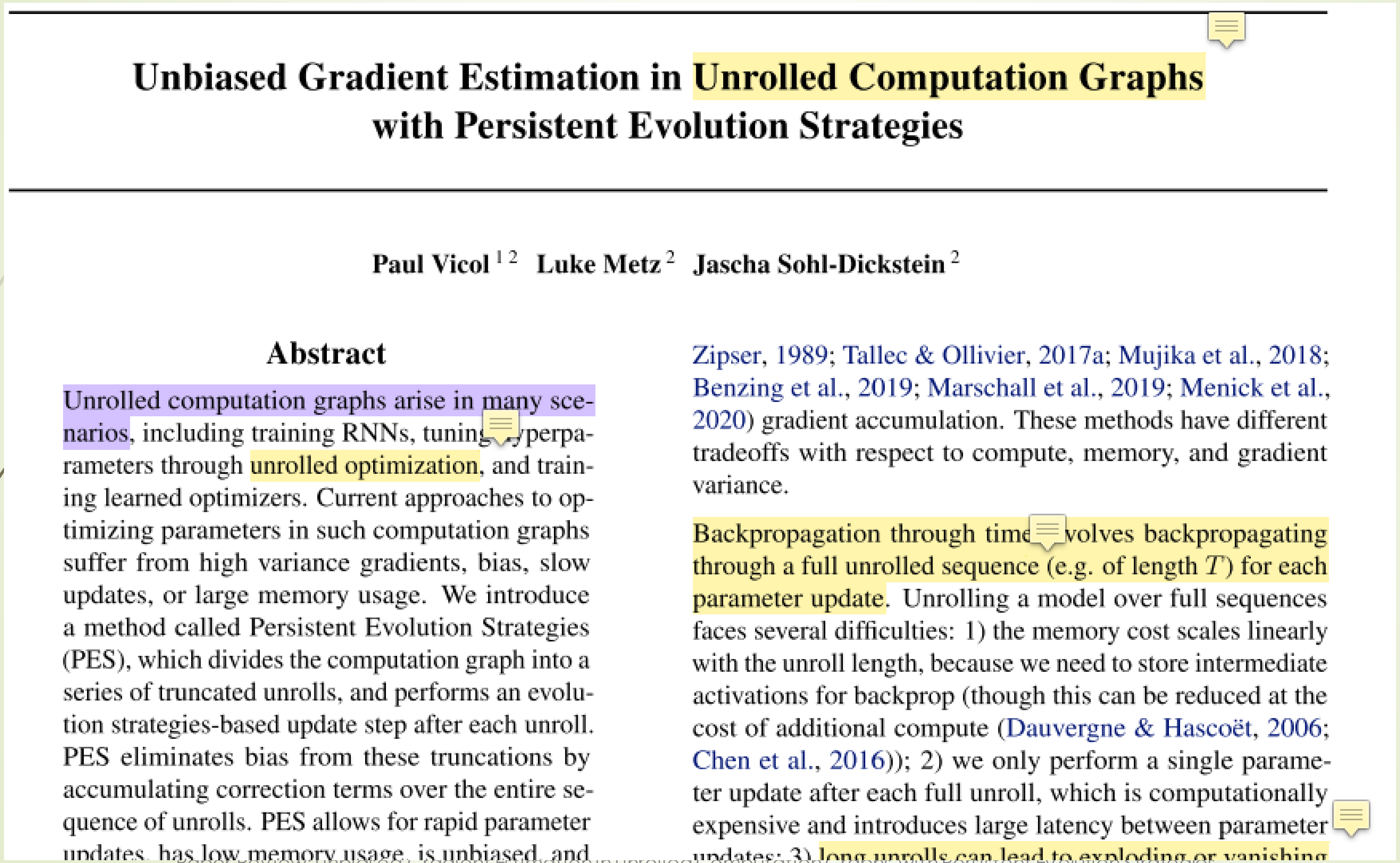
Paper Review: Unbiased Gradient Estimation in Unrolled Computation Graphs with Persistent Evolution Strategies

Abdullah Mamun

August 23, 2021

About this paper



Unbiased Gradient Estimation in Unrolled Computation Graphs with Persistent Evolution Strategies

Paul Vicol [1 2] **Luke Metz** [2] **Jascha Sohl-Dickstein** [2]

Abstract

Unrolled computation graphs arise in many scenarios, including training RNNs, tuning hyperparameters through unrolled optimization, and training learned optimizers. Current approaches to optimizing parameters in such computation graphs suffer from high variance gradients, bias, slow updates, or large memory usage. We introduce a method called Persistent Evolution Strategies (PES), which divides the computation graph into a series of truncated unrolls, and performs an evolution strategies-based update step after each unroll. PES eliminates bias from these truncations by accumulating correction terms over the entire sequence of unrolls. PES allows for rapid parameter updates, has low memory usage, is unbiased, and

Zipser, 1989; Tallec & Ollivier, 2017a; Mujika et al., 2018; Benzing et al., 2019; Marschall et al., 2019; Menick et al., 2020) gradient accumulation. These methods have different tradeoffs with respect to compute, memory, and gradient variance.

