# Competitive Normalized Least-Squares Regression

Waqas Jamil and Abdelhamid Bouchachia, *Senior Member, IEEE*

***Abstract*— Online learning has witnessed an increasing interest over the recent past due to its low computational requirements and its relevance to a broad range of streaming applications. In this brief, we focus on online regularized regression. We propose a novel efficient online regression algorithm, called online normalized least-squares (ONLS). We perform theoretical analysis by comparing the total loss of ONLS against the normalized gradient descent (NGD) algorithm and the best off-line LS predictor. We show, in particular, that ONLS allows for a better bias-variance tradeoff than those state-of-the-art gradient descent-based LS algorithms as well as a better control on the level of shrinkage of the features toward the null. Finally, we conduct an empirical study to illustrate the great performance of ONLS against some state-of-the-art algorithms using real-world data.**

***Index Terms*— Competitive analysis, least-squares, prediction.**

## I. Introduction

Sequential learning[1] is about qualitatively predicting an output upon the presentation, at each trial, of an input from a sequence (stream). We consider the following model:

$$y_t = \langle w_{t-1}, x_t \rangle + \epsilon_t \tag{1}$$

where $w_{t-1} \in \mathbb{R}^n$ is some weight vector, $x_t \in \mathbb{R}^n$ is the input, $y_t \in \mathbb{R}$ is the output, and $\epsilon_t \in \mathbb{R}$ denotes the noise. The goal is to obtain an estimate $\hat{y}_t \in \mathbb{R}$ for $y_t$ upon maintaining a weight vector $w_{t-1} \in \mathbb{R}^n$ for each trial $t = 1, 2, \ldots$

The application of model (1) includes *a priori* filtering, posterior filtering, and online regression. *A priori* filtering is used to recover uncorrupted output $\langle u, x_t \rangle$, before receiving the output $y_t$. The overall discrepancy (error) after T steps is given in the form of a cumulative sum: $\sum_{t=1}^{T}(\langle u, x_t \rangle - \langle w_{t-1}, x_t \rangle)^2$. In *a posteriori* filtering, we filter out the noise using the output $y_t$. The error is formulated as a cumulative sum: $\sum_{t=1}^{T}(\langle u, x_t \rangle - \langle w_t, x_t \rangle)^2$. Notice that in *a posteriori* filtering, we are able to use the most recent weight vector $w_t$ to measure the quality of the filter (error). In contrast, for *a priori* filtering, we use $w_{t-1}$ because we do not have access to the output $y_t$, which resembles the online learning setting. However, in filtering, the goal is not to estimate the output; instead, we want to recover the output by assuming that it is corrupted by some noise. In the following, $\langle u, x_t \rangle$ will refer to the prediction of the off-line algorithm.

Interested in online regression, we suggest a novel update rule which is inspired by the well-known NLS method from the filtering literature (see [7]). The proposed technique directly affects how neural networks learn. In this work, it is shown that the proposed technique has an advantage over the state of the art due to the inclusion of a well-known ridge penalty term. For details on the advantages of the ridge penalty, see [9]. The analysis of the least-squares algorithm was understood without normalization and given in the late 1990s. The main drawback of the least-squares algorithm is that it is sensitive to the scaling, which makes it very hard to choose a learning rate that guarantees the stability of the algorithm [13]. This work fills the gap in the literature by giving an analysis of a normalized algorithm.

To answer the question of how well an online learning algorithm predicts, often, competitive analyses [22] are performed. An algorithm is said to be competitive if it satisfies the following:

$$L_T \leq cL_T^* + R_T \tag{2}$$

where $L_T$ is the cumulative loss up-till time $t$, and $L_T^* = \inf_w \|\mathbf{Y} - \mathbf{X}w\|^2$, where $w \in \mathbb{R}^n$, $\mathbf{X} \in \mathbb{R}^{t \times n}$, and $\mathbf{Y} \in \mathbb{R}^t$. $R_T$ denotes the regret, and $c$ denotes a constant. An interesting feature of such a definition of competitiveness is that it requires an algorithm to perform well for both "NP-hard" and "NP-easy" inputs. This is a stronger notion than the worst case analysis [11], where the performance of the algorithm is only measured for "NP-hard" inputs.

In the literature of learning theory, regression algorithms learn by updating their weight (parameter) vectors at each trial. Such an update requires adjusting the inverse of the covariance matrix at each trial $t$. One can use gradient descent to approximate the inverse of the covariance matrix, which is computationally more efficient. In both approaches, the constant $c$ in the performance guarantee (2) is greater than or equal to unity. For example, covariance-based aggregating algorithm for regression (AAR) and ridge regression (RR) [25], recursive least-squares (RLS) [17], and adaptive regularization of weights (AROW) [6] for model (1) [18] has $c = 1$, and the gradient-based algorithms [5] have $c = 2.25$. In the completely online setup, the regret of the gradient-based algorithms is $\mathcal{O}(1)$; whereas, for covariance-based algorithms (for example, AAR), it is possible to have a logarithmic regret. Also, the bound of the gradient-based algorithms is better for noise-free data. However, such bound is weak when the true regression function includes the noise term $\epsilon_t$ since the regret is only bounded by $\mathcal{O}(1)$.

