Journal of Machine Learning Research 16 (2015) 1481-1517 Submitted 5/14; Revised 11/14; Published 8/15

Second-Order Non-Stationary Online Learning for Regression

Edward Moroshko EDWARD.MOROSHKO@GMAIL.COM
Nina Vaits NINAVAITS@GMAIL.COM
Koby Crammer KOBY@EE.TECHNION.AC.IL
Department of Electrical Engineering
The Technion - Israel Institute of Technology
Haifa 32000, Israel

Editor: Manfred Warmuth

Abstract

The goal of a learner in standard online learning, is to have the cumulative loss not much larger compared with the best-performing function from some fixed class. Numerous algorithms were shown to have this gap arbitrarily close to zero, compared with the best function that is chosen off-line. Nevertheless, many real-world applications, such as adaptive filtering, are non-stationary in nature, and the best prediction function may drift over time. We introduce two novel algorithms for online regression, designed to work well in non-stationary environment. Our first algorithm performs adaptive resets to forget the history, while the second is last-step min-max optimal in context of a drift. We analyze both algorithms in the worst-case regret framework and show that they maintain an average loss close to that of the best slowly changing sequence of linear functions, as long as the cumulative drift is sublinear. In addition, in the stationary case, when no drift occurs, our algorithms suffer logarithmic regret, as for previous algorithms. Our bounds improve over existing ones, and simulations demonstrate the usefulness of these algorithms compared with other state-of-the-art approaches.

Keywords: online learning, regret bounds, non-stationary input

1. Introduction

We consider the classical problem of online learning for regression. On each iteration, an algorithm receives a new instance (for example, input from an array of antennas) and outputs a prediction of a real value (for example distance to the source). The correct value is then revealed, and the algorithm suffers a loss based on both its prediction and the correct output value.

In the past half a century many algorithms were proposed (see e.g. a comprehensive book of Cesa-Bianchi and Lugosi 2006) for this problem, some of which are able to achieve an average loss arbitrarily close to that of the best function in retrospect. Furthermore, such guarantees hold even if the input and output pairs are chosen in a fully adversarial manner with no distributional assumptions. Many of these algorithms exploit first-order information (e.g. gradients).

Recently, there is an increased amount of interest in algorithms that exploit second-order information. For example the second-order perceptron algorithm (Cesa-Bianchi et al., 2005), confidence-weighted learning (Dredze et al., 2008; Crammer et al., 2008), adaptive regularization of weights (AROW) (Crammer et al., 2009), all designed for classification; and AdaGrad (Duchi et al., 2010) and FTPRL (McMahan and Streeter, 2010) for general loss functions.

Despite the extensive and impressive guarantees that can be made for algorithms in such settings, competing with the best *fixed* function is not always good enough. In many real-world applications, the true target function is not *fixed*, but is *slowly* changing over time. Consider a filter designed to cancel echoes in a hall. Over time, people enter and leave the hall, furniture are being moved, microphones are replaced and so on. When this drift occurs, the predictor itself must also change in order to remain relevant.

With such properties in mind, we develop new learning algorithms, based on second-order quantities, designed to work with target drift. The goal of an algorithm is to maintain an average loss close to that of the best slowly changing sequence of functions, rather than compete well with a single function. We focus on problems for which this sequence consists only of linear functions. Most previous algorithms (e.g. Littlestone and Warmuth 1994; Auer and Warmuth 2000; Herbster and Warmuth 2001; Kivinen et al. 2001) designed for this problem are based on first-order information, such as gradient descent, with additional control on the norm of the weight-vector used for prediction (Kivinen et al., 2001) or the number of inputs used to define it (Cavallanti et al., 2007).

In Section 2 we review three second-order learning algorithms: the recursive least squares (RLS) (Hayes, 1996) algorithm, the Aggregating Algorithm for regression (AAR) (Vovk, 1997, 2001), which can be shown to be derived based on a last-step min-max approach (Forster, 1999), and the AROWR algorithm (Vaits and Crammer, 2011) which is a modification of the AROW algorithm (Crammer et al., 2009) for regression. All three algorithms obtain logarithmic regret in the stationary setting, although derived using different approaches, and they are not equivalent in general.