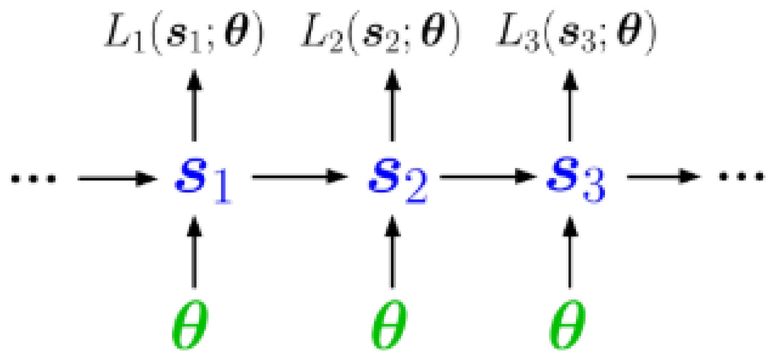
Backpropagation through time involves backpropagating through a full unrolled sequence (e.g. of length T) for each parameter update. Unrolling a model over full sequences faces several difficulties: 1) the memory cost scales linearly with the unroll length, because we need to store intermediate activations for backprop (though this can be reduced at the cost of additional compute (Dauvergne & Hascoët, 2006; Chen et al., 2016)); 2) we only perform a single parameter update after each full unroll, which is computationally expensive and introduces large latency between parameter updates; 3) long unrolls can lead to exploding or vanishing

About this paper

- Published in the **ICML** conference in 2021. Awarded the outstanding paper award.
- Cited by **2** as of August 23, 2021
- Main goal is to propose a new optimization method for recurrent models.

Introduction

Unbiased Gradient Estimation in Unrolled Computation Graphs with Persistent Evolution Strategies



$L_1(\boldsymbol{s}_1;\boldsymbol{\theta})$
$L_2(\boldsymbol{s}_2;\boldsymbol{\theta})$
$L_3(\boldsymbol{s}_3;\boldsymbol{\theta})$
$\boldsymbol{s}_1$
$\boldsymbol{s}_2$
$\boldsymbol{s}_3$
$\boldsymbol{\theta}$
$\boldsymbol{\theta}$
$\boldsymbol{\theta}$

Objective: $L(\boldsymbol{\theta}) = \sum_t L_t(\boldsymbol{s}_t;\boldsymbol{\theta})$

	$\boldsymbol{s}_t$	$\boldsymbol{\theta}$
RNN	RNN Hidden State	RNN Parameters
Unrolled Optimization	Model Parameters	Hyperparameters

Figure 1. **An unrolled computation graph**, illustrating how both RNNs and unrolled optimization can be described using Equations 1 and 2. In RNN training, $\boldsymbol{s}_t$ is the hidden state of the RNN, $L_t(\cdot)$ is the prediction cross entropy at each timestep, $\boldsymbol{\theta}$ refers to the RNN parameters, and f corresponds to the forward pass of the RNN, that takes an input and the previous hidden state ($\boldsymbol{s}_t$) and returns a new hidden state ($\boldsymbol{s}_{t+1}$). In unrolled optimization, $\boldsymbol{s}_t$ contains the parameters of the base model and optimizer accumulators (e.g. momentum), $L_t(\cdot)$ is a meta-objective such as validation performance, $\boldsymbol{\theta}$ contains hyperparameters (e.g. the learning rate, weight decay, etc.) that govern the optimization, and f corresponds to the update step of an optimization algorithm such as SGD, RMSprop (Tieleman & Hinton, 2012), or Adam (Kingma & Ba, 2015).

Motivation

- Usual loss:

$$L(\boldsymbol{\theta}) = \sum_{t=1}^{T} L_t(\boldsymbol{s}_t; \boldsymbol{\theta})$$

- Gradient vanishing/exploding problem with longer sequences.
- Evolution strategies solve the gradient vanishing problem but computationally expensive
- Truncated evolution strategies are computationally cheap but introduces a bias.

