Robust Stochastic Gradient Descent With Student-t Distribution Based First-Order Momentum

Wendyam Eric Lionel Ilboudo, Taisuke Kobayashi, *Member, IEEE*, and Kenji Sugimoto, *Member, IEEE*

***Abstract*— Remarkable achievements by deep neural networks stand on the development of excellent stochastic gradient descent methods. Deep-learning-based machine learning algorithms, however, have to find patterns between observations and supervised signals, even though they may include some noise that hides the true relationship between them, more or less especially in the robotics domain. To perform well even with such noise, we expect them to be able to detect outliers and discard them when needed. We, therefore, propose a new stochastic gradient optimization method, whose robustness is directly built in the algorithm, using the robust student-t distribution as its core idea. We integrate our method to some of the latest stochastic gradient algorithms, and in particular, Adam, the popular optimizer, is modified through our method. The resultant algorithm, called t-Adam, along with the other stochastic gradient methods integrated with our core idea is shown to effectively outperform Adam and their original versions in terms of robustness against noise on diverse tasks, ranging from regression and classification to reinforcement learning problems.**

***Index Terms*— Deep neural networks, robust optimization, stochastic gradient descent (SGD), student-t distribution.**

I. Introduction

THE field of machine learning which aims to find optimal (minimum or maximum) solutions is undoubtedly dominated by first-order optimization methods based on the gradient descent algorithm and particularly [1], its stochastic variant, the stochastic gradient descent (SGD) method [2]. The popularity of the SGD algorithm comes from its simplicity, its computational efficiency with respect to second-order methods, its applicability to online training, and its convergence rate that is independent of the training data set. In addition, SGD has a high affinity with deep learning [3], where network parameters are updated by backpropagation of their gradients, and is intensively used to train large deep neural networks. In other words, as a remark, the performance of deep learning greatly depends on SGD.

Despite such established popularity, a specific trait of SGD is the inherent noise, coming primarily from sampling training points and secondly from data drawn from noisy processes. In the robotics field, for example, sensory noise is not ignorable [4]; automatic annotation would often be wrong [5]; and control policies optimized by reinforcement learning (RL) [6] have no accurate supervised signals and the updates are done toward target values estimated from noisy data. Although numerous data can eliminate the adverse effects of noise, real robots have no time to collect them for specific tasks. Hence, robotics agents, forced to learn from a few noisy samples, best reveal the adverse effects of noise on SGD.

Many of the new optimizers proposed to improve the SGD algorithm and tackle complex training scenarios where gradient descent methods behave poorly also share the same weakness to aberrant values. Adam (Adaptive moment estimates) [7], one of the most widely used and practical optimizers for training deep learning models, is no exception, despite its well-defined convergence proof [8]. This is mainly due to the insufficient number of samples implicitly involved in its first-moment evaluation.

Therefore, robust SGD methods have been proposed to solve this problem. In particular, the work of Holland *et al.* [9] comes close to our current proposition, by relying on robust estimates of the gradients in order to stabilize the training and deal with heavy-tailed data sets. Their method is based on a convenient class of M-estimators for the location and scale parameters and uses two principal steps: a rescale stage to estimate the standard deviation, and a locate stage to compute the location gradient based on the previously obtained standard deviation. However, the parameters are computed based on arbitrary even functions that must be chosen beforehand to define the conditions they must fulfill.

In this article, we newly propose robust estimates of the first-order momentum of the gradients, which is used in the state-of-the-art SGD methods to stabilize and accelerate learning. The key idea for such robust estimates is the use of a student-t distribution, which is a model suitable for the estimates from a few samples [10]. The conventional way through which the first-order momentum is estimated, i.e. the exponential moving average, is regarded as the update of the location parameter of a normal distribution, which is sensitive to outliers. Hence, we replace it with the update of the location of the student-t distribution (let us call it t-momentum). By simply doing so, the estimates of the first-order momentum automatically exclude aberrant gradients computed from outliers, while normal gradients are used as before.

Three general problems for machine learning, regression, classification, and RL, are solved to verify our proposal. In

Manuscript received March 31, 2020; revised August 28, 2020 and October 1, 2020; accepted November 25, 2020. Date of publication December 16, 2020; date of current version March 1, 2022. *(Corresponding author: Wendyam Eric Lionel Ilboudo.)*

The authors are with the Division of Information Science, Nara Institute of Science and Technology, Nara 630-0192, Japan (e-mail: ilboudo.wendyam_eric.in1@is.naist.jp; kobayashi@is.naist.jp; kenji@is.naist.jp).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2020.3041755.

Digital Object Identifier 10.1109/TNNLS.2020.3041755

regression and classification problems, noisy data deteriorate the performance of the conventional SGD methods while the new SGD methods with the t-momentum resist the adverse effects of noise. In addition, although RL with the conventional SGD methods has to set the learning rate small in order to avoid wrong updates, RL with t-momentum-used Adam (named t-Adam) succeeds even with the default learning rate value which is typically larger.

II. Background and Previous Works

A. Background

1) Stochastic Gradient Descent: Let x_t be a random sample from the data set at iteration t, $J_\theta(x_t)$ the objective function evaluated on data x_t with the parameters θ, $g_t = \nabla_\theta J_\theta(x_t)$ its gradient, and α the learning rate. The SGD algorithm [2] updates θ_{t-1} to θ_t through the following update rule:

$$\theta_t = \theta_{t-1} - \alpha g_t. \tag{1}$$

This algorithm yields at least a local minima of J with respect to θ.

2) Improving SGD: Since its proposition, many ideas have been developed in order to improve the convergence property of the SGD algorithm. This feature heavily connects to the fluctuations of the gradients during learning and all the research that aims to accelerate the convergence rate have done so through several approaches. For instance, they improved 1) the update method of the parameters [11]–[14]; 2) the adjustment of the learning rate [15]–[18]; and 3) the robustness to aberrant values from heavy-tailed data [9], [19]–[21]. Those approaches have culminated into some pretty effective state-of-the-art first-order optimization methods, going from the momentum idea to the adaptive learning rate and variance reduction schemes. Below, we review some of the works related to 3) the robustness.

B. Previous Works

As stated before, SGD is inherently noisy and susceptible to produce bad minima estimates when facing aberrant gradient estimates. A lot of work has therefore been done to propose more robust methods for efficient machine learning under noise or data with heavy tails.

In this review, we ignore the general statistical methods for robust mean estimates [22] such as the median-based estimations [23]–[25] due to their practical limitations. Three main approaches are distinguished: 1) methods based on direct robust estimates of the loss (or risk) function [26]; 2) methods based on robust estimates of the gradients [9], [27] among which falls our algorithm; and 3) methods with small learning rates for wrong gradient estimates [21].

1) Robust Risk Estimation: In this approach, aberrant losses are directly ignored. Those methods usually require the use of all the available data in order to produce, for each parameter, a robust estimate of the loss function to be minimized. A specific inconvenient trait of this approach is the implicit definition of the robust estimate, which may introduce some computational roadblocks. As briefly explained by Holland *et al.* [9] since the estimates do not need to be convex even in the case where the loss function is, the nonlinear optimization can be both unstable and costly in high dimensions.

2) Robust Gradient Descent: This approach usually relies on the replacement of the empirical mean (first moment) gradient estimate with a more robust alternative, and simply differs in the method used to achieve this objective. Chen *et al.* [27] proposed the use of the geometric median of the gradients mean to aggregate multiple candidates. Using the same strategy, Prasad *et al.* [28] proposed a class of gradient estimator based on the idea that the gradient of a population loss could be regarded as the mean of a multivariate distribution, reducing the problem of gradient estimation to a multivariate mean estimation problem. Very close to our approach, Holland *et al.* [9] proposed to carefully reduce the effect of aberrant values instead of discarding them, which can also result in unfortunate discards of valuable data.

3) Adaptive Learning Rate: This approach is to reduce the effect of wrong gradient estimates by reducing the learning rate. One such approach has been proposed by Haimin *et al.* [21]. This method uses an exponential moving average (EMA) of the absolute value of the ratio between the current loss l_t and the past one l_{t-1} to scale the learning rate. This ratio corresponds to a relative prediction error and a large value implies a suspicious outlier. Haimin *et al.* [21] therefore proposed to divide the learning rate by it, so that a large relative prediction error would lead to a smaller effective learning rate. However, this strategy allows the outliers to modify the estimated gradient mean, and then uses the impact of the deviated mean on the loss function to reduce the effect on subsequent updates.

4) Our Contribution: As one of the problems in the EMA scheme, the lack of robustness has been dealt with in [19] and [20]. In those methods, the exponential decay parameter of the EMA is increased whenever a value that falls beyond some boundary is encountered. The common drawback in this strategy is that all outlier gradients are treated equally and discretely without consideration of how far they are from the normal values, and the boundary over which data would be considered to be an outlier must be set manually before training.

To the best of our knowledge, our approach, named t-momentum, is the first to employ the student-t distribution for the estimates of the first-order momentum, which is conventionally given through the EMA scheme. The main advantage of this approach is that it relies on the natural robustness of the student-t distribution and its ability to deal with outliers, and can easily be reduced to the conventional momentum for nonheavy-tailed data. Since the EMA-based first-order momentum is the key of the state-of-the-art SGD methods, our t-momentum can be integrated to various methods like Adam [7], RMSProp [17], VSGD-fd [19], Adasecant [20] or Adabound [18]. Specifically, in this article, we mainly focus on Adam with t-momentum, named t-Adam, to investigate its theoretical performance.

III. Proposal

Notation: We use $g = \nabla_\theta f(\theta)$ to denote the first derivative (gradient) of the function $f(\cdot)$ with respect to the

vector θ. For a vector x, x^2 refers to an element-wise square and the ith element of the vector x_t is referred to either as x_t^i or $x_{t,i}$. $\mathbb{E}[\cdot]$ is used for the expectation, and θ^* refers to the optimal parameters vector. $\|x\|$ is the euclidean norm, while $\|x\|_p$ refers to the p-norm. Finally, $\sum_j^d$ is used as a shorthand to $\sum_{j=1}^d$ to denote a sum over j, and we say a set F has a bounded diameter D_∞ if $\|x-y\|_\infty \leq D_\infty$ for all $x, y \in F$.

