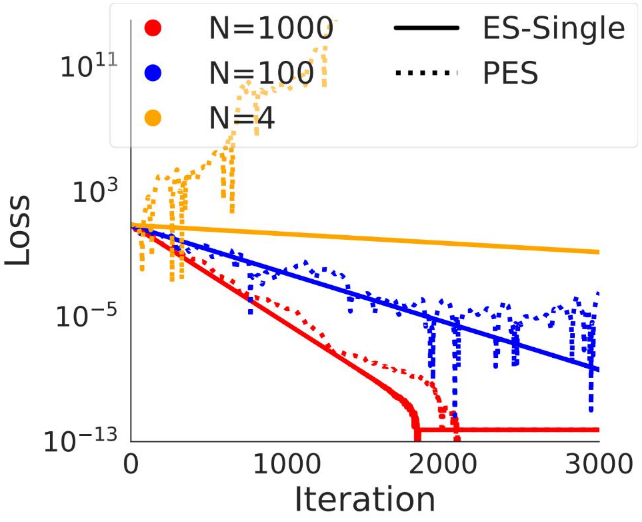
Influence Balancing: Task Setup



- **Task:** *Influence balancing*, introduced by Tallec & Olivier (2017)
  - Tune a scalar parameter θ that has a *negative influence in the short term*, but a *positive influence in the long term*
  - Designed such that *truncated methods move in the wrong direction*