In this brief, we make no assumptions on the incoming data. Our work is comparable to the work done in [5], where the performance guarantee on cumulative normalized square loss is obtained by using the generalized gradient descent. In order to obtain the performance guarantee, first, a lower bound on the progress (see [5, Lemma IV.4]) is computed assuming that $\eta = (\alpha/\|x_t\|^2)$ (see [5, Th. IV.2]), where $0 < \alpha < 2$. The chosen $\alpha$ ensures that the performance guarantee on the normalized cumulative loss is held. We dwell further on the discussion done by Cesa-Bianchi *et al.* [5] and do not impose a similar restriction on $\eta$ when bounding the progress of online normalized least-squares (ONLS). We bound $1/\|x_t\|$ and provide a performance guarantee on square loss and relax this condition for the guarantee on normalized squared loss. Consequently, the proposed algorithm's guarantees have the tuning parameter next to the ridge penalty—indicating superior bias-variance tradeoff properties than the generalized GD update rule. In summary, the major contributions of this brief are as follows.:

1) derivation of the ONLS algorithm for regression;
2) development of a competitive analysis of ONLS;
3) empirical and comparative study using real-world data.

Manuscript received October 19, 2018; revised May 29, 2019, September 26, 2019, January 22, 2020, and May 23, 2020; accepted July 11, 2020. Date of publication August 5, 2020; date of current version July 7, 2021. This work was supported by the European Commission through the Horizon 2020 for the Project *PROTEUS* under Grant 687691. *(Corresponding author: Waqas Jamil.)*

The authors are with the Department of Computing and Informatics, Bournemouth University, Poole BH12 5BB, U.K. (e-mail: wjamil@bournemouth.ac.uk; abouchachia@bournemouth.ac.uk).

Digital Object Identifier 10.1109/TNNLS.2020.3009777

[1]We use sequential learning and online learning interchangeably.

The structure of this brief reflects on these contributions in the subsequent sections after presenting the related work in Section II.

## II. Motivation and Related Work

RLS is a popular algorithm in the area of linear regression and consists of computing the weights as follows:

$$w_t = \underset{w}{\operatorname{argmin}} \left( \sum_{s=1}^{t} r^{t-s} (y_s - \langle w, x_s \rangle)^2 \right).$$

In each iteration, $t$, the prediction $\hat{y}_t = w'_{t-1} x_t$, is made, and after receiving the true output $y_t$, the weights are updated

$$w_t = w_{t-1} + \frac{A_{t-1}^{-1} (y_t - x'_t w_{t-1}) x_t}{r + x'_t A_{t-1} x_t} \tag{3}$$

where $A_t^{-1} = r A_{t-1}^{-1} + x_t x'_t$, with $r > 0$ and $A_0 = I \in \mathbb{R}^{n \times n}$. There exist many similar algorithms, such as AAR, RR, and AROWR. In particular, the weight update rule of AROWR is the same as RLS. The only difference is how the covariance matrix is updated: $A_t^{-1} = A_{t-1}^{-1} + (1/r) x_t x'_t$. AAR's and RR's update rules can be obtained by setting $r = 1$ in (3) with $A_t^{-1} = A_{t-1}^{-1} + x_t x'_t$ and $A_0 = a^{-1} I$, where $a > 0$. Another difference between AAR and the other algorithms is that AAR divides its prediction $w'_{t-1} x_t$ by $1 + x'_t A_{t-1} x_t$, whereas RR, AROWR, and RLS do not do that. Moreover, AAR is the only algorithm among the four that has the ability to perform shrinkage. It is worth noting that all of these algorithms perform regression when the target is assumed to be stationary. Sometimes, they are used as building blocks to develop second-order nonstationary algorithms [18] by adding a penalty term to handle drift of the target (nonstationarity).

The implementation of (3) has the time complexity $\mathcal{O}(n^2)$ that is significant for high-dimensional data. Often, LS [26] is considered as a less demanding solution since its time complexity is $\mathcal{O}(n)$. LS replaces the term $(A_{t-1}^{-1} / (r + x'_t A_{t-1}^{-1} x_t))$ by the learning rate $\eta > 0$, yielding the following update rule:

$$w_t = w_{t-1} + \eta (y_t - \langle x_t, w_{t-1} \rangle) x_t. \tag{4}$$

The LS algorithm not only has a better time complexity but it is also $H^\infty$ optimal for $\eta \le (1/\|x_t\|^2)$. That is

$$\max_u \frac{\sum_{t=1}^T (\langle u, x_t \rangle - \langle w_{t-1}, x_t \rangle)^2}{\sum_{t=1}^T (\langle u, x_t \rangle - y_t)^2 + \frac{\|x_t\|^2}{\eta}} \le 1. \tag{5}$$

In contrast, in the case of RLS, the right-hand side of (5) is replaced by 4. For further details on the matter, see [12].

In practice, often, normalized LS performs better than LS because NLS is not sensitive to the scale of the input [2], [3]. The existing work on NLS applies a normalized square loss to derive the update of the weights [5], [15]

$$w_t = w_{t-1} + \frac{\eta}{\|x_t\|^2} (y_t - \hat{y}_t) x_t \tag{6}$$

$$w_t = w_{t-1} + \frac{\eta}{1 + \eta \|x_t\|^2} (y_t - \hat{y}) x_t \tag{7}$$

for $\eta > 0$. When $x_t = 0$ or $\eta = 0$ with the convention that $(0/0) = 0$, the rules (6) and (7) output $w_t = w_{t-1}$.

Up until now, we have briefly revised some of the popular existing approaches for regression using the squared loss and normalized square loss. The squared loss is a renowned loss function despite not being robust. In particular, in the presence of substantial outliers, the square loss function is not the preferred choice since it penalizes the mistakes with more severity (the difference between actual and predicted is squared) compared with some other loss functions, such as absolute loss and normalized square loss. For this reason, in the next section, we will study a tunable loss function that (loosely speaking) incorporates the features of both absolute and squared loss functions.