A. Adaptive Moment Estimation: Adam

Before describing our proposal, let us introduce Adam [7], the baseline of t-Adam, and the most popular EMA-based momentum method, to make our target clear. Adam is a popular method that combines the advantages of SGD with momentum along with those of adaptive learning rate methods [16], [17]. Its update rule is implemented as follows:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{2}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) {g_t}^2 \tag{3}$$

$$\hat{m}_t = m_t/(1-\beta_1^t), \quad \hat{v}_t = v_t/(1-\beta_2^t) \tag{4}$$

$$\theta_t = \theta_{t-1} - \alpha \frac{\hat{m}_t}{(\sqrt{\hat{v}_t}+\epsilon)} \tag{5}$$

where m_t is the first-order momentum (i.e., mean of gradients) and v_t is the second-order momentum utilized to adjust learning rates at time step t. β_1 and β_2 are the exponential decay rates (by default 0.9 and 0.999, respectively). α is the global learning rate and ϵ (typically 10^{-8}) is a small value added to avoid division by zero.

Although the use of EMAs in (2) and (3) makes the gradients smooth and reduces the fluctuations inherent to SGD, they are also sensitive to outliers. In particular, EMA with a small value like β_1 (=0.9) implicitly includes a few samples, hence its momentum m_t is very likely to be pulled out by outliers and easily deviate from the true average. This fluctuation makes learning unstable, and therefore, more robust learning techniques are needed.

B. Overview

Our proposition relies on the fact that the EMA, like (2) and (3), can be regarded as an incremental update law of the mean in the normal distribution with a fixed number of samples. Indeed, given n i.i.d. random samples $x_1, x_2, \ldots, x_n$ of dimension d with the assumption that they follow a normal distribution with unknown mean μ and covariance Σ, the log likelihood is given by

$$\log p(x_1, \ldots, x_n | \mu, \Sigma) = -\frac{nd}{2}\log(2\pi) - \frac{n}{2}\log(|\Sigma|) - \frac{1}{2}\sum_{i=1}^{n}(x_i-\mu)^T \Sigma^{-1}(x_i-\mu). \tag{6}$$

Taking and solving the derivative with respect to the mean μ equals to zero yields

$$\frac{\partial \log p}{\partial \mu} = \sum_{i=1}^{n} \Sigma^{-1}(x_i-\mu) \stackrel{\text{set}}{=} 0 \tag{7}$$

$$\implies \sum_{i=1}^{n}(x_i-\mu) = \sum_{i=1}^{n} x_i - n\mu = 0 \tag{8}$$

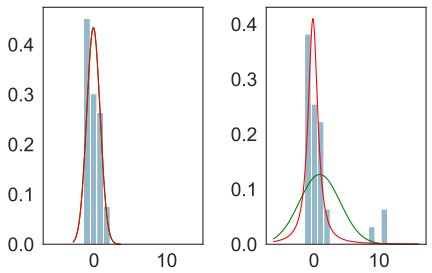


Fig. 1. Robustness to outliers: the normal distribution (in green) was pulled out by outliers; in contrast, the student-t distribution (in red) allowed their existence and hardly moved.

$$\implies \mu = \frac{1}{n}\sum_{i=1}^{n} x_i. \tag{9}$$

Let us denote by μ_n the estimated mean obtained after seeing n samples. In that case, the arithmetic mean of (9) can be converted to an iterative update

$$\mu_n = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{1}{n}x_n + \frac{1}{n-1}\sum_{i=1}^{n} x_i \tag{10}$$

$$= \frac{1}{n}x_n + \frac{1}{n}\frac{n-1}{n-1}\sum_{i=1}^{n-1} x_i = \frac{1}{n}x_n + \frac{1}{n}(n-1)\mu_{n-1} \tag{11}$$

$$= \left(1 - \frac{n-1}{n}\right)x_n + \frac{n-1}{n}\mu_{n-1}. \tag{12}$$

If we use the following change of variable, $(n-1/n) = \beta$, then we can write

$$\mu_n = (1-\beta)x_n + \beta\mu_{n-1} \tag{13}$$

which has the same form as an EMA. In this form, a fixed value of β uniquely defines a fixed number of samples n, due to the relation $\beta = (n-1/n)$. By analogy to this functional form, a regular EMA can be seen as an estimated gaussian mean that employs a fixed number of recent samples (defined by the decay coefficient β) and in particular, the fact that every new observation is given the same weight $(1-\beta)$ is a feature inherited from the normal distribution (through the arithmetic mean).

The sensitivity of Adam and other EMA-based momentum methods to aberrant gradient values is therefore just a feature inherited from the normal distribution, which is itself also sensitive to outliers.

In order for the EMA-based momentum to be robust, the distribution of the gradients must be assumed to come from a robust probability distribution that can yield a robust mean estimator. We, therefore, propose to replace the normal distribution momentum estimator with one drawn from the student-t distribution, which is well-known to be a robust probability distribution [10], [29], [30], as shown in Fig. 1, and a general form of the normal distribution. From the next section, we describe how the EMA is replaced using the student-t distribution, and the features of our implementation are subsequently analyzed. A pseudo-code of the t-momentum is given in Algorithm 1 and its integration to the popular Adam optimizer is summarized in Algorithm 2.

Algorithm 1 t-Momentum: Student-t Based Exponential Moving Average Momentum Algorithm

Input: Gradient g_t, Previous t-EMA: m_{t-1}
Input: Previous weight sum: W_{t-1}
Input: Previous variance: σ_{t-1}^2
Require: k: Scale factor for the degrees of freedom
Require: β: EMA decay parameter
$d \leftarrow dim[g_t]$
$\nu \leftarrow kd$
$z_t \leftarrow \phi(g_t)$ ▷ Transform according to the desired moment order, e.g. $\phi(u) = u$ for the first-order momentum, and $\phi(u) = u^2$ for the second-order momentum
$w_t \leftarrow (\nu + d)\left(\nu + \sum_j^d \frac{(z_t^j - m_{t-1}^j)^2}{(\sigma_{t-1}^j)^2 + \epsilon}\right)^{-1}$
$\beta_w \leftarrow \frac{W_{t-1}}{W_{t-1}+w_t}$
$\sigma_t^2 \leftarrow \beta\sigma_{t-1}^2 + w_t(1-\beta)(z_t - m_{t-1})^2$ ▷ Estimate the variance if needed
$m_t \leftarrow \beta_w m_{t-1} + (1-\beta_w)z_t$ ▷ Compute the t-EMA
$W_t \leftarrow \frac{2\beta-1}{\beta} W_{t-1} + w_t$
Output: Updated weight sum: W_t
Output: Updated variance: σ_t^2 ▷ Only if it is estimated
Output: Updated t-EMA: m_t

C. Formulation

To replace the EMA with the student-t distribution, a new hyperparameter, the degrees of freedom of the student-t distribution ν, is introduced to control the robustness.

We can derive the incremental update law of the first-order momentum μ for the student-t distribution using a maximum log-likelihood estimator. Given n i.i.d. random samples $x_1, \ldots, x_n$ of dimension d sampled from a multivariate student-t distribution p_t with mean μ, covariance Σ and degrees of freedom ν, the log-likelihood function of p_t is expressed as

$$\begin{aligned}\log p_t = \Bigg\{ & n \log \Gamma\left(\frac{\nu+d}{2}\right) - n \log \Gamma\left(\frac{\nu}{2}\right) \\ & - \frac{n\nu}{2}\log(\nu) - \frac{nd}{2}\log(\pi) - \frac{n}{2}\log(|\Sigma|) \\ & - \left(\frac{\nu+d}{2}\right)\sum_{i=1}^{n}\log(\nu + D_i)\Bigg\} \end{aligned} \tag{14}$$

where D_i is defined as $D_i = (x_i - \mu)^\top \Sigma^{-1}(x_i - \mu)$. Taking the gradient with respect to μ and setting it equal to 0 gives us

$$\begin{aligned}\frac{\partial \log p_t}{\partial \mu} &= \sum_{i=1}^{n}(\nu+d)\frac{x_i - \mu}{\nu + D_i} \\ &= \sum_{i=1}^{n} x_i \frac{\nu+d}{\nu+D_i} - \mu\sum_{i=1}^{n}\frac{\nu+d}{\nu+D_i} \overset{\text{set}}{=} 0. \end{aligned} \tag{15}$$

If we solve this equation for μ, we get the expression of the first-order momentum estimate given n samples

$$\hat{\mu}_n = \frac{\sum_{i=1}^{n} x_i w_i}{W_n} = \frac{\sum_{i=1}^{n-1} x_i w_i + x_n w_n}{W_{n-1} + w_n}$$

Algorithm 2 t-Adam: The Adam Algorithm Extended by the t-Momentum; in Typical Setting, $\beta_1 = 0.9$ Is Smaller Than $\beta_2 = 0.999$; a Good Default Value for the Degrees of Freedom Is Empirically Found to Be $\nu = d$ (i.e. $k = 1$), Where $d = dim[\nabla_\theta J(\theta)]$.

Require: α: Learning rate
Require: $\beta_1, \beta_2 \in [0,1)$: Exponential decay rates
Require: ϵ: Small term added to the denominator
Require: ν: Degrees of freedom
Require: $J_\theta(x_t)$: Objective function with parameters θ
Require: θ_1: Initial parameters
$m_0 \leftarrow 0$, $v_0 \leftarrow 0$, $t \leftarrow 0$
$W_0 \leftarrow \frac{\beta_1}{1-\beta_1}$
while θ_t not converged or $t < T_{\max}$ **do**
  $t \leftarrow t + 1$
  $x_t \leftarrow x_{t+1}$
  $g_t \leftarrow \nabla_\theta J_{\theta_t}(x_t)$
  $w_t \leftarrow (\nu + d)\left(\nu + \sum_j^d \frac{(g_t^j - m_{t-1}^j)^2}{v_{t-1}^j + \epsilon}\right)^{-1}$ ▷ We use Adam's second moment v_t as the variance
  $\beta_w \leftarrow \frac{W_{t-1}}{W_{t-1}+w_t}$
  $m_t \leftarrow \beta_w m_{t-1} + (1-\beta_w)g_t$
  $W_t \leftarrow \frac{2\beta_1 - 1}{\beta_1} W_{t-1} + w_t$
  $v_t \leftarrow \beta_2\, v_{t-1} + (1-\beta_2){g_t}^2$
  $\hat{m}_t \leftarrow m_t/(1-\beta_1^t)$, $\hat{v}_t \leftarrow v_t/(1-\beta_2^t)$
  $\theta_{t+1} \leftarrow \theta_t - \alpha\frac{\hat{m}_t}{(\sqrt{\hat{v}_t} + \epsilon)}$
end while
return θ_t

$$= \frac{W_{n-1}}{W_{n-1}+w_n}\hat{\mu}_{n-1} + \frac{w_n}{W_{n-1}+w_n}x_n \tag{16}$$

where $w_i = (\nu + d)/(\nu + D_i)$ and $W_n = \sum_{i=1}^{n} w_i$.