In Section 3 we formally present the non-stationary setting both in terms of algorithms and in terms of theoretical analysis. For the RLS algorithm, a variant called CR-RLS (Salgado et al., 1988; Goodwin et al., 83; Chen and Yen, 1999) for the non-stationary setting was described, yet not analyzed, before. In Section 4 we present two new algorithms for the non-stationary setting, that build on the other two algorithms (AROWR and AAR). Specifically, in Section 4.1 we extend the AROWR algorithm to the non-stationary setting, yielding an algorithm called ARCOR for adaptive regularization with covariance reset. Similar to CR-RLS, ARCOR performs a step called covariance-reset, which resets the second-order information from time-to-time, yet it is done based on the properties of this covariance-like matrix, and not based on the number of examples observed, as in CR-RLS.

In Section 4.2 we derive a different algorithm based on the last-step min-max approach proposed by Forster (1999) and later used (Takimoto and Warmuth, 2000) for online density estimation. On each iteration the algorithm makes the optimal min-max prediction with respect to the regret, assuming it is the last iteration. Yet, unlike previous work (Forster, 1999), it is optimal when a drift is allowed. As opposed to the derivation of the last-step min-max predictor for a fixed vector, the resulting optimization problem is not straightforward to solve. We develop a dynamic program (a recursion) to solve this problem, which allows to compute the optimal last-step min-max predictor. We call this algorithm LASER for last step adaptive regressor algorithm. We conclude the algorithmic part in Section 4.3

in which we compare all non-stationary algorithms head-to-head highlighting their similarities and differences. Additionally, after describing the details of our algorithms, we provide in Section 5 a comprehensive review of previous work, that puts our contribution in perspective. Both algorithms reduce to their stationary counterparts when no drift occurs.

We then move to Section 6 which summarizes our next contribution stating and proving regret bounds for both algorithms. We analyze both algorithms in the worst-case regret-setting and show that as long as the amount of average-drift is sublinear, the average-loss of both algorithms will converge to the average-loss of the best sequence of functions. Specifically, we show in Section 6.1 that the cumulative loss of ARCOR after observing T examples, denoted by $L_T(\text{ARCOR})$, is upper bounded by the cumulative loss of any *sequence* of weight-vectors $\{\boldsymbol{u}_t\}$, denoted by $L_T(\{\boldsymbol{u}_t\})$, plus an additional term $\mathcal{O}\left(T^{1/2}\left(V(\{\boldsymbol{u}_t\})\right)^{1/2}\log T\right)$ where $V(\{\boldsymbol{u}_t\})$ measures the differences (or variance) between consecutive weight-vectors of the sequence $\{\boldsymbol{u}_t\}$. Later, we show in Section 6.2 a similar bound for the loss of LASER, denoted by $L_T(\text{LASER})$, for which the second term is $\mathcal{O}\left(T^{2/3}\left(V(\{\boldsymbol{u}_t\})\right)^{1/3}\right)$. We emphasize that in both bounds the measure $V(\{\boldsymbol{u}_t\})$ of differences between consecutive weight-vectors is not defined in the same way, and thus, the bounds are not comparable in general.

In Section 7 we report results of simulations designed to highlight the properties of both algorithms, as well as the commonalities and differences between them. We conclude in Section 8 and most of the technical proofs appear in the appendix.

The ARCOR algorithm was presented in a shorter publication (Vaits and Crammer, 2011), as well with its analysis and some of its details. The LASER algorithm and its analysis was also presented in a shorter version (Moroshko and Crammer, 2013). The contribution of this submission is three-fold. First, we provide head-to-head comparison of three second-order algorithms for the stationary case. Second, we fill the gap of second-order algorithms for the non-stationary case. Specifically, we add to the CR-RLS (which extends RLS) and design second-order algorithms for the non-stationary case and analyze them, building both on AROWR and AAR. Our algorithms are derived from different principles, which is reflected in our analysis. Finally, we provide empirical evidence showing that under various conditions different algorithm performs the best.

Some notation we use throughout the paper: For a symmetric matrix Σ we denote its jth eigenvalue by $\lambda_j(\Sigma)$. Similarly we denote its smallest eigenvalue by $\lambda_{min}(\Sigma) = \min_j \lambda_j(\Sigma)$, and its largest eigenvalue by $\lambda_{max}(\Sigma) = \max_j \lambda_j(\Sigma)$. For a vector $\boldsymbol{u} \in \mathbb{R}^d$, we denote by $\|\boldsymbol{u}\|$ the ℓ_2-norm of the vector. Finally, for $y > 0$ we define $clip(x, y) = \text{sign}(x)\min\{|x|, y\}$.