Method

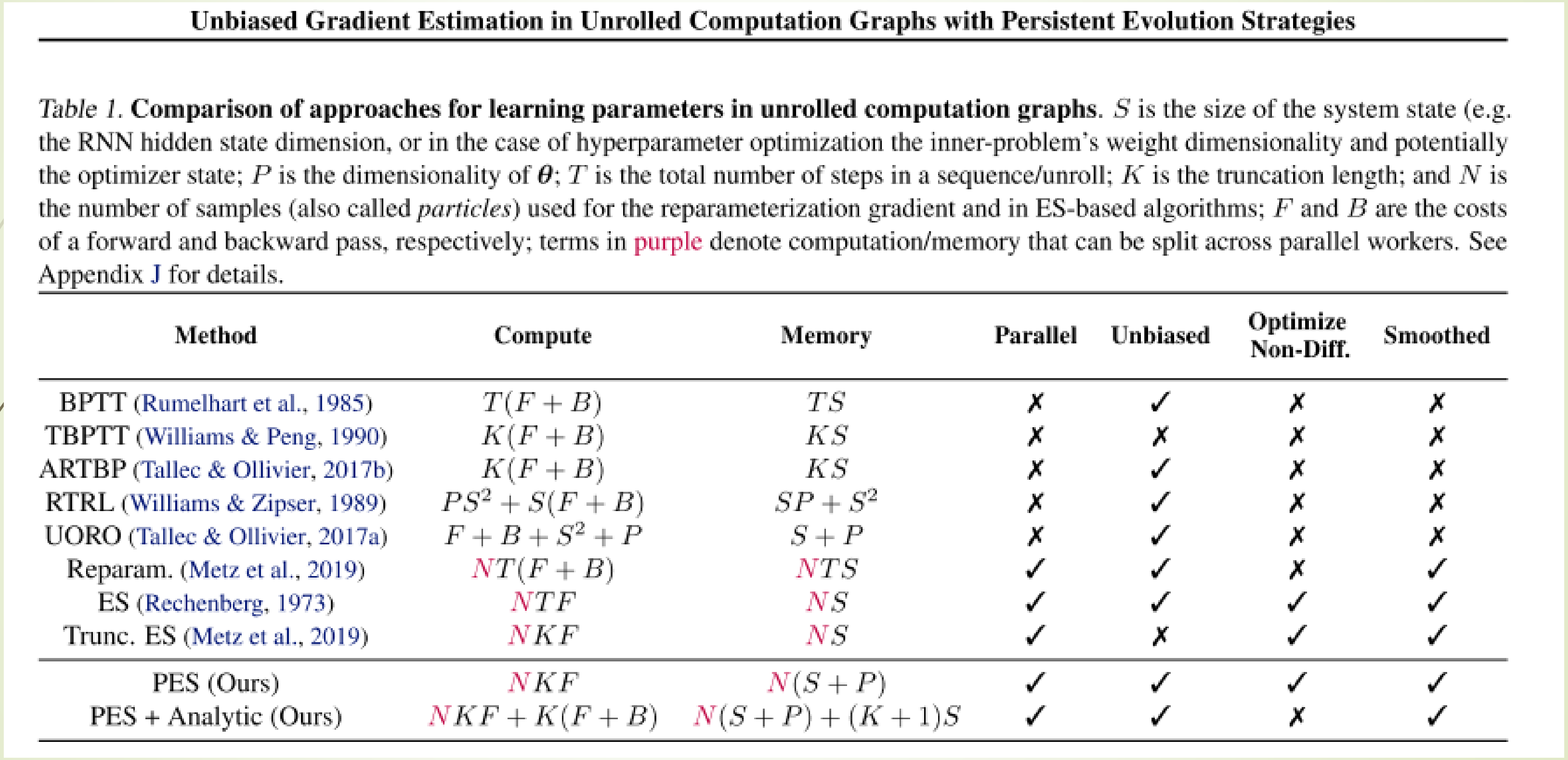
- Main feature is that it keeps different parameter for each time-step

$$\frac{dL(\boldsymbol{\theta})}{d\boldsymbol{\theta}} = \sum_{\tau=1}^{T} \frac{\partial L(\Theta)}{\partial \boldsymbol{\theta}_\tau}$$

- Usual loss before:

$$L(\boldsymbol{\theta}) = \sum_{t=1}^{T} L_t(\boldsymbol{s}_t; \boldsymbol{\theta})$$

Comparison



Unbiased Gradient Estimation in Unrolled Computation Graphs with Persistent Evolution Strategies

Table 1. **Comparison of approaches for learning parameters in unrolled computation graphs.** S is the size of the system state (e.g. the RNN hidden state dimension, or in the case of hyperparameter optimization the inner-problem's weight dimensionality and potentially the optimizer state; P is the dimensionality of $\boldsymbol{\theta}$; T is the total number of steps in a sequence/unroll; K is the truncation length; and N is the number of samples (also called *particles*) used for the reparameterization gradient and in ES-based algorithms; F and B are the costs of a forward and backward pass, respectively; terms in purple denote computation/memory that can be split across parallel workers. See Appendix J for details.