## III. Derivation and Analysis of ONLS

ONLS is an online regression algorithm, and so, it observes the following protocol:

```
FOR t=1,2,...
   (1) receive $x_t \in \mathbb{R}^n$
   (2) predict $\hat{y}_t = \langle w_t, x_t \rangle$
   (3) receive $y_t \in \mathbb{R}$
   (4) update  $w_t \in \mathbb{R}^n$
END FOR
```

Protocol 1. Online regression.

In Protocol 1, it is assumed that the prediction is given by $w'_t x_t$. Thus, the problem at hand is to design the update rule, which leads us to the following lemma.

*Lemma 1:* The following minimization problem with respect to $w_t$:

$$\min \left( \sum_{t=1}^T (w_t - w_{t-1})^2 \right)$$

with the constraint $y = w'_t x_t$ has the following solution:

$$w_t = w_{t-1} + \frac{(y_t - \hat{y}_t) x_t}{\|x_t\|^2}.$$

*Proof:* The proof is given in Appendix A. □

*Remark 1:* We perform the analysis of the following update rule:

$$w_t = w_{t-1} + \frac{(y_t - \hat{y}_t) x_t}{\eta + \|x_t\|^2} \tag{8}$$

for $\eta > -\|x_t\|^2$. The obvious advantage of using (8) is that we do not require any convention for the case when $\|x_t\| \to 0$. Later, we show that the addition of $\eta$ in the denominator results in a better performance guarantee.

ONLS is presented in Algorithm 1, where the weight vector is initially set to $\mathbf{0} \in \mathbb{R}^n$. The update rule can be written in the form: $w_t = w_{t-1} + \lambda x_t$, where $\lambda = ((y_t - \hat{y}_t)/(\eta + \|x_t\|^2)) \in \mathbb{R}$.

**Algorithm 1** ONLS
```
FOR t=1,2,...
  (1) receive $x_t \in \mathbb{R}^n$
  (2) predict $\hat{y}_t = \langle w_t, x_t \rangle$
  (3) receive $y_t \in \mathbb{R}$
  (4) update $w_t$ using eq.(8)
END FOR
```

We now analyze ONLS using the technique (difference of sum of squares) suggested by Duda *et al.* [8] for convergence analysis, and we start by the following theorem that shows the bound on ONLS' performance.

*Theorem 1:* For any sequence $x_1, y_1, x_2, y_2, \ldots$ with predictions $\hat{y}_1, \hat{y}_2, \ldots,$ given by Algorithm 1, the following holds:

$$L_T \le \frac{1}{\beta(1-\beta)} \inf_u ((\eta + X^2) \|u\|^2 + L_T(u)) + \mathcal{O}(1)$$

for $(1/\|x_t\|) \leq (1/X)$. For $\eta = bX^2$ and $\beta = (1/2)$, the following holds:

$$L_T \leq 4 \inf_u((b+1)X^2\|u\|^2 + L_T(u)) + \mathcal{O}(1)$$

where $b > -1$, $L_T$ is the cumulative square loss, and $L_T(u)$ is the cumulative square loss of the off-line LS algorithm.

*Proof:* The proof is given in Appendix B. □

The result obtained in Theorem 1 fulfills (2) with $c = (1/(\beta(1-\beta)))$, $L_T^*\mathbb{R}^t = \inf_u((\eta + X^2)\|u\|^2 + L_T(u))$, and $R_T = \mathcal{O}(1)$. Theorem 1[2] asserts for $\eta = 0$ and $\beta = (1/2)$

$$L_T \leq 4 \inf_u(X^2\|u\|^2 + L_T(u)). \tag{9}$$

It is clear that the addition of $\eta$ in the update rule of Lemma 1 is advantageous. In Theorem 1, the addition of $\eta$ decreases the dependence on the size of the data. It is worth noticing that (9) is the performance guarantee for the algorithm derived in Lemma 1, where we bound the input by the Euclidean norm. Also, notice when $\beta = (1/2)$ and as $b \to -1 \implies \inf_u((b+1)X^2\|u\|^2 + L_T(u)) \to L_T \leq 4\inf_u L_T(u)$, that is, ONLS is at most four times worse than the true regression function.

The following theorem presents the performance guarantee on the normalized squared loss.

*Theorem 2:* For any sequence $x_1, y_1, x_2, y_2, \ldots$ with $\eta = b\|x_t\|^2$ and predictions $\hat{y}_1, \hat{y}_2, \ldots,$ given by Algorithm 1, the following holds:

$$\bar{L}_T \leq \frac{1}{\beta(1-\beta)} \inf_{u\in\mathbb{R}^n}((b+1)(1-\beta)\|u\|^2 + \bar{L}_T(u)) + \mathcal{O}(1)$$

such that $b > -1$, $0 < \beta < 1$, $\bar{L}_T$ is the algorithms' loss, and $\bar{L}_T(u)$ is loss of the off-line algorithm.

*Proof:* Notice that

$$\beta\sum_{t=1}^{T}\frac{(\hat{y}_t - y_t)^2}{\|x_t\|^2}\frac{\|x_t\|^2}{\eta + \|x_t\|^2} = \frac{\beta}{(b+1)}\sum_{t=1}^{T}\frac{(\hat{y}_t - y_t)^2}{\|x_t\|^2}$$

and

$$\sum_{t=1}^{T}\frac{(y_t - \langle u, x_t\rangle)^2}{\|x_t\|^2}\frac{\|x_t\|^2}{(1-\beta)(\eta + \|x_t\|^2)} = \frac{1}{(1-\beta)(b+1)}\sum_{t=1}^{T}\frac{(y_t - \langle u, x_t\rangle)^2}{\|x_t\|^2}$$

for $\eta = b\|x_t\|^2$. Thus, from (14) (see Appendix B)

$$\frac{\beta}{(b+1)}\sum_{t=1}^{T}\frac{(\hat{y}_t - y_t)^2}{\|x_t\|^2} - \frac{1}{(1-\beta)(b+1)}\sum_{t=1}^{T}\frac{(y_t - \langle u, x_t\rangle)^2}{\|x_t\|^2} \leq \|u\|^2$$

and the result follows. □

In Theorem IV.2, the guarantee does not depend on the size of the input, and we do not bound the input. Also, the performance guarantee has no assumptions on the input, the output, and the weights.