2. Stationary Online Learning

We focus on the online regression task evaluated with the squared loss, where algorithms work in iterations (or rounds). On each round an online algorithm receives an input-vector $\boldsymbol{x}_t \in \mathbb{R}^d$ and predicts a real value $\hat{y}_t \in \mathbb{R}$. Then the algorithm receives a target label $y_t \in \mathbb{R}$ associated with $\boldsymbol{x}_t$, uses it to update its prediction rule, and proceeds to the next round.

On each iteration, the performance of the algorithm is evaluated using the squared loss, $\ell_t(\text{alg}) = \ell\left(y_t, \hat{y}_t\right) = \left(\hat{y}_t - y_t\right)^2$. The cumulative loss suffered by the algorithm over T iterations is, $L_T(\text{alg}) = \sum_{t=1}^{T} \ell_t(\text{alg})$.

The goal of the algorithm is to have low cumulative loss compared to predictors from some class. A large body of work, which we adopt as well, is focused on linear prediction functions of the form $f(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{u}$ where $\boldsymbol{u} \in \mathbb{R}^d$ is some weight-vector. We denote by $\ell_t(\boldsymbol{u}) = \left(\boldsymbol{x}_t^\top \boldsymbol{u} - y_t\right)^2$ the instantaneous loss of a weight-vector $\boldsymbol{u}$. The cumulative loss suffered by a fixed weight-vector $\boldsymbol{u}$ is, $L_T(\boldsymbol{u}) = \sum_{t=1}^T \ell_t(\boldsymbol{u})$.

The goal of the learning algorithm is to suffer low loss compared with the best linear function. Formally we define the regret of an algorithm to be

$$R(T) = L_T(\text{alg}) - \inf_{\boldsymbol{u}} L_T(\boldsymbol{u}) \ .$$

The goal of an algorithm is to have $R(T) = o(T)$, such that the average loss of the algorithm will converge to the average loss of the best linear function $\boldsymbol{u}$.

Numerous algorithms were developed for this problem, see a comprehensive review in the book of Cesa-Bianchi and Lugosi (2006). Among these, a few second-order online algorithms for regression were proposed in recent years, which we summarize in Table 1. One approach for online learning is to reduce the problem into consecutive batch problems, and specifically use all previous examples to generate a regressor, which is used to predict the current example. The least squares approach, for example, sets a weight-vector to be the solution of the following optimization problem

$$\boldsymbol{w}_t = \arg\min_{\boldsymbol{w}} \left(\sum_{i=1}^{t} r^{t-i} \left(y_i - \boldsymbol{w} \cdot \boldsymbol{x}_i\right)^2 \right) \ ,$$

for $0 < r \leq 1$. Since the last problem grows with time, the well known recursive least squares (RLS) (Hayes, 1996) algorithm was developed to generate a solution recursively. The RLS algorithm maintains both a vector $\boldsymbol{w}_t$ and a positive semi-definite (PSD) matrix Σ_t. On each iteration, after making a prediction $\hat{y}_t = \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}$, the algorithm receives the true label y_t and updates

$$\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \frac{(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1})\Sigma_{t-1}\boldsymbol{x}_t}{r + \boldsymbol{x}_t^\top \Sigma_{t-1} \boldsymbol{x}_t} \tag{1}$$

$$\Sigma_t^{-1} = r\Sigma_{t-1}^{-1} + \boldsymbol{x}_t \boldsymbol{x}_t^\top \ . \tag{2}$$

The update of the prediction vector $\boldsymbol{w}_t$ is additive, with vector $\Sigma_{t-1}\boldsymbol{x}_t$ scaled by the error $(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1})$ over the norm of the input measured using the norm defined by the matrix $\boldsymbol{x}_t^\top \Sigma_{t-1} \boldsymbol{x}_t$. The algorithm is summarized in the right column of Table 1.