Influence Balancing: Results



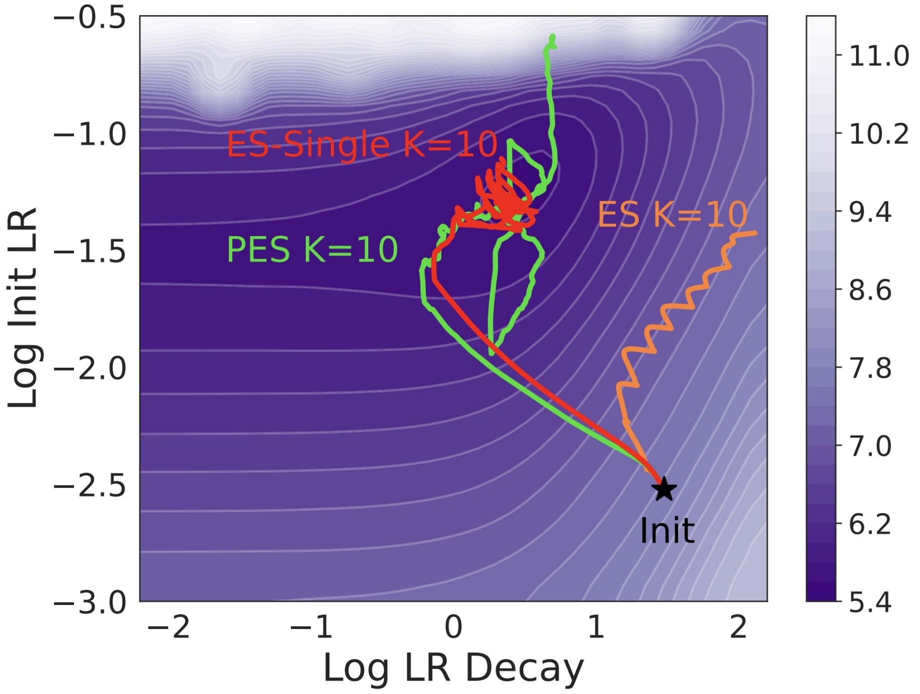


ES-Single is unbiased: it behaves like RTRL when using many particles

ES-Single is *more stable than PES when using fewer particles*

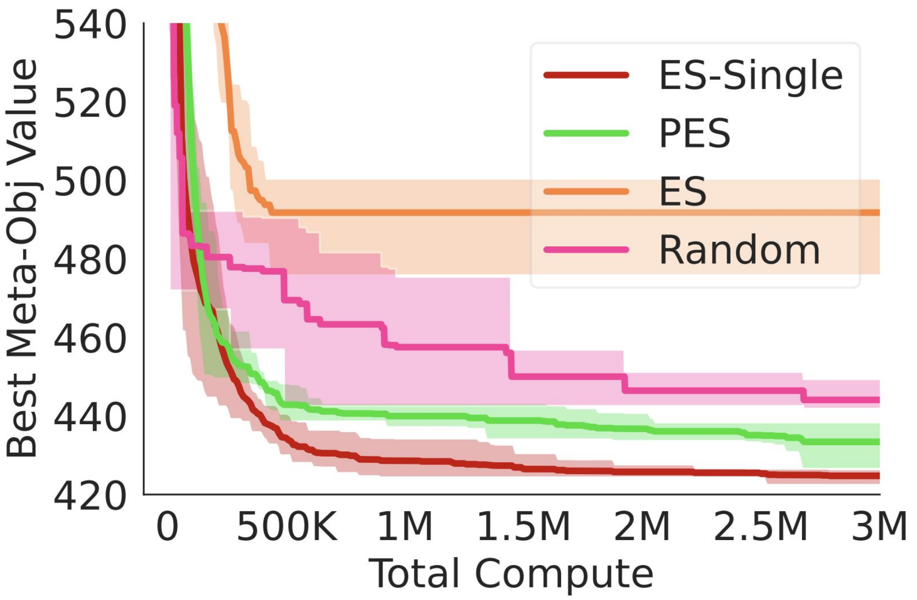
Meta-Optimizing an MNIST LR Schedule

- Meta-optimizing a learning rate schedule for training an MLP on MNIST
  - Tune the *initial learning rate and decay factor*
- ES-Single behaves similarly to PES, but has *more stable convergence at the optimum*



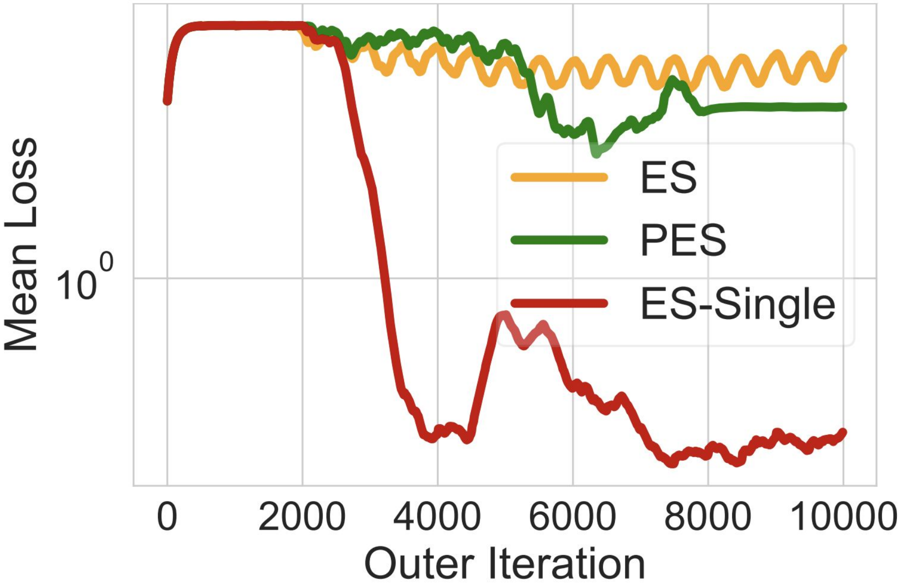
Optimizing Several Continuous & Discrete Hyperparams

- Training a 5-layer MLP on FashionMNIST
- *Optimizing 29 continuous and discrete hyperparameters*
    - Per-parameter block learning rates and momentum coefficients, and the number of hidden units per layer
- *ES-Single reaches lower meta-objective values using less total compute* than truncated ES, random search, or PES



Meta-Training a Learned Optimizer

- Meta-training a learned optimizer targeting the optimization of an MLP on FashionMNIST
- Here, T=5000 and K=10
- *ES-Single works in some scenarios where PES does not*