By assuming a diagonal distribution and fixing the number of samples (i.e., by decaying W_n with a decay rate ψ), we can derive the (17) as the t-momentum

$$m_t = \beta_w m_{t-1} + (1-\beta_w)g_t \tag{17}$$

where

$$\beta_w = \frac{W_{t-1}}{W_{t-1}+w_t} \tag{18}$$

$$w_t = \frac{\nu+d}{\nu+D_t} \tag{19}$$

$$W_t = \psi W_{t-1} + w_t \tag{20}$$

$$D_t = \sum_j^d \frac{(g_t^j - m_{t-1}^j)^2}{(\sigma_{t-1}^j)^2 + \epsilon} \tag{21}$$

where ϵ is a small number introduced to avoid division by 0 and $(\sigma^j)^2$ are the diagonal elements of the covariance matrix Σ. In the remaining of this article, we note $\beta_w = (W_{t-1}/W_{t-1} + w_t)$ so that $(w_t/W_{t-1} + w_t) = 1 - \beta_w$. Note that D_t corresponds to the squared Mahalanobis distance between the gradient of the parameter θ^j, g_t^j, and the corresponding previous estimate of the mean, m_{t-1}^j, with respect to the previous variance estimate $(\sigma_{t-1}^j)^2$.

In the implementation of t-Adam (Algorithm 2), due to the high value of β_2 (i.e., $\beta_2 = 0.999$ about 1000 samples) with

respect to β_1 (i.e., $\beta_1 = 0.9$ about ten samples), the second-order momentum is less sensitive to outliers. Therefore, only the first-order momentum in (2) is replaced by the previous rule of (17). This allows us to avoid estimating both the variance of the gradients and the squared gradients, by using the unmodified second momentum estimate coming from (3) as the variance for the first-order momentum, i.e., $\sigma_t^2 = \upsilon_t$. Note that, ultimately, the gradients converge to zero, and therefore, the second momentum would be consistent with the variance of the gradients.

Equations (2)–(5) of the Adam algorithm presents a computational complexity of roughly $13d = \mathcal{O}(d)$ floating point operations (flops). For t-Adam, (2) of Adam is expanded by (17)–(21), with (17) having the same flops as (2). Counting again the number of operations, we obtain $13d+(5d-1)+7 = 18d+6 = \mathcal{O}(d)$ flops. The computational complexity of the t-Adam algorithm therefore remains linear with respect to the gradients' dimension d.

The power of the t-EMA based momentum or, in short, the t-momentum, update rule is two folds: the outliers detection and the robustness control. The details are explained below.

D. Outliers Detection

The adaptive weight w_t plays the role of a filtering parameter to reject ($w_t \simeq 0$) or accept ($w_t \gg 0$) the gradients in the momentum [see (17)]. Again, we can notice that w_t depends on the Mahalanobis distance D_t (and also on ν and d). Hence, outlying gradient values are down-weighted since their Mahalanobis distances are larger than for normal values, $D_t > d$, and their contribution to the momentum update is therefore automatically dampened. On the contrary, the normal gradients are up-weighted ultimately by $1 + d/\nu$ due to zero Mahalanobis distances, although m is kept in that case since $m_{t-1} = g_t$. In short, the t-momentum automatically and continuously reduces only the adverse effects of the outlier gradients.

The t-momentum can be implemented in several versions according to the size of the subsets considered, i.e. according to the dimension value d: 1) if d is one, then the t-momentum is a parameter-wise estimates; 2) if instead d is the number of parameters in subsets (e.g. in layers of deep learning), then we have block-wise estimates; and 3) if d is the total number of all parameters (all subsets considered at once), then the t-momentum is used for whole estimates. Intuitively, if only one component of the gradient is aberrant, the version 1) updates the other parameters normally, even though their corresponding gradients are also more likely to have been computed from an outlier. On the other hand, the version 3) may tend to underestimate that aberrant value since d in w is extremely large and D_t will hardly go beyond d. Hence, we expect that the version 2), the block-wise t-momentum estimates, would have the best balance. To test that hypothesis, we investigate their performances in Section IV-A1.

E. Robustness Control

The student-t distribution has controllable robustness and the nice property of being similar to the normal distribution when the degrees of freedom grows larger. The same feature is left in the t-momentum, as can be seen in (19). Namely, when $\nu \to \infty$, we have

$$\lim_{\nu\to\infty} w_t = \frac{1 + d/\infty}{1 + D_t/\infty} = 1. \tag{22}$$

In this case, the t-momentum loses its robustness to outliers.

To ensure that the t-momentum is an extended version of the standard EMA-based momentum and reduces exactly to it in the limit of $\nu \to \infty$, the decay rule in (20) is designed to fulfill some requirements. Specifically, if $\nu \to \infty$, the decay rate β_w derived from W_{t-1} and w_t in (17) must be consistent with β_1 at any time. Since w_t is constant and equal to 1 as shown in 22, we therefore demand that W_t be defined by

$$W_t = W_0 = \frac{\beta_1}{1-\beta_1} \quad \forall t > 0. \tag{23}$$

To satisfy such a constant W, if the decay rule is expressed as $W_t \leftarrow \psi W_{t-1} + w_t$, then the decay rate ψ in (20) can be derived as follows:

$$\psi = \frac{W_t - w_t}{W_{t-1}} = \frac{2\beta_1 - 1}{\beta_1}. \tag{24}$$

By the above derivation, the t-momentum defined by (17)–(20) is proved to be the extended version of the EMA-based momentum defined by (2).

A different approach, however, would be to simply consider a constant W_t without any decay, i.e. $W_t = W_0 = (\beta_1/1-\beta_1), \forall t > 0$. In that case, although computing and storing W_t can be ignored, it would certainly be less robust than the proposed version, where W_t can become larger than W_0 when w_t is larger than 1 (i.e., g_t is a nonaberrant gradient), thereby making the t-momentum more discriminative and more likely to reduce the effects of outliers. These two versions are compared in Section IV-A1.

In practice, since the dimension of the gradients $d = \dim(g_t)$ can be arbitrarily large in deep learning neural networks, we design the degrees of freedom in terms of its relative importance with respect to d and set it to be a multiple of the gradients' dimension, that is,

$$\nu = kd \quad k > 0. \tag{25}$$

By doing so, we can easily and consistently define the degree of robustness of the algorithm through the scale factor k, without a heavy dependence on the structure of the neural network.

F. Regret Bound and t-Adam's Convergence

Since it would be rather difficult to study the convergence of all optimization methods augmented with the t-momentum, here, we only focus on Adam with the t-momentum, i.e., t-Adam, and investigate its convergence property in terms of the regret. This study (Theorem 1) hints to the fact that if the regret bound of a given baseline optimization algorithm is expressed as a function of the momentum decay parameter β, i.e. $R_T \leq f(\beta)$, then the corresponding t-algorithm (the algorithm extended with the t-momentum) can be expressed

with the same function in terms of the upper bound of the expected value of β_w, i.e. $R_T \leq f(\max \mathbb{E}[\beta_w])$.

Indeed, the convergence of the t-Adam algorithm is assured by the following two theorems, whose proofs can be found in the Appendix.

Theorem 1: Given $\{\theta_t\}_0^T$ and $\{v_t\}_0^T$, the sequences obtained from the t-Adam algorithm, $\alpha_t = (\alpha/\sqrt{t})$, $\beta_{1t} = \beta_w$, $\mathbb{E}[\beta_w] \leq \bar{\beta}_w < 1$ and $\gamma = (\bar{\beta}_w/(\beta_2)^{1/2}) < 1$. If F has a bounded diameter D_∞, and if $\|g_t\|_\infty = \|\nabla_{\theta_t} J(\theta_t)\|_\infty \leq G_\infty$ for all $t \in [T]$ and $\theta_t \in F$, then for θ_t generated using t-Adam (with the AMSGrad [8] scheme), we have the following upper bound on the regret:

$$R_T \leq \frac{D_\infty^2}{2\alpha_T(1-\bar{\beta}_w)} \sum_{i=1}^{d} \hat{v}_{T,i}^{1/2} + \frac{D_\infty^2}{(1-\bar{\beta}_w)^2} \sum_{t=1}^{T} \sum_{i=1}^{d} \frac{\beta_{1t} \hat{v}_{t,i}^{1/2}}{\alpha_t} + \frac{\alpha\sqrt{1+\log T}}{(1-\bar{\beta}_w)^2(1-\gamma)\sqrt{(1-\beta_2)}} \sum_{i=1}^{d} \|g_{1:T,i}\|_2 \quad (26)$$

where $\hat{v}_t$ is the bias-corrected second raw moment estimate, i.e. $\hat{v}_t = v_t/(1-\beta_2^t)$.

Corollary 1: Given the regret bound $R(T)$ as defined in Theorem 1, t-Adam achieves the following guarantee, for all $T \geq 1$:

$$\mathbb{E}_{t=1,..,T}\big[J_t(\theta_t) - J_t(\theta^*)\big] = \frac{R(T)}{T} = \mathcal{O}\left(\frac{1}{\sqrt{T}}\right) \quad (27)$$

and in terms of iteration complexity, the maximum number of iterations (or steps) t required to achieve a fixed expected optimization error of at most ϵ, for convex objective functions J_t, is of order

$$\mathcal{O}\left(\frac{1}{\sqrt{\epsilon}}\right). \quad (28)$$

Theorem 2: With respect to the central limit theorem, let us assume that the gradients g_t ultimately follow an asymptotic normal distribution $g_t \in \mathbb{R}^d \sim \mathcal{N}$ as $t \to \infty$. Then, the expected value of the adaptive decay parameter $\beta_w = (W_{t-1}/W_{t-1} + w_t)$ is constrained, for $\beta_1 < 1$, by the following relation:

$$\mathbb{E}[\beta_w] \leq \beta_1. \quad (29)$$

We can see that the difference between the upper bound of t-Adam and Adam lies in the value of $\bar{\beta}_w$, which corresponds to the upper bound of the expected value of the adaptive exponential decay parameter $\beta_w = (W_{t-1}/W_{t-1} + w_t)$. Theorem 2 tells us that, if the gradients are normally distributed, this value is bounded above by β_1, so that we can recover the same upper bound for t-Adam and Adam. Even if the aberrant gradients are given from outliers (i.e., $\bar{\beta}_w \neq \beta_1$), t-Adam still has the theoretical upper bound on the regret derived in theorem 1, since $\beta_w \in (0, 1)$. Note that, if we know the exact value of the expected value, a more precise upper bound for the regret can be obtained.

Corollary 1 is obtained by using $\|g_{1:T,i}\|_2 \leq G_\infty\sqrt{T}$ and it shows that t-Adam preserves the same order of the iteration complexity of the Adam optimizer.