The result of Theorem 2 fulfills (2) with $c = 4$ (when $\beta = (1/2)$) instead of $c = 2.25$, as mentioned in [5], for normalized gradient descent (NGD) (Theorem IV.2). However, for NGD, $L_T^* = \inf_{u\in\mathbb{R}^n}\|u\|^2 + \bar{L}_T$ instead of $\inf_{u\in\mathbb{R}^n} a\|u\|^2 + \bar{L}_T$ with $a > 0$.

*Corollary 1:* As $\|u\|^2 \to \infty$, Algorithm 1 has a better guarantee than NGD for $\bar{L}_T$ at any given trial $T = 1, 2, \ldots$ if $0 < a \leq 0.5625$.

*Proof:* By solving $4a\|u\|^2 + 4\bar{L}_T \leq 2.25\|u\|^2 + 2.25\bar{L}_T$, we obtain $0 < a \leq 0.5625 - 0.4375(\bar{L}_T/\|u\|^2)$. Thus, as $\|u\|^2 \to \infty$, $(\bar{L}_T/\|u\|^2) \to 0 \implies 0 < a \leq 0.5625$. □

*Remark 2:* Similarly, for the case of $L_T$, NGD is outperformed as $\|u\|^2 \to \infty$ and $(1/\|x_t\|) \leq (1/X)$ if $-1 < b \leq 0.5625X^2$.

We now present a guarantee that includes the learning rate $\eta$ in the cumulative loss, which we will refer to as tunable loss function.

*Theorem 3:* For any sequence $x_1, y_1, x_2, y_2, \ldots$ with predictions $\hat{y}_1, \hat{y}_2, \ldots,$ given by Algorithm 1, the following holds:

$$\hat{L}_T \leq \inf_{u\in\mathbb{R}^n}(2\|u\|^2 + 4\hat{L}_T(u)) + \mathcal{O}(1)$$

such that

$$\hat{L}_T = \sum_{t=1}^{T}\frac{(y_t - \hat{y}_t)^2}{\eta + \|x_t\|^2} \text{ and } \hat{L}_T(u) = \sum_{t=1}^{T}\frac{(y_t - (u_t, x_t))^2}{\eta + \|x_t\|^2}$$

with $\eta > -\|x_t\|^2$ and $0 < \beta < 1$.

*Proof:* We write (14) (see Appendix B) as

$$\hat{L}_T \leq \inf_{u\in\mathbb{R}^n}\frac{1}{\beta}\|u\|^2 + \frac{1}{\beta(1-\beta)}\hat{L}_T(u).$$

By setting $\beta = (1/2)$, we obtain the desired result. □

To conclude this theoretical analysis, we state the following: the addition of the learning rate is advantageous in the ONLS algorithm. The ridge penalty $\|u\|^2 \to \infty$, and ONLS algorithm has a better guarantee than NGD's guarantee when $-1 < b \leq 0.5625X^2$ and $0 < a \leq 0.5625$ for the squared and normalized squared loss, respectively. The presence of $a > 0$ and $b > -1$ next to the ridge penalty in the guarantees and update rule of ONLS implies a better control over the bias-variance tradeoff than NGD case where $a = b = 1$.

## IV. Empirical Study

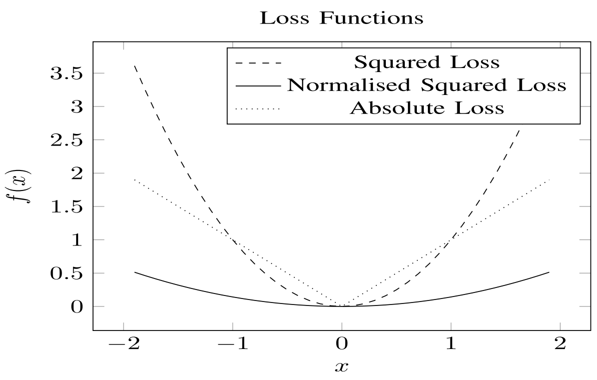
Fig. 1 compares some of the renowned loss functions and the behavior of the loss function studied in Theorem 3. Notice that when the learning rate $\eta = 0$, the tunable loss is the same as the normalized squared loss. When $\eta = -0.9\|x\|^2$, the tunable loss penalty is in a similar range as the absolute loss but with the shape of the squared loss. Also, the tunable square loss is differentiable for all values of $\eta > -\|x\|^2$, at every value of $x$. The same statement does not hold for the absolute loss. Thus, the suggested tunable loss function has the robustness of the absolute loss but in the shape of the squared loss.

The primary goal of this study is to compare[3] ONLS against NGD, but, for the sake of completeness, we also compare it against the most theoretically studied variant of gradient method known as Adam [14], [21] as well as against the  [5], ORR [17], and ONS [19]. The objective is to be as close as possible off-line solution (please see 2) $P_T^* = \mathbf{X}w^*$, where $w^* = \text{argmin}_w \|\mathbf{Y} - \mathbf{X}w\|^2$. The $P_T^*$ solution considers the entire data $\mathbf{X} \in \mathbb{R}^{t\times n}$ and $\mathbf{Y} \in \mathbb{R}^t$. Table I contains the minimum (min), maximum (max), and median (med) cook distances for outliers and mean and variance (var) for the level of noise.