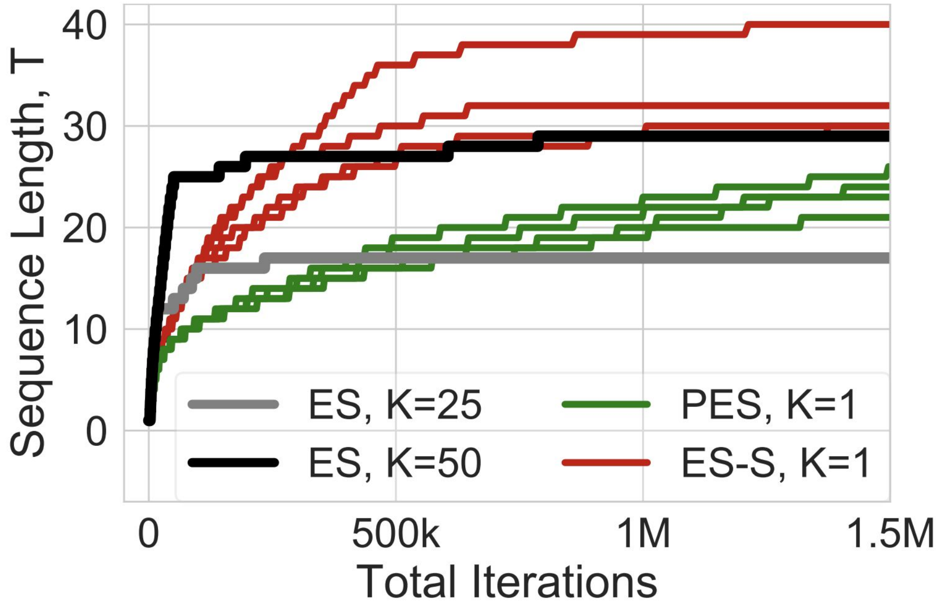
LSTM Copy Task

Copy Task

Input: `001101------`

Output: `------001101`

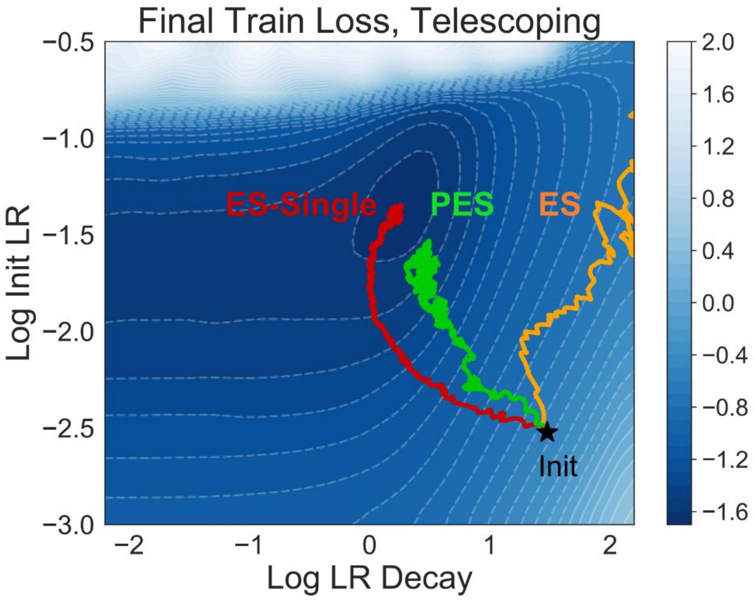
- **Challenge:** *Scaling to longer sequences*
- *Truncated methods* like TBPTT and truncated ES *fail to model long-term dependencies*
- PES with K=1 works, but is slow
- *ES-Single with K=1 is significantly faster than PES, and reaches larger T*