IV. Experiments

To assess the robustness of the t-momentum against noisy and/or heavy-tailed data, we conducted three types of experiments spanning the main machine learning frameworks, i.e., supervised learning (regression and classification) and RL. We compare t-Adam mainly with Adam, but also with another robust gradient descent algorithm, such as RoAdam [21], and also present the comparison between some popular or recent optimization methods (in majority, variants of Adam, i.e. AdaBound [18], AdamW [31], DiffGrad [32], RAdam [33], PAdam [34], Yogi [35], and LaProp [36]) and their t-versions.[1] Note that we are not exhaustive in our selection and that the t-momentum can be integrated in other momentum-based optimization methods.

A. Robust Supervised Learning

It has been shown [37] that training standard supervised learning algorithms with noisy data resulted in bad performance and accuracy of the resulting models. In real robotic tasks, for example, it is often unrealistic to assume that the true state is completely observable and noise-free, and perfect supervised signals are difficult to obtain. In the following experiments, t-Adam reveals to be useful in increasing the accuracy of the models, even when facing noisy inputs.

1) Robust Regression:

a) Experimental settings: We define a ground truth function: a simple sinusoidal function $f(x) = \sin(2\pi x)$ or a more complex function $f(x) = x^2 + ln(x+1) + \sin(2\pi x) * \cos(2\pi x)$. The observations are sampled from the true function with noise, ζ, as follows:

$$y = f(x) + \zeta \quad (30)$$

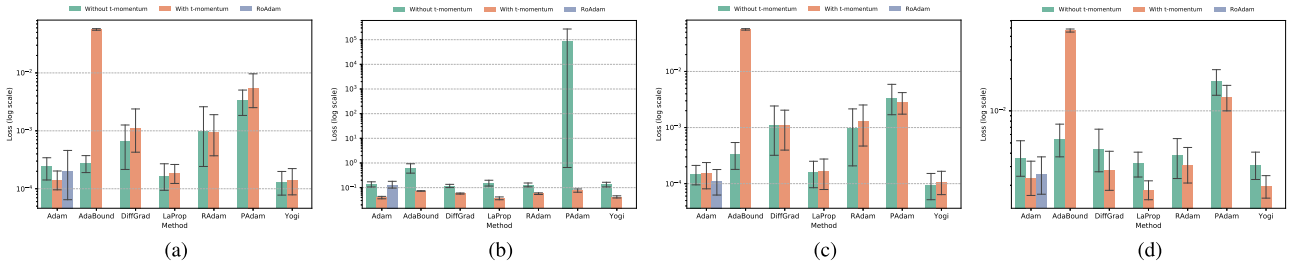
$$\zeta \sim \mathrm{St}(\nu_\zeta, 0, \lambda_\zeta)\mathrm{Bern}\left(\frac{p}{100}\right), \quad p = 0, 10, 20, \ldots, 100 \quad (31)$$

where $\mathrm{St}(\nu_\zeta, 0, \lambda_\zeta)$ designates a student-t distribution with degrees of freedom ν_ζ, 0 location, and scale λ_ζ. $\mathrm{Bern}(p/100)$ is a Bernoulli distribution with the probability p as its parameter. A thousand samples for the simple sinusoidal function and 4000 samples for the more complex function are sampled as observations.

A fully connected neural network with five linear layers, each composed of 50 neurons, approximates it from scattered observations y. The ReLU activation function [38] is used for all the hidden layers, while the loss function for the network is the mean squared error (MSE).

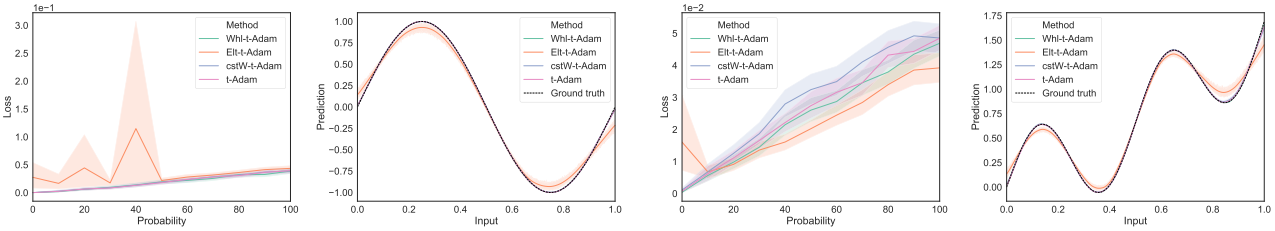
b) Experimental results: The results of the final training loss values for a noise presence probability p of 0% and 100%, on the regression task, are depicted in Fig. 2. Note that 50 trials with different random seeds are conducted for each noise probability p and each optimization method, and that the logarithmic scale is used for the loss axis. As it can be seen, the t-momentum improves the robustness of the baseline algorithms and is able to reach almost the same lowest point loss even when the noise probability is at its maximum [Fig. 2(b) and (d)]. Even though the loss of the t-AdaBound method was higher compared to AdaBound for the noise

[1] All codes are available on https://github.com/Mahoumaru/TAdam



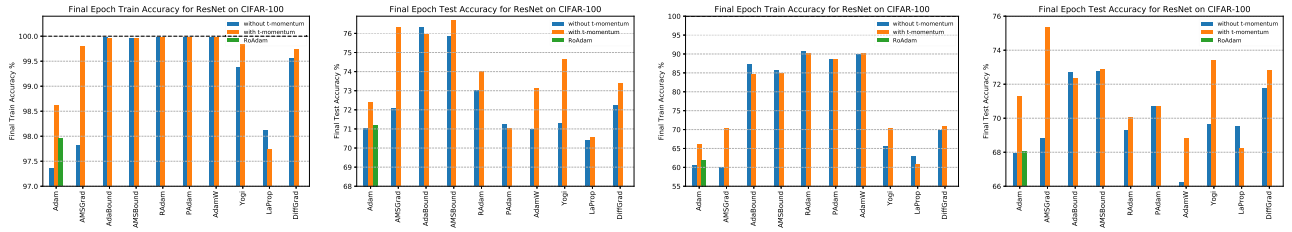
(a) (b) (c) (d)

Fig. 2. Results of the regression task (Sine function): the figures illustrate the loss for different noise settings with respect to the extreme probabilities $p \in \{0.0, 100.0\}$; in all settings, the t-momentum improved the robustness of the baseline algorithm and for almost all methods, it also improved the stability across multiple trials (with a thinner standard deviation). (a) $(\nu_\zeta, \lambda_\zeta) = (1.0, 0.05)$ with Probability $p = 0.0$. (b) $(\nu_\zeta, \lambda_\zeta) = (1.0, 0.05)$ with Probability $p = 100.0$. (c) $(\nu_\zeta, \lambda_\zeta) = (2.0, 0.03)$ with Probability $p = 0.0$. (d) $(\nu_\zeta, \lambda_\zeta) = (2.0, 0.03)$ with Probability $p = 100.0$.



(a) (b) (c) (d)

Fig. 3. Comparison between different versions of t-Adam: 1) the parameter or element-wise version (Elt-t-Adam $d = 1$); 2) the block-wise version (t-Adam $d = \dim(g)$); and 3) the whole estimate version (Whl-t-Adam). The version where W_t is kept constant is called cstW-t-Adam. (b) and (d) Form of the ground truth function with the predictions of the models trained with noise probability $p = 0.0\%$ (no noise). (a) $(\nu_\zeta, \lambda_\zeta) = (1.0, 0.05)$. (b) $(\nu_\zeta, \lambda_\zeta) = (1.0, 0.05)$. (c) $(\nu_\zeta, \lambda_\zeta) = (1.0, 0.05)$. (d) $(\nu_\zeta, \lambda_\zeta) = (1.0, 0.05)$.



(a) (b) (c) (d)

Fig. 4. Training and test accuracy (noise-free and noise-included) for ResNet-34 on CIFAR-100. (a) Noise-free training accuracy. (b) Noise-free test accuracy. (c) Noisy data training accuracy. (d) Noisy data test accuracy.

distribution $(\nu_\zeta, \lambda_\zeta) = (2.0, 0.03)$, we can see that the loss value is barely affected by the noise in the data from Fig. 2(c) and (d). The high value for the loss can however be explained by the fact that the baseline algorithms' hyperparameters were not optimized and the default values were employed both for the methods with and without the t-momentum. Note that only the robustness was of interest in the experiments, and it was important to have the same parameters for each one of the baseline algorithms and their corresponding t-momentum version. It is however obvious that the optimal hyperparameter values, such as the optimal learning rate, will be different between the baseline optimizer and its t-momentum-based version.

We also compare, in Fig. 3, different versions of Algorithm 2 depending on 1) the value of d, Elt-t-Adam and Whl-t-Adam, as suggested in the Section III-D; and 2) on whether or not the quantity W_t is kept constant, cstW-t-Adam: $W_t = \beta_1/(1 - \beta_1) = \text{const.}$ (see the Section III-E). The vanilla t-Adam is taken to be the block-wise algorithm where the dimension d is the number of parameters in each layer of the neural network (the weights and biases from the same layer are also treated as separate subsets). In the Elt-t-Adam version, however, w_t and W_t are calculated for each component of the gradient vectors, and therefore, each component has its own weight β_w. In complete contrast, in Whl-t-Adam, the dimension d is set to be equal to the number of all the parameters in the model, and the scalar weight w_t, along with its weighted sum W_t, yields one value computed for all of them.