Data sets used in this study are Gaze [20], NO$_2$ [24], ISE [1](Istanbul Stock Exchange) F-16 [23], Friedman [10], and Weather [4]. Gaze data consist of 450 observations of 12 features, estimating the positions of the eyes of the subject when the subject is looking at the monitor. NO$_2$ data consist of 500 observations from a road air pollution study collected by the Norwegian Public Roads Administration. The ISE data have 536 observations with eight attributes. Ailerons (F-16) data consist of 13 750 observations with a total of 40 attributes that describe the status of the F-16. Friedman data are a synthetic data set with ten features and 40 768 rows. Weather data

[2]The guarantee is expressed as a function of $\inf_u \sum_t (y_t - \langle u, x_t\rangle)^2$. The infimum is taken over all $u$.

[3]For all algorithms in all experiments, $\eta = (1/T)$, where $T$ denotes the length of the data.



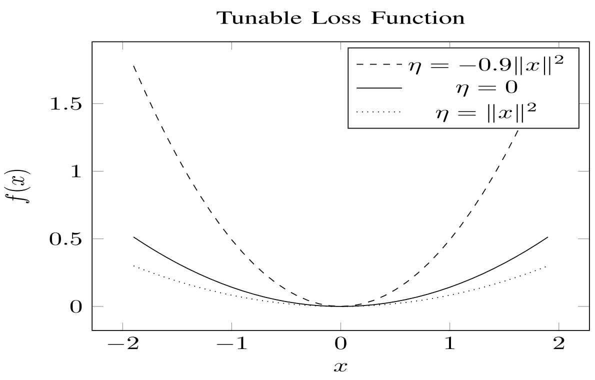


Fig. 1. Tunable loss function (see Theorem 3).

TABLE I
DATA CHARACTERIZATION: COOK DISTANCE, MEAN, AND VARIANCE

| Data | max.cook.dist | min.cook.dist | med.cooks.dist | label mean | label variance | model variance |
|---|---|---|---|---|---|---|
| Gaze | $1.90 \times 10^{-1}$ | $1.35 \times 10^{-8}$ | $7.18 \times 10^{-4}$ | $5.44 \times 10^{2}$ | $6.31 \times 10^{4}$ | $3.29 \times 10^{3}$ |
| ISE | $1.37 \times 10^{-1}$ | $7.28 \times 10^{-10}$ | $4.23 \times 10^{-4}$ | $1.55 \times 10^{-3}$ | $4.46 \times 10^{-4}$ | $3.23 \times 10^{-5}$ |
| $NO_2$ | $4.25 \times 10^{-2}$ | $3.11 \times 10^{-8}$ | $7.52 \times 10^{-4}$ | $2.18 \times 10^{-6}$ | $1.00 \times 10^{0}$ | $4.98 \times 10^{-1}$ |
| $F-16$ | $5.10 \times 10^{-2}$ | $1.50 \times 10^{-6}$ | $2.30 \times 10^{-5}$ | $-8.68 \times 10^{-4}$ | $1.69 \times 10^{-7}$ | $3.01 \times 10^{-8}$ |
| Friedman | $9.17 \times 10^{-6}$ | $1.61 \times 10^{-15}$ | $9.83 \times 10^{-4}$ | $1.44 \times 10^{-1}$ | $2.50 \times 10^{-1}$ | $6.92 \times 10^{0}$ |
| Weather | $9.18 \times 10^{-6}$ | $1.61 \times 10^{-15}$ | $9.83 \times 10^{-4}$ | $1.09 \times 10$ | $1.14 \times 10^{2}$ | $1.15 \times 10^{0}$ |

have historical weather around Szeged, Hungary, from 2006 to 2016 with nine features and 96 453 observations. Please see Table I for characteristics of these data sets.

Table II compares root-mean-squared error (RMSE), coefficient of determination ($R^2$), and mean absolute error (MAE) of the algorithms. Overall, ONLS performs best, and Adam performs worst on these data sets in terms of RMSE, $R^2$, and MAE. ONLS is a good choice when the true regression function is not corrupted by noise (see Table II). Importantly, ONLS provides a significant improvement over NGD in all the studied scenarios.

## V. CONCLUSION

We presented an exact formulation of the proposed algorithm, ONLS, along with its performance guarantee. We compared it against to the state-of-the-art LS-regression algorithms, showing that it allows for a better bias-variance tradeoff while providing feature shrinkage. In the future, we will study the tightness of the bounds of ONLS and potentially those of the state-of-the-art algorithms used in this study.