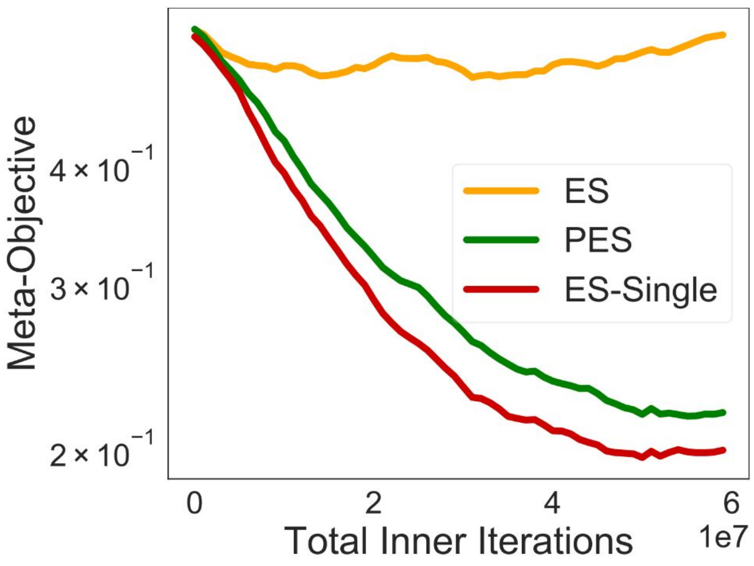


Telescoping Sums

- Can use telescoping sums to target the *final training loss*

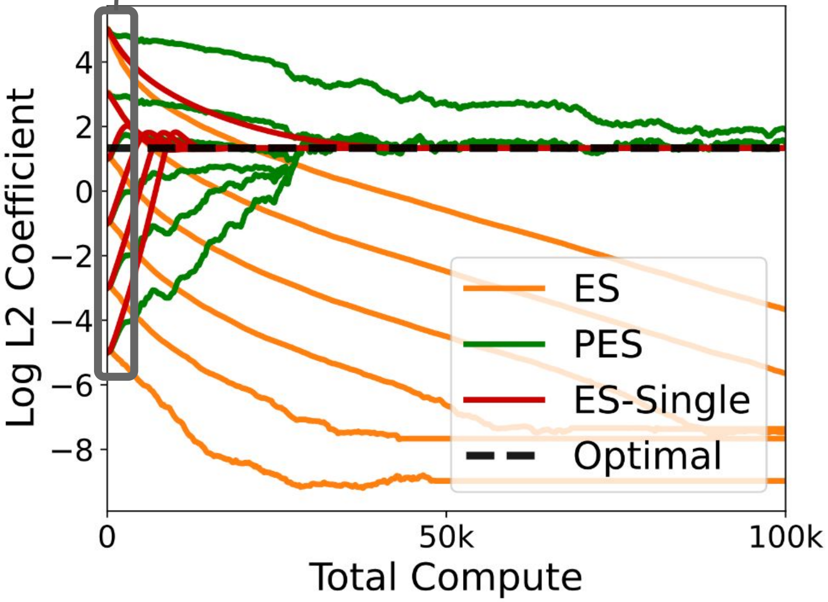
$$\boxed{\sum_{t=0}^{T} p_t = (\cancel{L_0} - L_{-1}) + \cdots + (L_T - \cancel{L_{T-1}}) = L_T}$$

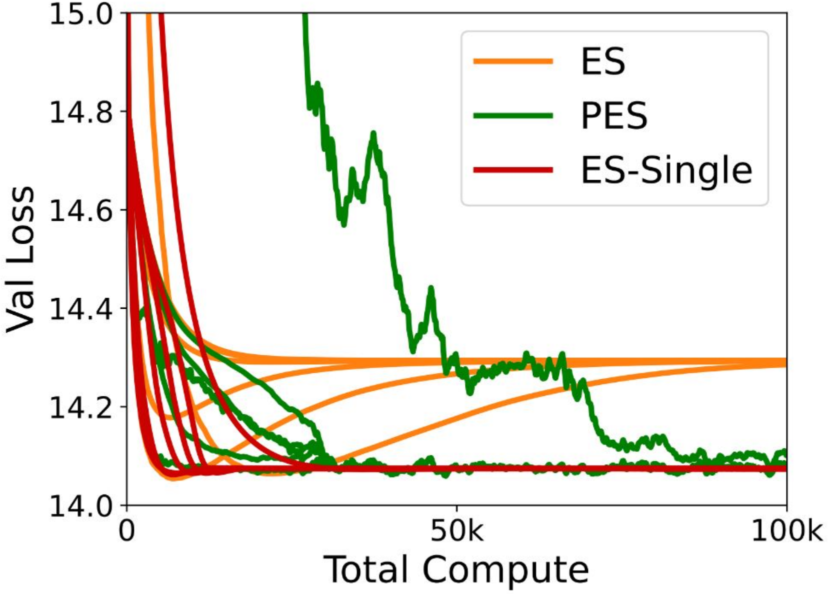




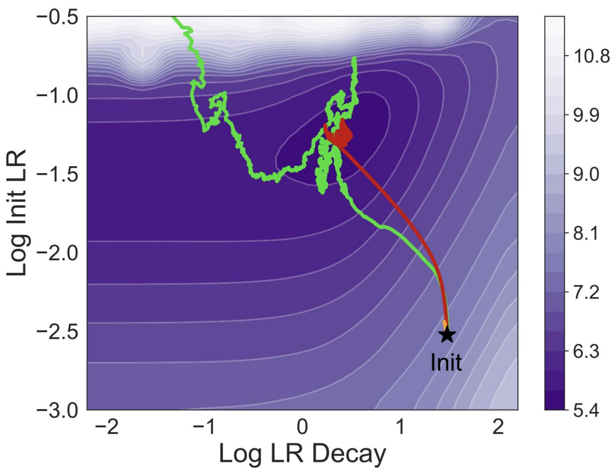
Tuning L_2 Regularization for UCI Regression