Method	Compute	Memory	Parallel	Unbiased	Optimize Non-Diff.	Smoothed
BPTT (Rumelhart et al., 1985)	$T(F+B)$	TS	✗	✓	✗	✗
TBPTT (Williams & Peng, 1990)	$K(F+B)$	KS	✗	✗	✗	✗
ARTBP (Tallec & Ollivier, 2017b)	$K(F+B)$	KS	✗	✓	✗	✗
RTRL (Williams & Zipser, 1989)	$PS^2+S(F+B)$	$SP+S^2$	✗	✓	✗	✗
UORO (Tallec & Ollivier, 2017a)	$F+B+S^2+P$	$S+P$	✗	✓	✗	✗
Reparam. (Metz et al., 2019)	$NT(F+B)$	NTS	✓	✓	✗	✓
ES (Rechenberg, 1973)	NTF	NS	✓	✓	✓	✓
Trunc. ES (Metz et al., 2019)	NKF	NS	✓	✗	✓	✓
PES (Ours)	NKF	$N(S+P)$	✓	✓	✓	✓
PES + Analytic (Ours)	$NKF+K(F+B)$	$N(S+P)+(K+1)S$	✓	✓	✗	✓

Result

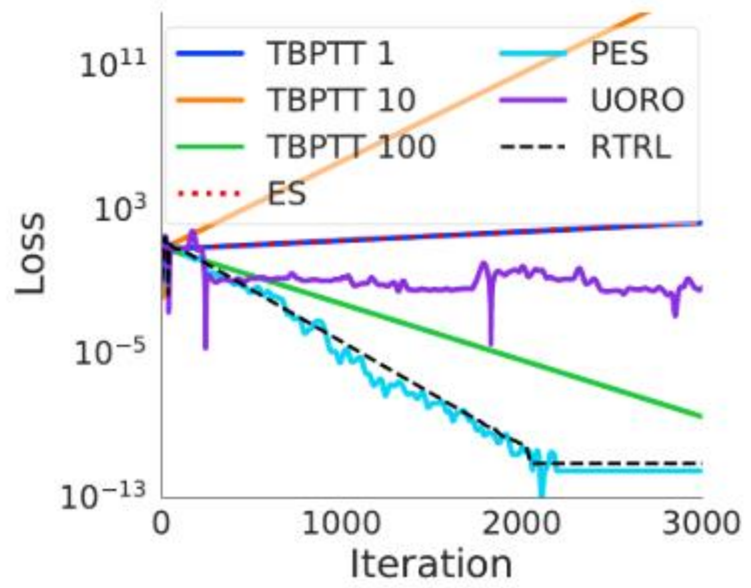


Figure 5. **Loss curves for the influence balancing task.** TBPTT with short truncations diverges, while PES performs nearly identically to exact RTRL. See Section 5.1 for experiment details.

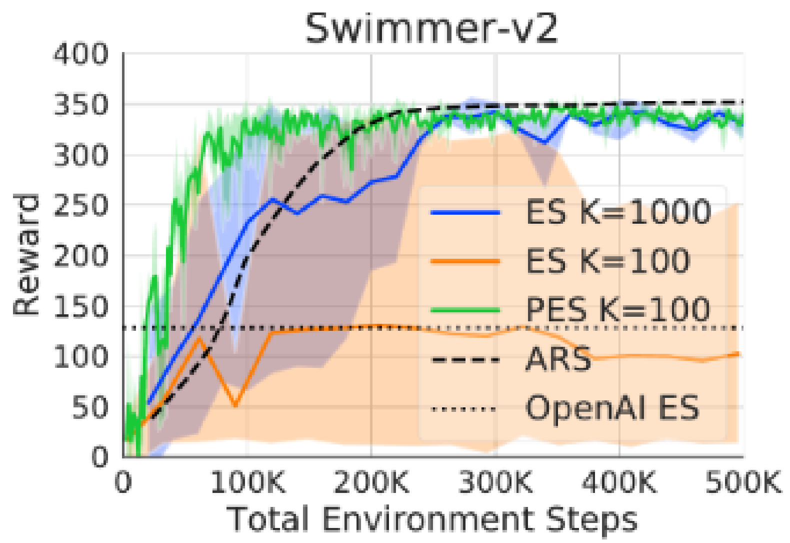


Figure 8. **Learning a policy for continuous control**. We find that PES is more efficient than ES applied to full episodes, while truncated ES fails due to bias. We plot the ARS V1 result from Mania et al. (2018) (dashed curve) to show that our full-unroll baseline is comparable to theirs. The dotted line shows the maximum reward reported for the ES approach in Salimans et al. (2017), which does not solve the Swimmer task. See Section 5.3 for details.

Contributions

Thank You!

Contact: abdullahal.mamun1@wsu.edu