The Aggregating Algorithm for regression (AAR) (Vovk, 1997; Azoury and Warmuth, 2001), summarized in the middle column of Table 1, was introduced by Vovk and it is similar to the RLS algorithm, except it shrinks its predictions. The AAR algorithm was shown to be last-step min-max optimal by Forster (1999). Given a new input $\boldsymbol{x}_T$ the algorithm predicts $\hat{y}_T$ which is the minimizer of the following problem

$$\arg\min_{\hat{y}_T} \max_{y_T} \left[\sum_{t=1}^{T} (y_t - \hat{y}_t)^2 - \inf_{\boldsymbol{u}} \left(b \left\| \boldsymbol{u} \right\|^2 + L_T(\boldsymbol{u}) \right) \right] . \tag{3}$$

Forster proposed also a simpler analysis with the same regret bound as of Vovk (1997).

The AROWR algorithm (Vaits and Crammer, 2011) is a modification of the AROW algorithm (Crammer et al., 2009) for regression. In a nutshell, the AROW algorithm maintains a Gaussian distribution parameterized by a mean $\boldsymbol{w}_t \in \mathbb{R}^d$ and a full covariance matrix $\Sigma_t \in \mathbb{R}^{d\times d}$. Intuitively, the mean $\boldsymbol{w}_t$ represents a current linear function, while the covariance matrix Σ_t captures the uncertainty in the linear function $\boldsymbol{w}_t$. Given a new input $\boldsymbol{x}_t$ the algorithm uses its current mean to make a prediction $\hat{y}_t = \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}$. Then, given the true label y_t, AROWR sets the new distribution to be the solution of the following optimization problem

$$\arg\min_{\boldsymbol{w},\Sigma} \left[\mathrm{D}_{\mathrm{KL}}[\mathcal{N}(\boldsymbol{w},\Sigma) \,\|\, \mathcal{N}(\boldsymbol{w}_{t-1},\Sigma_{t-1})] + \frac{1}{2r}\left(y_t - \boldsymbol{w}^\top \boldsymbol{x}_t\right)^2 + \frac{1}{2r}\left(\boldsymbol{x}_t^\top \Sigma \boldsymbol{x}_t\right)\right] , \qquad (4)$$

for $r > 0$. This optimization problem is similar to the one of AROW (Crammer et al., 2009) for classification, except we use the square loss rather than squared-hinge loss used in AROW. Intuitively, the optimization problem trades off between three requirements. The first term forces the parameters not to change much per example, as the entire learning history is encapsulated within them. The second term requires that the new vector $\boldsymbol{w}_t$ should perform well on the current instance, and finally, the third term reflects the fact that the uncertainty about the parameters reduces as we observe the current example $\boldsymbol{x}_t$.

The weight vector solving this optimization problem (details given by Vaits and Crammer 2011) is given by

$$\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \left(\frac{y_t - \boldsymbol{w}_{t-1}\cdot \boldsymbol{x}_t}{r + \boldsymbol{x}_t^\top \Sigma_{t-1}\boldsymbol{x}_t}\right)\Sigma_{t-1}\boldsymbol{x}_t , \qquad (5)$$

and the optimal covariance matrix is

$$\Sigma_t^{-1} = \Sigma_{t-1}^{-1} + \frac{1}{r}\boldsymbol{x}_t\boldsymbol{x}_t^\top . \qquad (6)$$

The algorithm is summarized in the left column of Table 1. Comparing AROWR to RLS we observe that while the update of the weights of (5) is equivalent to the update of RLS in (1), the update of the matrix (2) for RLS is not equivalent to (6), as in the former case the matrix goes via a multiplicative update as well as additive, while in (6) the update is only additive. The two updates are equivalent only by setting $r = 1$. Moving to AAR, we note that the update rules for $\boldsymbol{w}_t$ and Σ_t in AROWR and AAR are the same if we define $\Sigma_t^{AAR} = \Sigma_t^{AROWR}/r$, but AROWR does not shrink its predictions as AAR. Thus all three algorithms are not equivalent, although very similar.

3. Non-Stationary Online Learning

The analysis of all algorithms discussed above compares their performance to that of a single fixed weight vector $\boldsymbol{u}$, and all suffer regret that is logarithmic in T. However, in many real-world applications, the true target function is not fixed, but is slowly changing over time.