For the parameter-wise version (Elt-t-Adam), we can see that it can perform better than the vanilla t-Adam depending on the conditions [Fig. 3(c)], but can also be unstable. Elt-t-Adam is similar to a univariate t-distribution applied to each component with degrees of freedom $\nu = d = 1$. Even though it can be viewed as a Cauchy distribution that normally has no defined moment, it can also be viewed as a very robust version

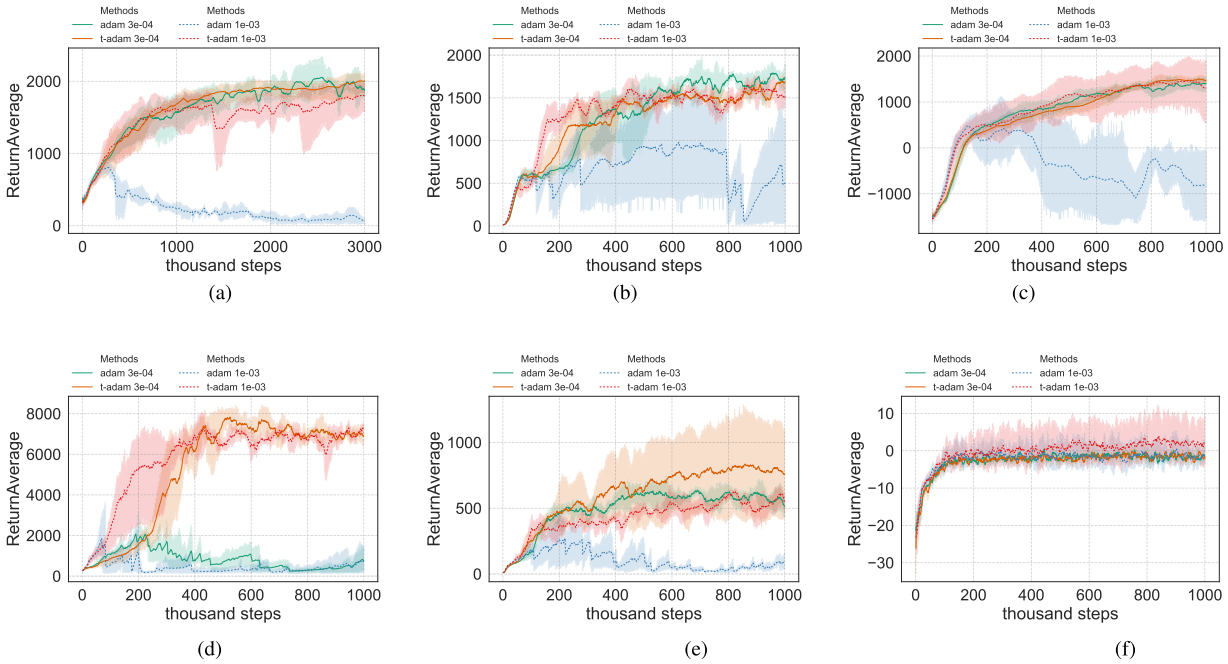


Fig. 5. Training curves for PPO agents with different learning rates (3e-4 or 1e-3) and t-Adam or Adam; Adam with large learning rate (1e-3) often failed to learn tasks due to wrong updates; in contrast, t-Adam basically succeeded in acquiring tasks thanks to conservative updates. (a) Pybullet Ant-v0. (b) Pybullet Hopper-v0. (c) Pybullet HalfCheetah-v0. (d) Pybullet InvertedDoublePendulum-v0. (e) Pybullet Walker2D-v0. (f) Pybullet Reacher-v0.

of the t-distribution. This can explain why for lower noise probability p, this version failed to output a small loss, going furthermore to be unstable, producing imperfect models [as shown in Fig. 3(b) and (d)], while for higher noise probability values, it can perform similarly or slightly better than the vanilla t-Adam.

On the other hand, the whole estimate version (Whl-t-Adam) performs very well. Since noise in observations or target values simultaneously affects the gradient of every parameters, regardless of the layer, a general weight β_w assures that such gradients are uniformly accepted or rejected in the exponential moving average across the whole model. We argue that this uniform update contributes to making the network more stable and constitutes the main reason for its good performance compared to the vanilla algorithm. However, this t-Adam version requires a more complicated implementation (to flatten and concatenate all the parameters once) and computational cost than the blockwise t-Adam.

Finally, the results of cstW-t-Adam, which keeps W_t constant, instead of updating it through the weighted sum of the weights w_t are analyzed. Indeed, this version does not, theoretically, change the upper bound of β_w and therefore constitutes a legitimate candidate that could produce a more computationally efficient algorithm. In the results, however, we can see that this algorithm is the less robust one of all four proposed versions. This can be explained by the fact that the expected value of w_t is larger than 1 (see the Appendix), which leads to $W_t > W_0$ in the case of decaying W_t, and further reduces the adverse effects of noise.

In summary, the variants of t-Adam possess their own drawbacks, and therefore, the vanilla version, i.e., the one using block (or subset)-wise updates with decaying W_t, provides the best overall performance. That is, it is able to perform well in all cases while presenting a good overall computational cost. The block (or subset)-wise t-momentum is, therefore, the one we use in the subsequent experiments.

2) Robust Classification:

a) Experimental settings: Here, we use the same experimental settings described in [18] and compare, Adam, AMSGrad, Adabound, RAdam, PAdam, AdamW, Yogi, and LaProp to their t-momentum-used versions, on an image classification task on the standard CIFAR-100 data sets. RoAdam is also included to compare its robustness with respect to Adam and t-Adam.

The architecture of the convolutional network involved in the described experiments is the ResNet-34 [39]. A fixed budget of 200 epochs are used throughout the training, and the learning rates are reduced by 10 after 150 epochs.

The optimizers are launched with following hyperparameter values shown in Table I (where the γ refers to the bound functions timestep weights $(1-\beta)$ of the AdaBound methods [18]). All t-momentum-used algorithms use the default degrees of freedom {degrees of freedom = dimension of the gradients}, i.e. $k = 1$. The third beta value of RoAdam is also set to 0.999, and the default values for hyperparameters different from the basic ones of Adam and AdaBound are used (for PAdam, the partial $P = 1/4$, and AdamW, the warmup $= 0$). Note that this experiment is not designed to compare the optimizers among them, but rather to compare each method with its t-momentum augmented version.

b) Experimental results: We first launched a simulation using directly the unmodified and uncorrupted data set. The results for that simulation are found in Fig. 4(a) and (b) for the CIFAR-100. We can see that the t-momentum improves

TABLE I
SETTINGS FOR THE CLASSIFICATION EXPERIMENTS

Optimizers	α	β_1	β_2	Final α	γ
Adam, AMSGrad, t-Adam, t-AMSGrad RoAdam RAdam, PAdam t-RAdam, t-PAdam Yogi, LaProp t-Yogi, t-LaProp DiffGrad, AdamW t-DiffGrad, t-AdamW	0.001	0.99	0.999	N.A.	N.A.
AdaBound, AMSBound t-AdaBound, t-AMSBound	0.001	0.9	0.999	0.1	0.001

the generalization ability of almost all base methods, and t-AMSGrad is even able to reach the same level of generalization as the AdaBound methods. This result points out the fact that t-AMSGrad builds on the combined improvement of the first moment (t-Adam) and second moment (AMSGrad) in order to provide a more stable algorithm with a standout performance.

Next, we applied, with a probability of 25%, a color jittering effect on the CIFAR-100 training data set and replaced 20% of the original training data points with fake ones, in order to test the ability of the optimizers to extract the most useful information from corrupted data sets. The results can be seen in Fig. 4(c) and (d).

The benefits of the t-momentum are again highlighted. Even though the value of β_1 is larger (0.99 instead of default 0.9), Adam and most of the other base algorithms remain sensitive to outliers, and in most of the latest optimizers, the t-momentum improved the robustness. In particular, we can see that t-Adam, t-AMSGrad, t-AdamW, and t-Yogi highly improve the performance of the base algorithms against aberrant values in corrupted data sets, and achieve similar performance as with the original data set. However, for some of the optimization methods (t-AMSBound, t-RAdam, t-PAdam, and t-Diffgrad), the t-momentum could not keep, against the corrupted data set, the test accuracy from dropping almost by the same amount the accuracy of the base optimizer (with regular EMA momentum) dropped. Despite that, we notice that the accuracy of the t-algorithm remained higher than the base algorithm. Finally, for the last two optimizers, i.e. Adabound and LaProp, the t-momentum failed to improve the accuracy of the baselines.

The result of the Adabound methods can be explained by their fast convergence property. Indeed, as pointed out by Hardt *et al.* [40], *fast training time by itself is sufficient to prevent overfitting*. Because the algorithm prevents itself from overfitting, it is also less sensitive to the presence of outliers in the data set. This allows it to quickly converge to a suboptimal point regardless of the aberrant values. However, fast convergence is not a reliable source of robustness to avoid being affected by corrupted data sets. In fact, compared with the result of t-AMSGrad, we see that a method that takes time to extract general and dominant trends from the data set instead of keeping itself from overfitting is a more efficient strategy.

TABLE II
SETTINGS FOR THE RL EXPERIMENTS

Value loss coef.	1
Entropy loss coef.	0
GAE parameter λ	0.95
Num. Epochs	10
Ratio clipping ϵ	0.2
Horizon T	2048
Minibatch size	64
t-Adam d.o.f. ν	$\dim(g)$

As for LaProp, the method itself, as explained by the authors in the original article [36], is designed so that more importance is given to the momentum, bounding the effect of outliers (mainly large gradients) on the exponential average. The proposition 1 in their article states that the magnitude of the updates (therefore of the momentum) has an upper bound that only depends on β_2, the decay factor of the second moment. According to this, we hypothesize that in order to further improve the robustness of the LaProp method, it is necessary to integrate the t-EMA, not only to the first moment, but also to the second moment. This hypothesis is consistent with the suggestion made by the LaProp authors about varying the value of the decay factor β_2 for noisy or complex tasks, which the t-EMA can automatically perform.

B. Robust Reinforcement Learning

Whether it comes from sensors, from bad estimates during learning, or different feedbacks of different human instructors (e.g., nontechnical users in real-world robotics situations), noisiness is inseparable from robotics RL. Especially for reducing the effects of bad estimates, the latest RL algorithms carefully update the value and policy functions by modifying their optimization targets [41]–[43]. In order to test the robustness properties of t-Adam in RL tasks, we conducted some simulations on six different Pybullet gym environments [44].

1) Experimental Settings: The algorithm employed in this article is the proximal policy optimization (PPO) [41], from the Berkley artificial intelligence research implementation, rlpyt [45], with hyperparameters summarized in Table II. The simulations involved two different learning rates: the widely used and fine-tuned value for Adam on RL, 3×10^{-4}, and the defined default for supervised learning, yet larger value for RL, 1×10^{-3}. Searching for the optimal learning rate is commonly known to be a tedious and serious problem in SGD based algorithms, and high learning rates (particularly the default Adam step value 1×10^{-3}) are usually not used in RL due to the amount of noise coming from the early bootstrapping stage, but also to avoid the agent from reaching an early deterministic policy.

Here, as a remark, no gradient norm clipping was used throughout the simulations, since the property at test is the robustness of the optimizers to aberrant gradient values and their ability to produce good policies. Gradient norm clipping introduces a manually defined heuristic threshold, which depends on the task and various conditions, and moreover, is used for the norm of all gradients larger than its value.

Such a trick would therefore introduce some undesirable bias in the results.