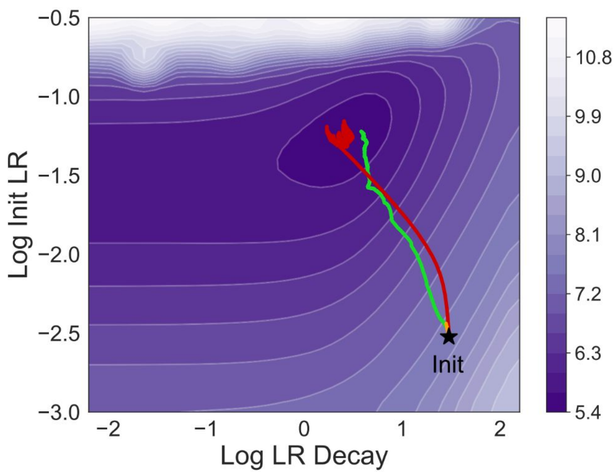
Different random initializations of the log L_2 coefficient

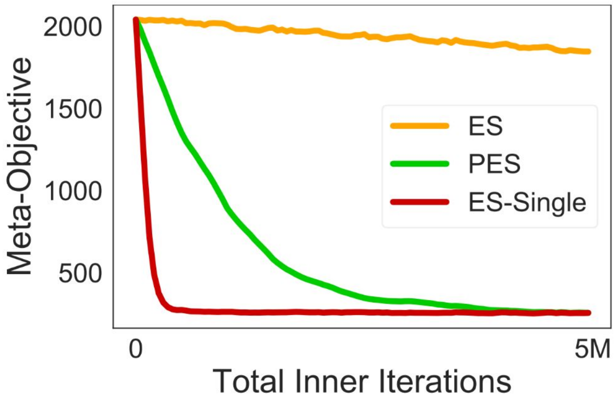




PES vs ES-Single: Stability and Performance







Large LR for PES

↓

Diverges after initially getting to the optimal region

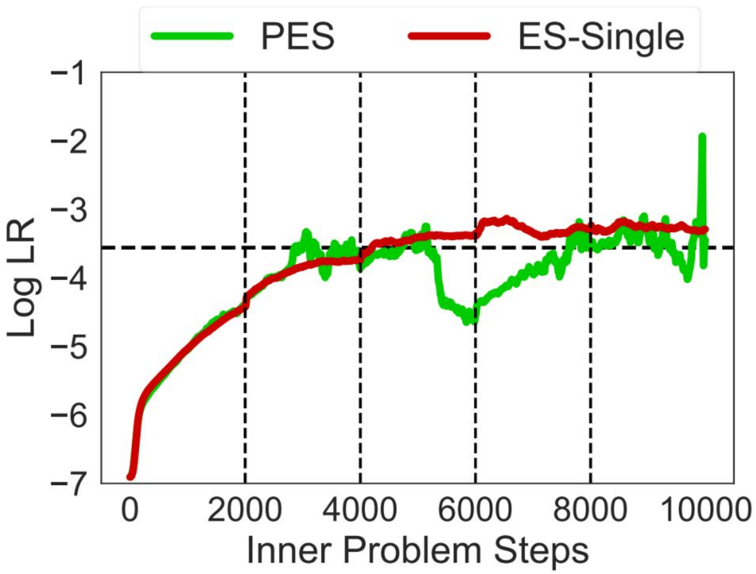
Small LR for PES

↓

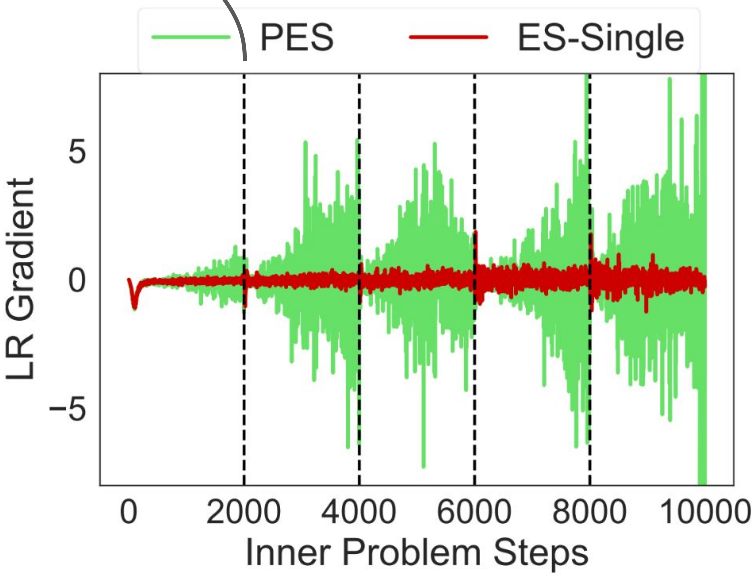
Stable convergence, but very slow compared to ES-Single