We use an extended notion of evaluation, comparing our algorithms to a sequence of functions. We define the loss suffered by such a sequence to be

$$L_T(\boldsymbol{u}_1,\ldots,\boldsymbol{u}_T) = L_T(\{\boldsymbol{u}_t\}) = \sum_{t=1}^{T} \ell_t(\boldsymbol{u}_t) ,$$

and the tracking regret is then defined to be

$$R(T) = L_T(\text{alg}) - \inf_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_T} L_T(\{\boldsymbol{u}_t\}) \ .$$

We focus on algorithms that are able to compete against sequences of weight-vectors, $(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_T) \in \mathbb{R}^d \times \cdots \times \mathbb{R}^d$, where $\boldsymbol{u}_t$ is used to make a prediction for the tth example $(\boldsymbol{x}_t, y_t)$. Note the difference between tracking regret (where the algorithm is compared to a good sequence of experts, as we do) and adaptive regret (Adamskiy et al., 2012), which measures how well the algorithm approximates the best expert locally on some time interval.

Clearly, with no restriction over the set $\{\boldsymbol{u}_t\}$ the right term of the regret can easily be zero by setting, $\boldsymbol{u}_t = \boldsymbol{x}_t(y_t/\left\|\boldsymbol{x}_t\right\|^2)$, which implies $\ell_t(\boldsymbol{u}_t) = 0$ for all t. Thus, in the analysis below we will make use of the total drift of the weight-vectors defined to be

$$V^{(P)} = V_T^{(P)}(\{\boldsymbol{u}_t\}) = \sum_{t=1}^{T-1} \left\|\boldsymbol{u}_t - \boldsymbol{u}_{t+1}\right\|^P \ ,$$

where $P \in \{1, 2\}$, and the total loss of the algorithm is allowed to scale with the total drift.

For the three algorithms in Table 1 the matrix Σ can be interpreted as adaptive learning rate, as was also observed previously in the context of CW (Dredze et al., 2008), AROW (Crammer et al., 2009), AdaGrad (Duchi et al., 2010) and FTPRL (McMahan and Streeter, 2010). As these algorithms process more examples, that is larger values of t, the eigenvalues of the matrix Σ_t^{-1} increase, the eigenvalues of the matrix Σ_t decrease, and we get that the rate of updates is getting smaller, since the additive term $\Sigma_{t-1}\boldsymbol{x}_t$ is getting smaller. As a consequence the algorithms will gradually stop updating using current instances which lie in the subspace of examples that were previously observed numerous times. This property leads to a very fast convergence in the stationary case. However, when we allow these algorithms to be compared with a sequence of weight-vectors, each applied to a different input example, or equivalently, there is a drift or shift of a good prediction vector, these algorithms will perform poorly, as they will converge and will not be able to adapt to the non-stationarity nature of the data.

This phenomena motivated the proposal of the CR-RLS algorithm (Salgado et al., 1988; Goodwin et al., 83; Chen and Yen, 1999), which re-sets the covariance matrix every fixed number of input examples, causing the algorithm not to converge or get stuck. The pseudocode of CR-RLS algorithm is given in the right column of Table 2. The only difference of CR-RLS from RLS is that after updating the matrix Σ_t, the algorithm checks whether T_0 (a predefined natural number) examples were observed since the last restart, and if this is the case, it sets the matrix to be the identity matrix. Clearly, if $T_0 = \infty$ the CR-RLS algorithm is reduced to the RLS algorithm.

4. Algorithms for Non-Stationary Regression

In this work we fill the gap and propose extension to non-stationary setting for the two other algorithms in Table 1. Similar to CR-RLS, both algorithms modify the matrix Σ_t to prevent its eigenvalues to shrink to zero. The first algorithm, described in Section 4.1,