## APPENDIX A
## PROOF OF LEMMA 1

Minimizing $\sum_{t=1}^{T}(w_t - w_{t-1})^2$ under the constraint $y_t - w_t' x_t = 0$. Introducing the Lagrangian multipliers $\alpha_t,\ t = 1, 2, \ldots, T$ and instead of solving the primal optimization problem mentioned earlier, we find the saddle point of the following:

$$\sum_{t=1}^{T}(w_t - w_{t-1})^2 + \sum_{t=1}^{T}\alpha_t\big(y_t - w_t' x_t\big). \tag{10}$$

In accordance with the Kuhn–Tucker theorem [16], there exists values of Lagrangian multipliers $\alpha = \alpha^{KT}$ for which solving the primal problem is equivalent to finding the saddle point. Thus

$$\frac{\partial}{\partial w_t}\left(\sum_{t=1}^{T}(w_t - w_{t-1})^2 + \sum_{t=1}^{T}\alpha_t\big(y_t - w_t' x_t\big)\right) = 0$$

$$w_t = w_{t-1} - \frac{1}{2}\alpha_t x_t. \tag{11}$$

TABLE II
RESULTS: ONLS VERSUS NGD, $P_T^*$, AND OTHERS

| Algorithm | RMSE | $R^2$ | MAE |
|---|---|---|---|
| **Gaze** | | | |
| ORR | $2.19 \times 10^{17}$ | $1.19 \times 10^{-4}$ | $1.35 \times 10^{16}$ |
| ONS | $5.33 \times 10^{3}$ | $9.91 \times 10^{-4}$ | $1.06 \times 10^{3}$ |
| NGD | $4.39 \times 10^{2}$ | $4.20 \times 10^{-3}$ | $3.66 \times 10^{2}$ |
| Adam | $4.97 \times 10^{4}$ | $1.74 \times 10^{-3}$ | $4.26 \times 10^{4}$ |
| ONLS | $2.70 \times 10^{2}$ | $1.04 \times 10^{-1}$ | $2.19 \times 10^{2}$ |
| $P_T^*$ | $5.65 \times 10$ | $9.49 \times 10^{-1}$ | $4.48 \times 10$ |
| **ISE** | | | |
| ORR | $2.79 \times 10^{-2}$ | $4.85 \times 10^{-1}$ | $1.98 \times 10^{-2}$ |
| ONS | $2.08 \times 10^{-2}$ | $5.50 \times 10^{-1}$ | $1.56 \times 10^{-2}$ |
| NGD | $8.90 \times 10^{-2}$ | $5.57 \times 10^{-1}$ | $1.40 \times 10^{-2}$ |
| Adam | $5.48 \times 10^{-2}$ | $4.20 \times 10^{-1}$ | $3.95 \times 10^{-2}$ |
| ONLS | $7.12 \times 10^{-3}$ | $8.87 \times 10^{-1}$ | $4.87 \times 10^{-3}$ |
| $P_T^*$ | $5.63 \times 10^{-3}$ | $9.29 \times 10^{-1}$ | $4.30 \times 10^{-3}$ |
| $\mathbf{NO_2}$ | | | |
| ORR | $8.90 \times 10^{2}$ | $1.59 \times 10^{-1}$ | $4.78 \times 10^{2}$ |
| ONS | $8.25 \times 10^{-1}$ | $4.04 \times 10^{-1}$ | $6.23 \times 10^{-1}$ |
| NGD | $9.65 \times 10^{-1}$ | $3.06 \times 10^{-1}$ | $7.63 \times 10^{-1}$ |
| Adam | $2.15 \times 10^{0}$ | $1.31 \times 10^{-1}$ | $1.64 \times 10^{0}$ |
| ONLS | $8.86 \times 10^{-1}$ | $3.91 \times 10^{-1}$ | $6.62 \times 10^{-1}$ |
| $P_T^*$ | $7.01 \times 10^{-1}$ | $5.07 \times 10^{-1}$ | $5.47 \times 10^{-1}$ |
| $\mathbf{F-16}$ | | | |
| ORR | $1.75 \times 10^{7}$ | $2.83 \times 10^{-4}$ | $1.60 \times 10^{6}$ |
| ONS | $2.30 \times 10^{4}$ | $1.11 \times 10^{-2}$ | $1.79 \times 10^{4}$ |
| NGD | $9.41 \times 10^{-4}$ | $2.70 \times 10^{-3}$ | $8.45 \times 10^{-4}$ |
| Adam | $1.55 \times 10^{4}$ | $4.44 \times 10^{-3}$ | $1.05 \times 10^{2}$ |
| ONLS | $5.90 \times 10^{-4}$ | $2.02 \times 10^{-1}$ | $3.40 \times 10^{-4}$ |
| $P_T^*$ | $1.73 \times 10^{-4}$ | $8.23 \times 10^{-1}$ | $1.27 \times 10^{-4}$ |
| **Friedman** | | | |
| ORR | $4.73 \times 10^{5}$ | $1.69 \times 10^{-2}$ | $3.70 \times 10^{5}$ |
| ONS | $2.66 \times 10^{0}$ | $7.20 \times 10^{-1}$ | $2.06 \times 10^{0}$ |
| NGD | $1.13 \times 10^{-1}$ | $3.49 \times 10^{2}$ | $1.01 \times 10^{-1}$ |
| Adam | $1.78 \times 10^{-1}$ | $2.42 \times 10^{2}$ | $1.69 \times 10^{-1}$ |
| ONLS | $3.73 \times 10^{0}$ | $5.30 \times 10$ | $2.91 \times 10^{0}$ |
| $P_T^*$ | $2.63 \times 10^{0}$ | $7.22 \times 10$ | $2.03 \times 10^{0}$ |
| **Weather** | | | |
| ORR | $5.38 \times 10^{15}$ | $1.55 \times 10^{-5}$ | $1.34 \times 10^{14}$ |
| ONS | $5.73 \times 10^{5}$ | $5.18 \times 10^{-1}$ | $5.51 \times 10^{5}$ |
| NGD | $1.29 \times 10$ | $1.19 \times 10^{-3}$ | $1.06 \times 10$ |
| Adam | $7.33 \times 10^{2}$ | $2.50 \times 10^{-3}$ | $6.33 \times 10^{2}$ |
| ONLS | $1.96 \times 10^{0}$ | $9.67 \times 10^{-1}$ | $7.23 \times 10^{-1}$ |
| $P_T^*$ | $1.07 \times 10^{0}$ | $9.90 \times 10^{-1}$ | $8.43 \times 10^{-1}$ |

Substituting the obtained value of $w_t$ from (11) in the constraint, we get

$$\alpha_t = \frac{2}{\|x_t\|^2}(\hat{y}_t - y_t). \tag{12}$$

Substitution of $\alpha_t$ from (12) in (11) gives

$$w_t = w_{t-1} + \frac{(y_t - \hat{y}_t)x_t}{\|x_t\|^2}. \tag{13}$$

In order to avoid the scenario $w_t \to \infty$ as $\|x_t\|^2 \to 0$, we use the convention $(0/0) = 0$.