Meta-Gradient Comparison

Vertical lines denote the *start of a new inner problem* (T=2000)



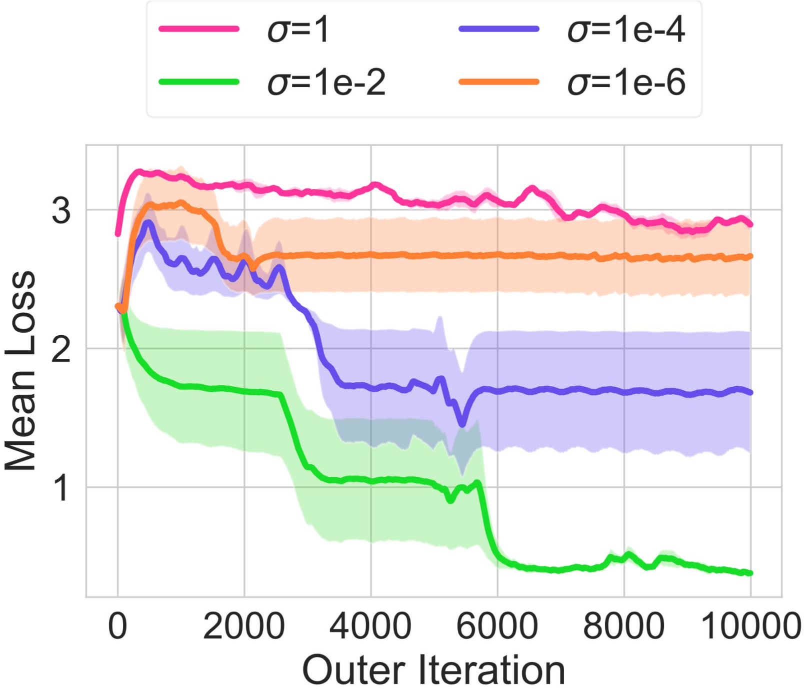
(a) Adaptation of the log-learning rate.



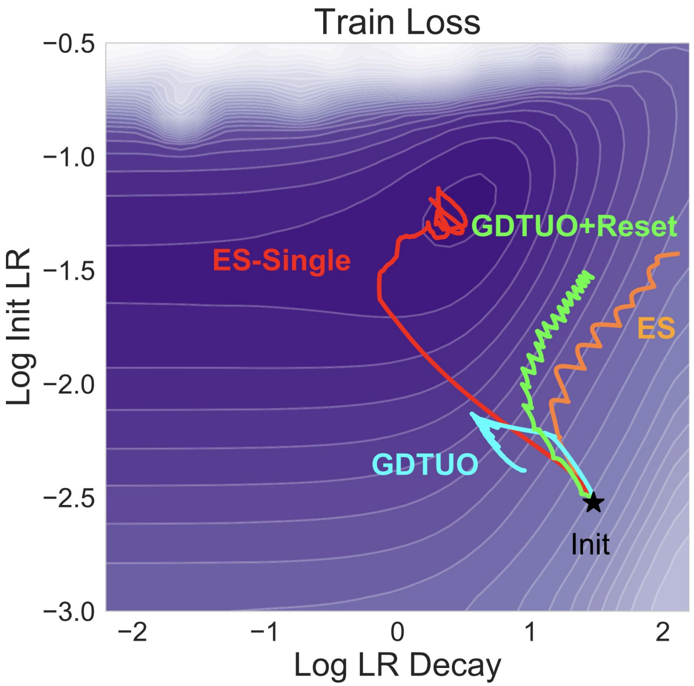
(b) PES and ES-Single meta-gradients over the course of multiple inner problems.