2) Experimental Results: The results, which were gained from four trials with different random seeds on each environment with each condition, are summarized in Fig. 5. As displayed by the results in Fig. 5, a high learning rate causes Adam to suffer from both these problems and makes it unable to converge to a good policy. On the other hand, t-Adam proves to be robust enough to sustain different learning rates and learns the tasks with both given hyperparameter values. Thanks to its careful updates of the agent, t-Adam can still reach a suboptimal policy that may even be better than the one reached with smaller learning rates [see Fig. 5(c) and (f)]. This feature offered by t-Adam not only allows for the use of higher learning rates in order to accelerate the learning process [see Fig. 5(b) and (d)], but also reduces the difficulties related to the tuning of the learning rate since the default learning rate can be directly used.

In addition, as stated in the experimental settings section, no gradient norm clipping was used during the simulations. Without this trick, we can see that Adam fails altogether on the inverted double pendulum task, while t-Adam naturally and automatically ignores or reduces the effect of large gradients, keeping the gradient (momentum) from overshooting during learning and making the gradient norm clipping stratagem unnecessary.

V. Discussion

The above experiments and simulations showed the robustness of our proposition with respect to the existing approaches. Indeed, t-Adam proves to be robust on supervised tasks (see Figs. 2 and 4), and also on RL problems (see Fig. 5). However, we have to discuss its limitations as below.

A. Additional Computational Cost

Our proposal, the t-momentum, requires new computations for deriving the weights (see 17–20). In general, as presented by Luo *et al.* [18], adaptive methods such as Adam usually display generalization abilities that are worse than the non-adaptive optimization methods. This fact is widely attributed to unstable and extreme learning rates, and even AMSGrad [8] was failing to significantly improve the generalization ability of Adam on unseen data, leading to the proposition of methods such as AdaBound [18], which requires heuristic design, and RAdam [33]. However, as shown by the results on the classification tasks, t-AMSGrad proves to be able to achieve the same level of generalization as the AdaBound methods, while also displaying faster progress than Adam and AMSGrad during the early stage of training. Therefore, we conclude that the additional computational cost for the t-momentum is undeniable to improve optimizers.

B. Necessity of Variance

One condition for the integration of our method to existing algorithms is the requirement of a variance estimate. In Adam and its variants, the second moment is already available and provided by the method itself, but this is not always the case. In practice, if the variance is not estimated, the easiest way is to set it to a constant value (for example, 1). But of course, this still yields the danger to produce bad estimates if the true variance is smaller (it may not be enough robust) or greater (in that case, it may be too robust). Hence, our t-momentum works better when the variance is estimated along with the first moment, although that requires further computations. The variance estimator can, similar to the mean, be obtained using the log maximum likelihood of the student-t distribution.

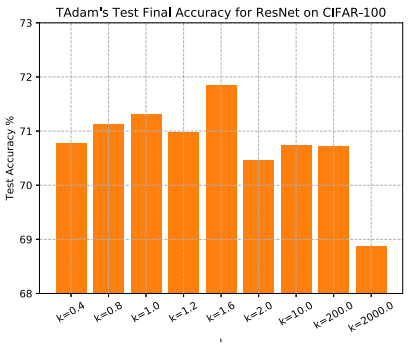


Fig. 6. Comparison between different values of the degrees of freedom $\nu = k * \dim(g)$: ν seems to be unimodal, namely, it has an optimal value (i.e., not too small and too large).

C. Optimal Degrees of Freedom

We must also address the choice of the degrees of freedom parameter, which controls the robustness of the t-momentum. Fig. 6 shows the effect of different degrees of freedom $\nu = k * \dim(g)$ on the classification task. We can see that for this particular case, t-Adam performs well for a wide range of values, as long as the factor k is not too large (which would bring back the t-Adam to Adam). The unimodality of the bar chart also reveals an interesting trend. Indeed, it shows that when the degrees of freedom ν are too small, the algorithm is too robust (i.e. conservative) preventing efficient updates, and when it is too large, the algorithm becomes too sensitive to outliers. During our experiments, we found that setting the degrees of freedom to be equal to the dimension size d (i.e. $k = 1$) worked well in most cases, so we propose it as a default value. However, this result highlights the fact that the optimal value is task-dependent, and therefore, ν should be optimized according to the presence or absence of outliers. Currently, a typical hyperparameter optimization method can be employed to tune the degrees of freedom. For example, a grid search can be applied to k in the subset $[0.4, 4.] \cup \{10, \infty\}$ where the ∞ case corresponds to simply using the regular momentum (with fixed β_1). Alternatively, if the data set is available before training, a Student-t distribution can be used to try to fit it (for example, in python, the scipy library provides a fit function) and in this case, the estimated data set degrees of freedom, which will depend on whether the data set contains many or few outliers, can be employed.

VI. Conclusion

In this article, we proposed the t-momentum, a new estimate of the first-order momentum of gradients for SGD, inspired by

the student-t distribution. It makes the Adam and other EMA-based algorithms much more robust and provides a way to produce stable and efficient machine learning applications. t-Adam based on this robust estimate is specifically described as an extended version of Adam. We verified that t-Adam outperformed Adam in terms of robustness on supervised learning (regression and classification) tasks, and RL tasks. In addition, the other SGD methods with the t-momentum also showed the robustness in the classification task.

In this work, the t-momentum used a fixed degree of freedom ν which is based on the dimension of the gradients and therefore has fixed robustness. As mentioned in the discussion, a straightforward improvement is therefore to design a mechanism that automatically updates the parameter ν during the learning process, according to the presence or absence of outliers. Furthermore, the second moment in Adam and in its variants can also be modified with a t-EMA to both reduce the variance of the second moment and improve its robustness. It would be interesting, particularly when integrated into the LaProp method, to analyze the performance of such an approach. Finally, since our proposal is potentially suitable for robotics applications, it will be applied for robot learning, like the imitation of human demonstrations.

APPENDIX A
PROOFS OF UPPER BOUND ON REGRET

In the following proofs, the following notation rule is used: $\langle x, y\rangle$ corresponds to the inner product between the vectors x and y, $\|x\|$ is the euclidean norm, while $\|x\|_p$ refers to the p-norm (in particular, $\|X_i\|_2$ will be used to express the l_2-norm (euclidean) of the i^{th} row of the matrix X).

A. Proof of Theorem 1

Proof: First, we start by noticing that the basic bound of the regret from the convergence proof by Reddi *et al.* [8] also holds for t-Adam, that is,

$$R_T = \sum_{t=1}^{T} J_t(\theta_t) - J_t(\theta^*) \leq \sum_{t=1}^{T} \langle g_t, (\theta_t - \theta^*)\rangle$$
$$\langle g_t, (\theta_t - \theta^*)\rangle \leq R_{1t} + R_{2t} + R'_{3t} \quad (32)$$

where

$$\begin{aligned} R_{1t} &= \frac{\|\hat{V}_t^{1/4}(\theta_t - \theta^*)\|^2 - \|\hat{V}_t^{1/4}(\theta_{t+1} - \theta^*)\|^2}{2\alpha_t(1-\beta_{1t})} \\ R_{2t} &= \frac{\alpha_t \|\hat{V}_t^{-1/4} m_t\|^2}{2(1-\beta_{1t})} \\ R'_{3t} &= -\frac{\beta_{1t}\langle m_{t-1}, \theta_t - \theta^*\rangle}{1-\beta_{1t}} \\ &\leq \frac{\beta_{1t}\langle m_{t-1}, \theta_t - \theta^*\rangle}{1-\beta_{1t}} \\ &\leq \frac{\beta_{1t}\alpha_t \|\hat{V}_t^{-1/4} m_{t-1}\|^2}{2(1-\beta_{1t})} + \frac{\beta_{1t}\|\hat{V}_t^{1/4}(\theta_t - \theta^*)\|^2}{2\alpha_t(1-\beta_{1t})} \\ &\leq R_{2t} + \frac{\beta_{1t}\|\hat{V}_t^{1/4}(\theta_t - \theta^*)\|^2}{2\alpha_t(1-\beta_{1t})} \\ &= R_{2t} + R_{3t}. \end{aligned}$$

So that

$$R_T \leq \sum_{t=1}^{T}(R_{1t} + 2R_{2t} + R_{3t}). \quad (33)$$

However, to further refine this upper bound, we need to redefine the *Lemma 2* used in the proof of Reddi *et al.* [8], since $\beta_{1t} = \beta_w = (W_{t-1}/W_{t-1} + w_t)$ does not satisfy $\beta_w \leq \beta_1$ anymore for all time step t. For this purpose, we give the explicit upper bound of the expected value of β_w, $\mathbb{E}[\beta_w] \leq \bar{\beta}_w < 1$. Note that $\bar{\beta}_w < 1$ is obvious since $\beta_w \in (0, 1)$. Now, following the same process as Reddi *et al.* [8], defines a similar expression to their *Lemma 2* in the case of t-Adam. Indeed, we can write that

$$\sum_{t=1}^{T} \alpha_t \|\hat{V}_t^{-1/4} m_t\|^2 = \sum_{t=1}^{T-1} \alpha_t \|\hat{V}_t^{-1/4} m_t\|^2 + C$$

where

$$\begin{aligned} C &= \alpha_T \sum_{i=1}^{d} \frac{m_{T,i}^2}{\sqrt{\hat{v}_{T,i}}} \leq \alpha \sum_{i=1}^{d} C_i \\ C_i &\leq \frac{\left\{\sum_{j=1}^{T}\prod_{k=1}^{T-j}(1-\beta_{1(T-k+1)})\beta_{1(T-k+1)} g_{j,i}\right\}^2}{\sqrt{T(1-\beta_2)\sum_{j=1}^{T}\beta_2^{T-j}\, g_{j,i}^2}} \\ &\leq \frac{\left(\sum_{j=1}^{T}\prod_{k=1}^{T-j}\beta_{1(T-k+1)}\right)\left(\sum_{j=1}^{T}\prod_{k=1}^{T-j}\beta_{1(T-k+1)} g_{j,i}^2\right)}{\sqrt{T(1-\beta_2)\sum_{j=1}^{T}\beta_2^{T-j}\, g_{j,i}^2}} \\ &\leq \frac{\left(\sum_{j=1}^{T}\bar{\beta}_w^{T-j}\right)\left(\sum_{j=1}^{T}\bar{\beta}_w^{T-j} g_{j,i}^2\right)}{\sqrt{T(1-\beta_2)\sum_{j=1}^{T}\beta_2^{T-j}\, g_{j,i}^2}} = \bar{C}_i. \end{aligned}$$

Back to C

$$\begin{aligned} C &\leq \alpha \sum_{i=1}^{d} \bar{C}_i \\ &\leq \frac{(1-\bar{\beta}_w^T)\alpha}{(1-\bar{\beta}_w)\sqrt{T(1-\beta_2)}} \sum_{i=1}^{d}\sum_{j=1}^{T} \frac{\bar{\beta}_w^{T-j} g_{j,i}^2}{\sqrt{\beta_2^{T-j}\, g_{j,i}^2}} \\ &\leq \frac{(1-\bar{\beta}_w^T)\alpha}{(1-\bar{\beta}_w)\sqrt{T(1-\beta_2)}} \sum_{i=1}^{d}\sum_{j=1}^{T} \left(\frac{\bar{\beta}_w}{\sqrt{\beta_2}}\right)^{T-j} |g_{j,i}|. \end{aligned}$$

Since $\bar{\beta}_w < 1$, we can further bound the previous inequality

$$C \leq \frac{\alpha}{(1-\bar{\beta}_w)\sqrt{T(1-\beta_2)}} \sum_{i=1}^{d}\sum_{j=1}^{T} \gamma^{T-j} |g_{j,i}|$$

where $\gamma = \bar{\beta}_w/(\beta_2)^{1/2}$. It is worth noting that this is also less than 1, just as the ratio $\beta_1/(\beta_2)^{1/2}$.