		AROWR	**AAR**	**RLS**
Parameters		$0 < r$	$0 < b$	$0 < r \leq 1$
Initialize		$\boldsymbol{w}_0 = 0 \ , \ \Sigma_0 = I$	$\boldsymbol{w}_0 = 0 \ , \ \Sigma_0 = b^{-1} I$	$\boldsymbol{w}_0 = 0 \ , \ \Sigma_0 = I$
for $t = 1 ... T$		Receive an instance $\boldsymbol{x}_t$		
	Output prediction	$\hat{y}_t = \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}$	$\hat{y}_t = \frac{\boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}}{1 + \boldsymbol{x}_t^\top \Sigma_{t-1} \boldsymbol{x}_t}$	$\hat{y}_t = \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}$
		Receive a correct label y_t		
	Update Σ_t	$\Sigma_t^{-1} = \Sigma_{t-1}^{-1} + \frac{1}{r} \boldsymbol{x}_t \boldsymbol{x}_t^\top$	$\Sigma_t^{-1} = \Sigma_{t-1}^{-1} + \boldsymbol{x}_t \boldsymbol{x}_t^\top$	$\Sigma_t^{-1} = r \Sigma_{t-1}^{-1} + \boldsymbol{x}_t \boldsymbol{x}_t^\top$
	Update $\boldsymbol{w}_t$	$\boldsymbol{w}_t = \boldsymbol{w}_{t-1}$ $+ \frac{(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}) \Sigma_{t-1} \boldsymbol{x}_t}{r + \boldsymbol{x}_t^\top \Sigma_{t-1} \boldsymbol{x}_t}$	$\boldsymbol{w}_t = \boldsymbol{w}_{t-1}$ $+ \frac{(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}) \Sigma_{t-1} \boldsymbol{x}_t}{1 + \boldsymbol{x}_t^\top \Sigma_{t-1} \boldsymbol{x}_t}$	$\boldsymbol{w}_t = \boldsymbol{w}_{t-1}$ $+ \frac{(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}) \Sigma_{t-1} \boldsymbol{x}_t}{r + \boldsymbol{x}_t^\top \Sigma_{t-1} \boldsymbol{x}_t}$
Output		$\boldsymbol{w}_T \ , \ \Sigma_T$	$\boldsymbol{w}_T \ , \ \Sigma_T$	$\boldsymbol{w}_T \ , \ \Sigma_T$
Extension to non-stationary setting		**ARCOR** Section 4.1 below	**LASER** Section 4.2 below	**CR-RLS** (Goodwin et al., 83)
Analysis		yes, Section 6.1 below	yes, Section 6.2 below	No

Table 1: Algorithms for stationary setting and their extension to non-stationary case

extends AROWR to the non-stationary setting and is similar in spirit to CR-RLS, yet the restart operations it performs depend on the spectral properties of the covariance matrix, rather than the time index t. Additionally, this algorithm performs a projection of the weight vector into a predefined ball. Similar technique was used in first order algorithms by Herbster and Warmuth (2001), and Kivinen and Warmuth (1997). Both steps are motivated from the design and analysis of AROWR. Its design is composed of solving small optimization problems defined in (4), one per input example. The non-stationary version performs explicit corrections to its update, in order to prevent from the covariance matrix to shrink to zero, and the weight-vector to grow too fast.

The second algorithm, described in Section 4.2, is based on a last-step min-max prediction principle and objective, where we replace $L_T(\boldsymbol{u})$ in (3) with $L_T(\{\boldsymbol{u}_t\})$ and some additional modifications preventing the solution being degenerate. Here the algorithmic modifications from the original AAR algorithm are implicit and are due to the modifications of the objective. The resulting algorithm smoothly interpolates the covariance matrix with the identity matrix.

4.1 ARCOR: Adaptive Regularization of Weights for Regression with Covariance Reset

Our first algorithm is based on the AROWR. We propose two modifications to (5) and (6), which in combination overcome the problem that the algorithm's learning rate gradually goes to zero. The modified algorithm operates on segments of the input sequence. In each segment indexed by i, the algorithm checks whether the lowest eigenvalue of Σ_t is greater than a given lower bound Λ_i. Once the lowest eigenvalue of Σ_t is smaller than Λ_i the algorithm resets $\Sigma_t = I$ and updates the value of the lower bound Λ_{i+1}. Formally, the algorithm uses the update (6) to compute an intermediate candidate for Σ_t, denoted by

$$\tilde{\Sigma}_t = \left(\Sigma_{t-1}^{-1} + \frac{1}{r}\boldsymbol{x}_t\boldsymbol{x}_t^\top\right)^{-1} . \tag{7}$$

If indeed $\tilde{\Sigma}_t \succeq \Lambda_i I$ then it sets $\Sigma_t = \tilde{\Sigma}_t$, otherwise it sets $\Sigma_t = I$ and the segment index is increased by 1.

Additionally, before our modification, the norm of the weight vector $\boldsymbol{w}_t$ did not increase much as the effective learning rate (the matrix Σ_t) went to zero. After our update, as the learning rate is effectively bounded from below, the norm of $\boldsymbol{w}_t$ may increase too fast, which in turn will cause a low update-rate in non-stationary inputs.