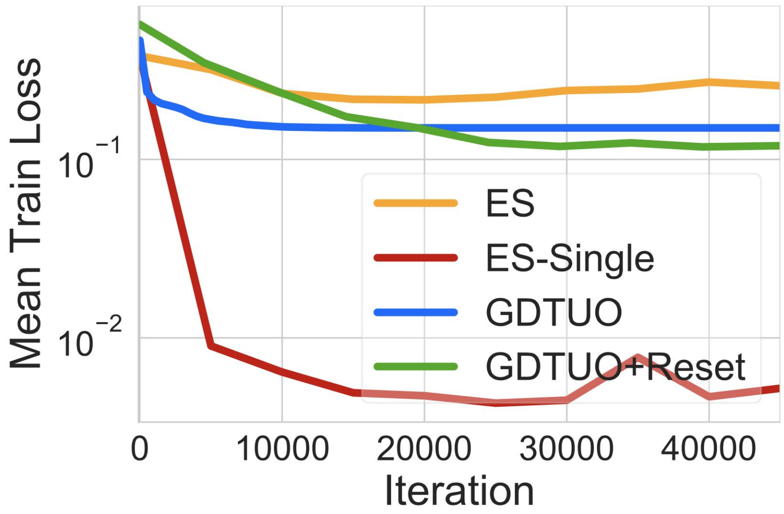
Importance of Smoothing

- Ablation over the perturbation scale σ, to optimize an MLP learned optimizer.
- *Small perturbation scales* lead to behavior similar to gradient-based methods, which may *get stuck in sub-optimal local minima in chaotic loss landscapes*.
  - For σ = 1e-6, meta-optimization fails to make progress
- In contrast, *using an appropriate scale, σ=1e-2, leads to stable convergence*



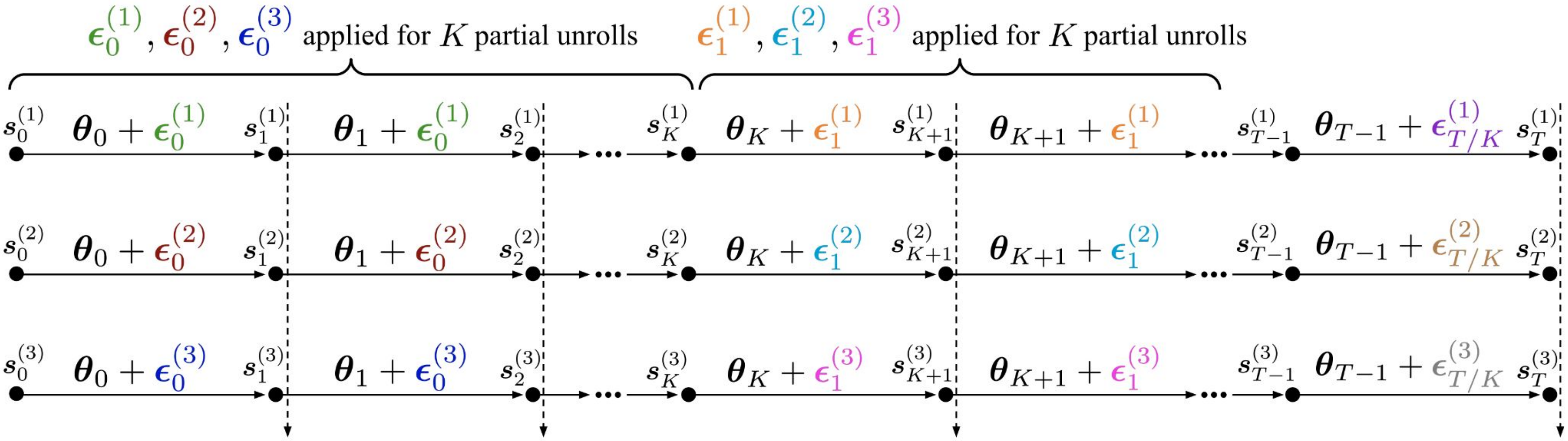
Comparison to a Gradient-Based Heuristic





- Here, we compared to *"Gradient Descent: The Ultimate Optimizer" (GDTUO)*
  - This is based on *Hypergradient Descent (HD), which adapts optimizer hyperparameters via a 1-step lookahead meta-objective*
- GDTUO is a gradient-based analogue to vanilla truncated ES, and behaves like ES

Generalization of PES and ES-Single



- This computation graph *generalizes ES-Single and PES*
- Uses the *same perturbation for K sequential truncated unrolls*
- After K unrolls, it adds the current perturbation to the perturbation accumulator, and samples a new perturbation

Generalization of PES and ES-Single: Algorithm

Algorithm 3 Truncated Evolution Strategies (ES) applied to partial unrolls of a computation graph.