By using a similar upper bound for all time steps, we can write that

$$\sum_{t=1}^{T} \alpha_t \|\hat{V}_t^{-1/4} m_t\|^2 \leq \sum_{t=1}^{T} \frac{\alpha \sum_{i=1}^{d}\sum_{j=1}^{t} \gamma^{t-j} |g_{j,i}|}{(1-\bar{\beta}_w)\sqrt{t(1-\beta_2)}}.$$

Then, following the same process as in Reddi *et al.* [8], we get the equivalent expression of the *Lemma 2* in the case of t-Adam

$$\sum_{t=1}^{T} \alpha_t \|\hat{V}_t^{-1/4} m_t\|^2 \leq \frac{\alpha\sqrt{1+\log T}\sum_{i=1}^{d}\|g_{1:T,i}\|_2}{(1-\bar{\beta}_w)(1-\gamma)\sqrt{(1-\beta_2)}}. \quad (34)$$

Based on this new lemma, the remaining steps are completely identical to the proof of Reddi *et al.* [8]

$$\begin{aligned} R_T &\leq \sum_{t=1}^{T}(R_{1t}+2R_{2t}+R_{3t}) \\ &= \sum_{t=1}^{T}(R_{1t}+R_{3t}) + \sum_{t=1}^{T}\frac{\alpha_t\|\hat{V}_t^{-1/4}m_t\|^2}{1-\beta_{1t}} \\ &\leq \sum_{t=1}^{T}(R_{1t}+R_{3t}) + \frac{\sum_{t=1}^{T}\alpha_t\|\hat{V}_t^{-1/4}m_t\|^2}{1-\bar{\beta}_w} \\ &\leq \sum_{t=1}^{T}(R_{1t}+R_{3t}) + \frac{(\alpha\sqrt{1+\log T})\sum_{i=1}^{d}\|g_{1:T,i}\|_2}{(1-\bar{\beta}_w)^2(1-\gamma)\sqrt{(1-\beta_2)}} \\ &= \sum_{t=1}^{T}(R_{1t}+R_{3t}) + R_2. \end{aligned} \quad (35)$$

With $K = (2(1-\bar{\beta}_w))^{-1}$, we can derive

$$\begin{aligned} \sum_{t=1}^{T} R_{1t} &\leq K\sum_{t=1}^{T}\frac{\|\hat{V}_t^{1/4}(\theta_t-\theta^*)\|^2 - \|\hat{V}_t^{1/4}(\theta_{t+1}-\theta^*)\|^2}{\alpha_t} \\ &\leq K\frac{\|\hat{V}_1^{1/4}(\theta_1-\theta^*)\|^2}{\alpha_1} \\ &\quad + K\sum_{t=2}^{T}\left(\frac{\|\hat{V}_t^{1/4}(\theta_t-\theta^*)\|^2}{\alpha_t} - \frac{\|\hat{V}_{t-1}^{1/4}(\theta_t-\theta^*)\|^2}{\alpha_{t-1}}\right) \\ &\leq \frac{\sum_{i=1}^{d}\sqrt{\hat{v}_{1,i}}(\theta_{1,i}-\theta_i^*)}{2\alpha_1(1-\bar{\beta}_w)} \\ &\quad + \frac{\sum_{t=2}^{T}\sum_{i=1}^{d}(\theta_{t,i}-\theta_i^*)^2\left(\sqrt{\hat{v}_{t,i}}\alpha_t^{-1} - \sqrt{\hat{v}_{t-1,i}}\alpha_{t-1}^{-1}\right)}{2(1-\bar{\beta}_w)} \\ &\leq \frac{D_\infty^2\sum_{i=1}^{d}\sqrt{\hat{v}_{1,i}}}{2\alpha_1(1-\bar{\beta}_w)} \\ &\quad + \frac{D_\infty^2\sum_{t=2}^{T}\sum_{i=1}^{d}\left(\sqrt{\hat{v}_{t,i}}\alpha_t^{-1} - \sqrt{\hat{v}_{t-1,i}}\alpha_{t-1}^{-1}\right)}{2(1-\bar{\beta}_w)}. \end{aligned}$$

Using a simple telescopic sum on the right side of the inequality, we get

$$\sum_{t=1}^{T} R_{1t} \leq \frac{D_\infty^2\sum_{i=1}^{d}\sqrt{\hat{v}_{T,i}}}{2\alpha_T(1-\bar{\beta}_w)} = R_1. \quad (36)$$

Next, we have

$$\begin{aligned} \sum_{t=1}^{T} R_{3t} &\leq K\sum_{t=1}^{T}\frac{\beta_{1t}\|\hat{V}_t^{1/4}(\theta_t-\theta^*)\|^2}{\alpha_t} \\ &\leq \sum_{t=1}^{T}\frac{\beta_{1t}\|\hat{V}_t^{1/4}(\theta_t-\theta^*)\|^2}{\alpha_t(1-\bar{\beta}_w)^2} \\ &\leq \frac{\sum_{t=1}^{T}\sum_{i=1}^{d}\beta_{1t}(\theta_{t,i}-\theta_i^*)\sqrt{\hat{v}_{t,i}}\alpha_t^{-1}}{(1-\bar{\beta}_w)^2} \\ &\leq \frac{D_\infty^2\sum_{t=1}^{T}\sum_{i=1}^{d}\beta_{1t}\sqrt{\hat{v}_{t,i}}\alpha_t^{-1}}{(1-\bar{\beta}_w)^2}. \end{aligned} \quad (37)$$

By bringing together the relations expressed by (36), (35), and (37), the final regret bound of t-Adam is given by

$$\begin{aligned} R_T &\leq R_1 + R_2 + R_3 \\ &= \frac{D_\infty^2}{2\alpha_T(1-\bar{\beta}_w)}\sum_{i=1}^{d}\hat{v}_{T,i}^{1/2} + \frac{D_\infty^2}{(1-\bar{\beta}_w)^2}\sum_{t=1}^{T}\sum_{i=1}^{d}\frac{\beta_{1t}\hat{v}_{t,i}^{1/2}}{\alpha_t} \\ &\quad + \frac{\alpha\sqrt{1+\log T}}{(1-\bar{\beta}_w)^2(1-\gamma)\sqrt{(1-\beta_2)}}\sum_{i=1}^{d}\|g_{1:T,i}\|_2. \end{aligned} \quad (38)$$

□

B. Proof of Theorem 2

Proof: With respect to the central limit theorem, let us assume that the gradients g_t ultimately follow an asymptotic normal distribution $g_t \in \mathbb{R}^d \sim \mathcal{N}(\mu, \Sigma)$ as $t \to \infty$. Then we know that the Mahalanobis distance follows the chi-squared distribution with d the degrees of freedom, $(g_t-\mu)^\top\Sigma^{-1}(g_t-\mu) \sim \chi^2(d)$. Applying this to the Mahalanobis distance in t-Adam, we have

$$D_t = \sum_{j}^{d}\frac{(g_{j,t}-m_{j,t-1})^2}{v_{j,t-1}} \sim \chi^2(d). \quad (39)$$

Now, we know that its expected value is $\mathbb{E}[D_t] = d$. We can therefore define

$$\mathbb{E}\left[\frac{1}{D_t+\nu}\right] \geq \frac{1}{\mathbb{E}[D_t]+\nu} = \frac{1}{d+\nu}.$$

This inequality comes from the Jensen's inequality. The expected value of the weights w_t is derived as

$$\mathbb{E}[w_t] = \mathbb{E}\left[\frac{\nu+d}{\nu+D_t}\right] \geq 1. \quad (40)$$

With $a = (2\beta_1-1)/\beta_1$, we can then infer the mean of the weighted sum W_t

$$\begin{aligned} W_t &= \frac{\beta_1}{1-\beta_1}a^{t-1} + \sum_{i=1}^{t-1} w_i a^{t-1-i} \\ \mathbb{E}[W_t] &= \frac{\beta_1}{1-\beta_1}\mathbb{E}[a^{t-1}] + \mathbb{E}[w_t]\mathbb{E}\left[\sum_{i=1}^{t-1}a^{t-1-i}\right] \\ &= \frac{\beta_1}{1-\beta_1}\mathbb{E}[a^{t-1}] + \mathbb{E}[w_t]\frac{\beta_1}{1-\beta_1}\mathbb{E}[(1-a^{t-1})] \\ &= \frac{\beta_1}{1-\beta_1}\left\{\mathbb{E}[a^{t-1}] + \mathbb{E}[w_t](1-\mathbb{E}[a^{t-1}])\right\} \\ &\leq \frac{\beta_1}{1-\beta_1}\mathbb{E}[w_t]. \end{aligned} \quad (41)$$

The last inequality comes from $\mathbb{E}[w_t] \geq 1$ and $0 < a < 1$, i.e., $(1-\mathbb{E}[w_t])\mathbb{E}[a^{t-1}]+\mathbb{E}[w_t] \leq \mathbb{E}[w_t]$. Now, since $\mathbb{E}[W_t] > 0$, we have

$$\frac{\mathbb{E}[w_t]}{\mathbb{E}[W_t]} \geq \frac{1-\beta_1}{\beta_1}. \quad (42)$$

We move on to express the upper bound for $\mathbb{E}[\beta_w]$ where $\beta_w = W_{t-1}/(W_{t-1}+w_t)$. For this purpose, we make use of the Hartley and Ross unbiased estimator for the mean of the ratio between two random variables [46], [47]. Indeed, considering two random variables R and S, where S has no mass at 0, and $G = g(R/S) = R/S$; the Hartley and Ross identity states that

$$\mathbb{E}[G] = \frac{\mathbb{E}[R]}{\mathbb{E}[S]} - \frac{1}{\mathbb{E}[S]}\text{Cov}\left(\frac{R}{S}, S\right)$$

where $\text{Cov}(\cdot,\cdot)$ is the covariance between two variables. This relation is easily derived from the covariance formula, and furthermore, we can prove that, in our case, it is positive as derived below

$$\begin{aligned}\text{Cov}(f(S), S) &= \mathbb{E}[Sf(S)] - \mathbb{E}[S]\mathbb{E}[f(S)] \\ &= \mathbb{E}[(S - \mathbb{E}[S])f(S)] \\ &= \mathbb{E}[(S - \mathbb{E}[S])(f(S) - f(\mathbb{E}[S]))]\end{aligned}$$

where $\mathbb{E}[(S - \mathbb{E}[S])f(\mathbb{E}[S])]$ is zero. So that if the function $f(S)$ is an increasing one, the above equation, i.e., the covariance, will always be positive. For the t-momentum, $R = W_{t-1}$ and $S = W_{t-1}+w_t$. In that time, $f(S) = (S-w_t)/S = 1-w_t/S$ with $w_t > 0$ is an increasing function of S.