## Appendix B Proof of Theorem 1

*Lemma 2:* Let $\hat{y}_t = \langle w_{t-1}, x_t\rangle$, $w_t = w_{t-1} + \lambda x_t$, where $x_t, w_{t-1}, u \in \mathbb{R}^n$, $y_t \in \mathbb{R}$ and $\lambda = ((y_t - \hat{y}_t)/(\eta + \|x_t\|^2))$, and the following holds:

$$\|w_{t-1} - u\|^2 - \|w_t - u\|^2$$
$$= (y_t - \hat{y}_t)^2\left(\frac{2}{\eta + \|x_t\|^2} - \frac{\|x_t\|^2}{\eta + \|x_t\|^2}\right) - \frac{2(y_t - \hat{y}_t)(y_t - \langle u, x_t\rangle)}{\|x_t\|^2 + \eta}.$$

*Proof:*

$$\|w_t - u\|^2 - \|w_{t-1} - u\|^2$$
$$= 2\lambda\langle x_t, (w_{t-1} - u)\rangle + \lambda^2\|x_t\|^2$$
$$= 2\lambda(\hat{y}_t - y_t) + 2\lambda(y_t - \langle u, x_t\rangle) + \lambda^2\|x_t\|^2.$$

Substitution of $\lambda = ((y_t - \hat{y}_t)/(\eta + \|x_t\|^2))$ leads to the desired result. □

*Lemma 3:* For all $a, b, r, \beta \in \mathbb{R}$ such that $0 < \beta < 1$

$$a^2 - ab \geq \beta(ab)^2 - \frac{b^2}{4(1-\beta)}.$$

*Proof:* The inequality is equivalent to the following:

$$a^2 - ab - \beta(ab)^2 + \frac{b^2}{4(1-\beta)} \geq 0$$
$$\frac{4a^2 - 8a^2\beta + 4a^2\beta^2 + b^2 - 4ab(1-\beta)}{4(1-\beta)} \geq 0.$$

It is clear that the left-hand side can be written as $(((2a - 2a\beta) - b)^2/4(1-\beta))$ for $0 < \beta < 1$; thus, the inequality holds. □

We now prove a lower bound on Lemma 2 using inequality proven in Lemma 3. This can be interpreted as the lower bound on the progress per trial of Algorithm 1.

*Lemma 4:* Let $\hat{y}_t = \langle w_{t-1}, x\rangle, w_t = w_{t-1} + ((y_t - \hat{y}_t)/(\eta + \|x_t\|^2))x_t$, where $x_t, w_t, u \in \mathbb{R}^n$, $y_t \in \mathbb{R}$, and the following holds:

$$\|w_{t-1} - u\|^2 - \|w_t - u\|^2 \geq \frac{\beta(y_t - \hat{y}_t)^2}{(\eta + \|x_t\|)^2} - \frac{(y_t - \langle u, x\rangle)^2}{(1-\beta)(\eta + \|x_t\|^2)}$$

for $0 < \beta < 1$.

*Proof:* From Lemma 2

$$\|w_{t-1} - u\|^2 - \|w_t - u\|^2$$
$$= (y_t - \hat{y}_t)^2\left(\frac{2}{\eta + \|x_t\|^2} - \frac{\|x_t\|^2}{\eta + \|x_t\|^2}\right) - \frac{2(y_t - \hat{y}_t)(y_t - \langle u, x_t\rangle)}{\|x_t\|^2 + \eta}$$
$$\geq \left(\frac{2}{\eta + \|x_t\|^2} - \frac{1}{(\eta + \|x_t\|^2)^2}\right)(y_t - \hat{y}_t)^2$$
$$- \frac{2(y_t - \hat{y}_t)(y_t - \langle u, x\rangle)}{\eta + \|x_t\|^2}$$
$$\geq \frac{1}{\eta + \|x_t\|^2}\left(\beta(y_t - \hat{y}_t)^2 - \frac{(y_t - \langle u, x\rangle)^2}{1-\beta}\right).$$

The last inequality holds due to Lemma 4. □

*Proof:* The proof of Theorem 1 is given as follows. From left-hand side of Lemma 4

$$\sum_{t=1}^{T}(\|w_t - u\|^2 - \|w_{t+1} - u\|^2) = \|w_1 - u\|^2 - \|w_{T+1} - u\|^2 \leq \|u\|^2.$$

Initialization of the weights is $\mathbf{0}$, and $\|\cdot\|$ is nonnegative; thus

$$\beta\sum_{t=1}^{T}\frac{(\hat{y}_t - y_t)^2}{\|x_t\|^2}\frac{\|x_t\|^2}{\eta + \|x_t\|^2}$$
$$- \sum_{t=1}^{T}\frac{(y_t - \langle u, x_t\rangle)^2}{\|x_t\|^2}\frac{\|x_t\|^2}{(1-\beta)(\eta + \|x_t\|^2)} \leq \|u\|^2. \tag{14}$$