We thus employ additional modification which is exploited by the analysis. After updating the mean $\boldsymbol{w}_t$ as in (5),

$$\tilde{\boldsymbol{w}}_t = \boldsymbol{w}_{t-1} + \frac{(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1})\Sigma_{t-1}\boldsymbol{x}_t}{r + \boldsymbol{x}_t^\top \Sigma_{t-1}\boldsymbol{x}_t} , \tag{8}$$

we project it into a ball B around the origin of radius R_B using a Mahalanobis distance. Formally, we define the function $\mathrm{proj}(\tilde{\boldsymbol{w}}, \Sigma, R_B)$ to be the solution of the following optimization problem

$$\arg \min_{\|\boldsymbol{w}\| \leq R_B} \frac{1}{2} \left(\boldsymbol{w} - \tilde{\boldsymbol{w}}\right)^\top \Sigma^{-1} \left(\boldsymbol{w} - \tilde{\boldsymbol{w}}\right) .$$

We write the Lagrangian,

$$\mathcal{L} = \frac{1}{2} \left(\boldsymbol{w} - \tilde{\boldsymbol{w}}\right)^\top \Sigma^{-1} \left(\boldsymbol{w} - \tilde{\boldsymbol{w}}\right) + \alpha \left(\frac{1}{2}\|\boldsymbol{w}\|^2 - \frac{1}{2}R_B^2\right) .$$

Setting the gradient with respect to $\boldsymbol{w}$ to zero we get, $\Sigma^{-1}\left(\boldsymbol{w} - \tilde{\boldsymbol{w}}\right) + \alpha\boldsymbol{w} = 0$. Solving for $\boldsymbol{w}$ we get

$$\boldsymbol{w} = \left(\alpha I + \Sigma^{-1}\right)^{-1} \Sigma^{-1}\tilde{\boldsymbol{w}} = \left(I + \alpha\Sigma\right)^{-1} \tilde{\boldsymbol{w}} .$$

From KKT conditions we get that if $\|\tilde{\boldsymbol{w}}\| \leq R_B$ then $\alpha = 0$ and $\boldsymbol{w} = \tilde{\boldsymbol{w}}$. Otherwise, α is the unique positive scalar that satisfies $\|\left(I + \alpha\Sigma\right)^{-1}\tilde{\boldsymbol{w}}\| = R_B$. The value of α can be found using binary search and eigen-decomposition of the matrix Σ. We write explicitly $\Sigma = V\Lambda V^\top$ for a diagonal matrix Λ. By denoting $\boldsymbol{u} = V^\top\tilde{\boldsymbol{w}}$ we rewrite the last equation, $\|\left(I + \alpha\Lambda\right)^{-1}\boldsymbol{u}\| = R_B$. We thus wish to find α such that $\sum_j^d \frac{u_j^2}{(1+\alpha\Lambda_{j,j})^2} = R_B^2$. It can be done using a binary search for $\alpha \in [0, a]$ where $a = (\|\boldsymbol{u}\|/R_B - 1)/\lambda_{\min}(\Lambda)$. To summarize, the projection step can be performed in time cubic in d and logarithmic in R_B and Λ_i.

We call the algorithm ARCOR for adaptive regularization with covariance reset. A pseudo-code of the algorithm is summarized in the left column of Table 2. We defer a comparison of ARCOR and CR-RLS after the presentation of our second algorithm now.