Based on the fact that the covariance between W_{t-1} and $W_{t-1}+w_t$ is positive, we can define an upper bound for $\mathbb{E}[G = \beta_w]$, as follows:

$$\begin{aligned}\mathbb{E}[\beta_w] = \mathbb{E}\left[\frac{W_{t-1}}{W_{t-1}+w_t}\right] &\leq \frac{\mathbb{E}[W_{t-1}]}{\mathbb{E}[W_{t-1}+w_t]} \\ &= \frac{1}{1+\mathbb{E}[w_t]\mathbb{E}[W_{t-1}]^{-1}} \\ &\leq \frac{1}{1+(1-\beta_1)\beta_1^{-1}} = \beta_1. \end{aligned} \tag{43}$$

□

REFERENCES

[1] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proc. COMPSTAT*. Berlin, Germany: Springer-Verlag, 2010, pp. 177–186.

[2] H. Robbins and S. Monro, "A stochastic approximation method," *Ann. Math. Statist.*, vol. 1, pp. 400–407, Sep. 1951.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[4] A. Gupta, A. Murali, D. P. Gandhi, and L. Pinto, "Robot learning in homes: Improving generalization and reducing dataset bias," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 9094–9104.

[5] M. Suchi, T. Patten, D. Fischinger, and M. Vincze, "EasyLabel: A semi-automatic pixel-wise object annotation tool for creating robotic RGB-D datasets," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 6678–6684.

[6] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[7] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[8] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," 2019, *arXiv:1904.09237*. [Online]. Available: http://arxiv.org/abs/1904.09237

[9] M. J. Holland and K. Ikeda, "Efficient learning with robust gradient descent," *Mach. Learn.*, vol. 108, nos. 8–9, pp. 1523–1560, Sep. 2019.

[10] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.

[11] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *USSR Comput. Math. Math. Phys.*, vol. 4, no. 5, pp. 1–17, Jan. 1964.

[12] Y. Nesterov, "A method for unconstrained convex minimization problem with the rate of convergence $O(1/k^2)$," in *Proc. Doklady*, vol. 269, 1983, pp. 543–547.

[13] N. L. Roux, M. Schmidt, and F. R. Bach, "A stochastic gradient method with an exponential convergence _rate for finite training sets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 2663–2671.

[14] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 315–323.

[15] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, pp. 2121–2159, Feb. 2011.

[16] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," 2012, *arXiv:1212.5701*. [Online]. Available: http://arxiv.org/abs/1212.5701

[17] T. Tieleman and G. Hinton, "Lecture 6.5-RMSPROP: Divide the gradient by a running average of its recent magnitude," *Neural Netw. Mach. Learn.*, vol. 4, no. 2, pp. 26–31, 2012.

[18] L. Luo, Y. Xiong, Y. Liu, and X. Sun, "Adaptive gradient methods with dynamic bound of learning rate," 2019, *arXiv:1902.09843*. [Online]. Available: http://arxiv.org/abs/1902.09843

[19] T. Schaul and Y. LeCun, "Adaptive learning rates and parallelization for stochastic, sparse, non-smooth gradients," 2013, *arXiv:1301.3764*. [Online]. Available: http://arxiv.org/abs/1301.3764

[20] C. Gulcehre, J. Sotelo, M. Moczulski, and Y. Bengio, "A robust adaptive stochastic gradient method for deep learning," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 125–132.

[21] Y. Haimin, P. Zhisong, and T. Qing, "Robust and adaptive online time series prediction with long short-term memory," *Comput. Intell. Neurosci.*, vol. 2017, Dec. 2017, Art. no. 9478952.

[22] M. Lerasle and R. I. Oliveira, "Robust empirical mean estimators," 2011, *arXiv:1112.3914*. [Online]. Available: http://arxiv.org/abs/1112.3914

[23] S. Minsker, "Geometric median and robust estimation in Banach spaces," *Bernoulli*, vol. 21, no. 4, pp. 2308–2335, Nov. 2015.

[24] G. Lugosi and S. Mendelson, "Risk minimization by median-of-means tournaments," 2016, *arXiv:1608.00757*. [Online]. Available: http://arxiv.org/abs/1608.00757

[25] D. Hsu and S. Sabato, "Loss minimization and parameter estimation with heavy tails," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 543–582, 2016.

[26] C. Brownlees *et al.*, "Empirical risk minimization for heavy-tailed losses," *The Ann. Statist.*, vol. 43, no. 6, pp. 2507–2536, 2015.

[27] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 46, no. 1, p. 96, 2019.

[28] A. Prasad, A. Sai Suggala, S. Balakrishnan, and P. Ravikumar, "Robust estimation via robust gradient estimation," 2018, *arXiv:1802.06485*. [Online]. Available: http://arxiv.org/abs/1802.06485

[29] O. Arslan, P. D. Constable, and J. T. Kent, "Convergence behavior of the em algorithm for the multivariate t-distribution," *Commun. Statist.-Theory Methods*, vol. 24, no. 12, pp. 2981–3000, 1995.

[30] F. Z. Doğru, Y. M. Bulut, and O. Arslan, "Doubly reweighted estimators for the parameters of the multivariate t-distribution," *Commun. Statist.-Theory Methods*, vol. 47, no. 19, pp. 4751–4771, Oct. 2018.

[31] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2017, *arXiv:1711.05101*. [Online]. Available: http://arxiv.org/abs/1711.05101

[32] S. Ram Dubey, S. Chakraborty, S. Kumar Roy, S. Mukherjee, S. Kumar Singh, and B. Baran Chaudhuri, "DiffGrad: An optimization method for convolutional neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 11, pp. 4500–4511, Nov. 2020.

[33] L. Liu *et al.*, "On the variance of the adaptive learning rate and beyond," 2019, *arXiv:1908.03265*. [Online]. Available: http://arxiv.org/abs/1908.03265

[34] J. Chen, D. Zhou, Y. Tang, Z. Yang, Y. Cao, and Q. Gu, "Closing the generalization gap of adaptive gradient methods in training deep neural networks," 2018, *arXiv:1806.06763*. [Online]. Available: http://arxiv.org/abs/1806.06763

[35] M. Zaheer, S. Reddi, D. Sachan, S. Kale, and S. Kumar, "Adaptive methods for nonconvex optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 9793–9803.

[36] L. Ziyin, Z. T. Wang, and M. Ueda, "LaProp: Separating momentum and adaptivity in adam," 2020, *arXiv:2002.04839*. [Online]. Available: http://arxiv.org/abs/2002.04839
[37] D. F. Nettleton, A. Orriols-Puig, and A. Fornells, "A study of the effect of different types of noise on the precision of supervised learning techniques," *Artif. Intell. Rev.*, vol. 33, no. 4, pp. 275–306, 2010.
[38] W. Shang, K. Sohn, D. Almeida, and H. Lee, "Understanding and improving convolutional neural networks via concatenated rectified linear units," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 2217–2225.
[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[40] M. Hardt, B. Recht, and Y. Singer, "Train faster, generalize better: Stability of stochastic gradient descent," 2015, *arXiv:1509.01240*. [Online]. Available: http://arxiv.org/abs/1509.01240
[41] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*. [Online]. Available: http://arxiv.org/abs/1707.06347
[42] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1587–1596.
[43] Y. Tsurumine, Y. Cui, E. Uchibe, and T. Matsubara, "Deep reinforcement learning with smooth policy update: Application to robotic cloth manipulation," *Robot. Auto. Syst.*, vol. 112, pp. 72–83, Feb. 2019.
[44] E. Coumans and Y. Bai. (2016). *Pybullet, a Python Module for Physics Simulation for Games, Robotics and Machine Learning*. [Online]. Available: http://pybullet.org
[45] A. Stooke and P. Abbeel, "Rlpyt: A research code base for deep reinforcement learning in PyTorch," 2019, *arXiv:1909.01500*. [Online]. Available: http://arxiv.org/abs/1909.01500
[46] H. O. Hartley and A. Ross, "Unbiased ratio estimators," *Nature*, vol. 174, no. 4423, pp. 270–271, Aug. 1954.
[47] L. A. Goodman and H. O. Hartley, "The precision of unbiased ratio-type estimators," *J. Amer. Stat. Assoc.*, vol. 53, no. 282, pp. 491–508, Jun. 1958.

Wendyam Eric Lionel Ilboudo received the bachelor's degree in electronics and automation from New Dawn University, Burkina Faso, in 2016. He is currently pursuing the master's degree with the Nara Institute of Science and Technology, Nara, Japan.

His current research interest includes robust optimization methods for machine learning, and reinforcement learning applied to robotics.

Mr. Ilboudo received a MEXT Scholarship from the Japanese Government in 2018.

Taisuke Kobayashi (Member, IEEE) received the B.E., M.E., and Ph.D. degrees in engineering from Nagoya University, Aichi, Japan, in 2012, 2014, and 2016, respectively.

He is currently an Assistant Professor with the Graduate School of Information Science, Nara Institute of Science and Technology (NAIST), Nara, Japan. His research interests include locomotion control, nonlilnear dynamics, and autonomous systems.

Kenji Sugimoto (Member, IEEE) received the M.S. and Ph.D. degrees from Kyoto University, Kyoto, Japan, in 1982 and 1989, respectively.

He was with Mitsubishi Electric Corporation, Japan. He became an Assistant Professor with Kyoto University, in 1985. He was an Associate Professor with Okayama University, Japan and Nagoya University, Japan. Since 1999, he has been a Professor with the Nara Institute of Science and Technology, Nara, Japan. His current research interests include control theory and system science.

Dr. Sugimoto is a member of SICE and ISCIE.