Input: $\boldsymbol{s}_0$, initial state
K, truncation length for partial unrolls
N, number of particles
σ, standard deviation of perturbations
α, learning rate for outer optimization
Initialize $\boldsymbol{s} = \boldsymbol{s}_0$
while inner problem not finished **do**
  $\hat{\boldsymbol{g}}^{\text{ES}} \leftarrow \mathbf{0}$
  for $i = 1, \ldots, N$ **do**
    $\boldsymbol{\epsilon}^{(i)} = \begin{cases} \text{draw from } \mathcal{N}(0, \sigma^2 I) & i \text{ odd} \\ -\boldsymbol{\epsilon}^{(i-1)} & i \text{ even} \end{cases}$
    $\hat{L}_K^{(i)} \leftarrow \text{unroll}(\boldsymbol{s}, \boldsymbol{\theta} + \boldsymbol{\epsilon}^{(i)}, K)$
    $\hat{\boldsymbol{g}}^{\text{ES}} \leftarrow \hat{\boldsymbol{g}}^{\text{ES}} + \boldsymbol{\epsilon}^{(i)} \hat{L}_K^{(i)}$
  end for
  $\hat{\boldsymbol{g}}^{\text{ES}} \leftarrow \frac{1}{N\sigma^2} \hat{\boldsymbol{g}}^{\text{ES}}$
  $\boldsymbol{s} \leftarrow \text{unroll}(\boldsymbol{s}, \boldsymbol{\theta}, K)$
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \hat{\boldsymbol{g}}^{\text{ES}}$
end while

Algorithm 4 Generalization of ES-Single and PES, with an arbitrary re-sampling interval M.

Input: $\boldsymbol{s}_0$, initial state
K, truncation length for partial unrolls
M, re-sampling interval
N, number of particles
σ, standard deviation of perturbations
α, learning rate for outer optimization
Initialize $\boldsymbol{s}^{(i)} = \boldsymbol{s}_0$ for $i \in \{1, \ldots, N\}$
Initialize $\boldsymbol{\xi}^{(i)} \leftarrow \mathbf{0}$ for $i \in \{1, \ldots, N\}$
while inner problem not finished, iteration j **do**
  if $j \mod M = 0$ **then**
    for $i = 1, \ldots, N$ **do**
      $\boldsymbol{\epsilon}^{(i)} = \begin{cases} \text{draw from } \mathcal{N}(0, \sigma^2 I) & i \text{ odd} \\ -\boldsymbol{\epsilon}^{(i-1)} & i \text{ even} \end{cases}$
      $\boldsymbol{\xi}^{(i)} \leftarrow \boldsymbol{\xi}^{(i)} + \boldsymbol{\epsilon}^{(i)}$
    end for
  end if
  $\hat{\boldsymbol{g}}^{\text{ES-Gen}} \leftarrow \mathbf{0}$
  for $i = 1, \ldots, N$ **do**
    $\boldsymbol{s}^{(i)}, \hat{L}_K^{(i)} \leftarrow \text{unroll}(\boldsymbol{s}^{(i)}, \boldsymbol{\theta} + \boldsymbol{\epsilon}^{(i)}, K)$
    $\hat{\boldsymbol{g}}^{\text{ES-Gen}} \leftarrow \hat{\boldsymbol{g}}^{\text{ES-Gen}} + \boldsymbol{\xi}^{(i)} \hat{L}_K^{(i)}$
  end for
  $\hat{\boldsymbol{g}}^{\text{ES-Gen}} \leftarrow \frac{1}{N\sigma^2} \hat{\boldsymbol{g}}^{\text{ES-Gen}}$

  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \hat{\boldsymbol{g}}^{\text{ES-Gen}}$
end while

Conclusion

- ES-Single is a simple method for gradient estimation in unrolled computation graphs
- **Key idea:** sample a *single perturbation per particle at the start of each inner problem, and keep it fixed over all partial unrolls of the problem*
- ES-Single has *constant variance with respect to the number of partial unrolls* per inner problem
    - Addresses a key challenge faced by PES, and makes it scalable to long inner problems with short unrolls
- Empirically, *ES-Single outperforms PES on a range of tasks*, including hyperparameter optimization and RNN training

https://colab.research.google.com/drive/1fgSzwaIXfJKbYTntEFfNbc2UuTXwEw0A?usp=sharing

Thank you!