Setting $(1/\|x_t\|) \leq (1/X)$ to get $L_T$

$$\frac{\beta}{(\eta + X^2)}L_T - \frac{L_T(u)}{(1-\beta)(\eta + X^2)} \leq \|u\|^2$$
$$\frac{\beta}{(\eta + X^2)}L_T \leq \|u\|^2 + \frac{L_T(u)}{(1-\beta)(\eta + X^2)}$$
$$L_T \leq \frac{(\eta + X^2)}{\beta}\|u\|^2 + \frac{L_T(u)}{(1-\beta)(\eta + X^2)}.$$

Thus

$$L_T \leq \frac{1}{\beta(1-\beta)}((\eta + X^2)\|u\|^2 + L_T(u)).$$

□

## References

[1] O. Akbilgic, H. Bozdogan, and M. E. Balaban, “A novel hybrid RBF neural networks model as a forecaster,” *Statist. Comput.*, vol. 24, no. 3, pp. 365–375, May 2014.

[2] N. Bershad, “Analysis of the normalized LMS algorithm with Gaussian inputs,” *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 34, no. 4, pp. 793–806, Aug. 1986.

[3] R. Bitmead and B. Anderson, “Performance of adaptive estimation algorithms in dependent random environments,” *IEEE Trans. Autom. Control*, vol. 25, no. 4, pp. 788–794, Aug. 1980.

[4] N. Budincsevity. (2016). *Weather in Szeged 2006-2016.* [Online]. Available: https://www.kaggle.com/budincsevity/szeged-weather#

[5] N. Cesa-Bianchi, P. M. Long, and M. K. Warmuth, “Worst-case quadratic loss bounds for prediction using linear functions and gradient descent,” *IEEE Trans. Neural Netw.*, vol. 7, no. 3, pp. 604–619, May 1996.

[6] K. Crammer, M. Dredze, and F. Pereira, “Exact convex confidence-weighted learning,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 345–352.

[7] R. C. de Lamare and R. Sampaio-Neto, “Reduced-rank adaptive filtering based on joint iterative optimization of adaptive filters,” *IEEE Signal Process. Lett.*, vol. 14, no. 12, pp. 980–983, Dec. 2007.

[8] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification and Scene Analysis*, 2nd ed. Hoboken, NJ, USA: Wiley, 1995.

[9] J. Friedman, T. Hastie, and R. Tibshirani, *The Elements of Statistical Learning*, vol. 1. New York, NY, USA: Springer, 2001.

[10] J. H. Friedman, “Multivariate adaptive regression splines,” *Annu. Statist.*, vol. 19, no. 1, pp. 1–67, Mar. 1991.

[11] D. H. Greene and D. E. Knuth, *Mathematics for the Analysis of Algorithms*. Basel, Switzerland: Birkhäuser, 1981.

[12] B. Hassibi, A. H. Sayed, and T. Kailath, “$H^\infty$ optimality of the LMS algorithm,” *IEEE Trans. Signal Process.*, vol. 44, no. 2, pp. 267–280, Feb. 1996.

[13] S. Haykin, *Adaptive Filter Theory*. Upper Saddle River, NJ, USA: Prentice-Hall, 1996.

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

[15] J. Kivinen and M. K. Warmuth, “Exponentiated gradient versus gradient descent for linear predictors,” *Inf. Comput.*, vol. 132, no. 1, pp. 1–63, Jan. 1997.

[16] H. Kuhn and A. Tucker, "Proceedings of the second Berkeley symposium on mathematical statistics and probability," in *Proc. 2nd Berkeley Symp. Math. Statist. Probab.* 1951, p. 666.

[17] J. McWhirter, "Recursive least-squares minimization using a systolic array," *Proc. SPIE*, vol. 431, pp. 105–114, Nov. 1983.

[18] E. Moroshko, N. Vaits, and K. Crammer, "Second-order non-stationary online learning for regression," *J. Mach. Learn. Res.*, vol. 16, pp. 1481–1517, Jan. 2015.

[19] F. Orabona, N. Cesa-Bianchi, and C. Gentile, "Beyond logarithmic bounds in online learning," in *Proc. 15th Int. Conf. Artif. Intell. Statist.*, 2012, pp. 823–831.

[20] J. Quinonero-Candela, I. Dagan, B. Magnini, and F. d'Alché Buc, *Machine Learning Challenges: Evaluating Predictive Uncertainty, Visual Object Classification, and Recognizing Textual Entailment, First Pascal Machine Learning Challenges Workshop, MLCW 2005, Southampton, UK, April 11–13, 2005, Revised Selected Papers*, vol. 3944. Berlin, Germany: Springer, 2006.

[21] 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

[22] D. D. Sleator and R. E. Tarjan, "Amortized efficiency of list update and paging rules," *Commun. ACM*, vol. 28, no. 2, pp. 202–208, Feb. 1985.

[23] J. N. Van Rijn *et al.*, "OpenML: A collaborative science platform," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, 2013, pp. 645–649.

[24] P. Vlachos and M. Meyer. (2005). *Statlib Datasets Archive*. [Online]. Available: http://lib.stat.cmu.edu/datasets

[25] V. Vovk, "Competitive on-line statistics," *Int. Stat. Rev.*, vol. 69, no. 2, pp. 213–248, Aug. 2001.

[26] B. Widrow and E. Walach, "On the statistical efficiency of the LMS algorithm with nonstationary inputs," *IEEE Trans. Inf. Theory*, vol. IT-30, no. 2, pp. 211–221, Mar. 1984.