		ARCOR	**LASER**	**CR-RLS**
Parame-ters		$0 < r, R_B$, a sequence $1 > \Lambda_1 \geq \Lambda_2 ...$	$0 < b < c$	$0 < r \leq 1, T_0 \in \mathbb{N}$
Initialize		$\boldsymbol{w}_0 = 0$, $\Sigma_0 = I$, $i = 1$	$\boldsymbol{w}_0 = 0$, $\Sigma_0 = \frac{c-b}{bc} I$	$\boldsymbol{w}_0 = 0$, $\Sigma_0 = I$
for $t = 1...T$		Receive an instance $\boldsymbol{x}_t$		
	Output prediction	$\hat{y}_t = \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}$	$\hat{y}_t = \frac{\boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}}{1 + \boldsymbol{x}_t^\top \left(\Sigma_{t-1} + c^{-1} I\right) \boldsymbol{x}_t}$	$\hat{y}_t = \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}$
		Receive a correct label y_t		
	Update Σ_t	$\tilde{\Sigma}_t^{-1} = \Sigma_{t-1}^{-1} + \frac{1}{r} \boldsymbol{x}_t \boldsymbol{x}_t^\top$ If $\tilde{\Sigma}_t \succeq \Lambda_i I$ set $\Sigma_t = \tilde{\Sigma}_t$ else set $\Sigma_t = I$, $i = i + 1$	$\Sigma_t^{-1} = \left(\Sigma_{t-1} + c^{-1} I\right)^{-1} + \boldsymbol{x}_t \boldsymbol{x}_t^\top$	$\tilde{\Sigma}_t^{-1} = r \Sigma_{t-1}^{-1} + \boldsymbol{x}_t \boldsymbol{x}_t^\top$ If $\mod(t, T_0) > 0$ set $\Sigma_t = \tilde{\Sigma}_t$ else set $\Sigma_t = I$
	Update $\boldsymbol{w}_t$	$\tilde{\boldsymbol{w}}_t = \boldsymbol{w}_{t-1} + \frac{(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}) \Sigma_{t-1} \boldsymbol{x}_t}{r + \boldsymbol{x}_t^\top \Sigma_{t-1} \boldsymbol{x}_t}$ $\boldsymbol{w}_t = \text{proj}\left(\tilde{\boldsymbol{w}}_t, \Sigma_t, R_B\right)$	$\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \frac{(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}) \left(\Sigma_{t-1} + c^{-1} I\right) \boldsymbol{x}_t}{1 + \boldsymbol{x}_t^\top \left(\Sigma_{t-1} + c^{-1} I\right) \boldsymbol{x}_t}$	$\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \frac{(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}) \Sigma_{t-1} \boldsymbol{x}_t}{r + \boldsymbol{x}_t^\top \Sigma_{t-1} \boldsymbol{x}_t}$
Output		$\boldsymbol{w}_T$, Σ_T	$\boldsymbol{w}_T$, Σ_T	$\boldsymbol{w}_T$, Σ_T

Table 2: ARCOR, LASER and CR-RLS algorithms

4.2 Last-Step Min-Max Algorithm for Non-Stationary Setting

Our second algorithm is based on a last-step min-max predictor proposed by Forster (1999) and later modified by Moroshko and Crammer (2012) to obtain sub-logarithmic regret in the stationary case. On each round, the algorithm predicts as in the last round, and assumes a worst case choice of y_t given the algorithm's prediction.

We extend the rule given in (3) to the non-stationary setting, and re-define the last-step minmax predictor $\hat{y}_T$ to be[1]

$$\arg \min_{\hat{y}_T} \max_{y_T} \left[\sum_{t=1}^{T} \left(y_t - \hat{y}_t\right)^2 - \min_{\boldsymbol{u}_1,..,\boldsymbol{u}_T} Q_T\left(\boldsymbol{u}_1, ..., \boldsymbol{u}_T\right) \right] , \tag{9}$$

where

$$Q_t\left(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_t\right) = b \left\|\boldsymbol{u}_1\right\|^2 + c \sum_{s=1}^{t-1} \left\|\boldsymbol{u}_{s+1} - \boldsymbol{u}_s\right\|^2 + \sum_{s=1}^{t} \left(y_s - \boldsymbol{u}_s^\top \boldsymbol{x}_s\right)^2 , \tag{10}$$

for some positive constants b, c. The first term of (9) is the loss suffered by the algorithm while $Q_t\left(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_t\right)$ defined in (10) is a sum of the loss suffered by some sequence of linear functions $\left(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_t\right)$, and a penalty for consecutive pairs that are far from each other, and for the norm of the first to be far from zero.

[1] y_T and $\hat{y}_T$ serve both as quantifiers (over the max and min operators, respectively), and as the optimal arguments of this optimization problem.

We develop the algorithm by solving the three optimization problems in (9), first, minimizing the inner term, $\min_{\boldsymbol{u}_1,..,\boldsymbol{u}_T} Q_T\left(\boldsymbol{u}_1, ..., \boldsymbol{u}_T\right)$, maximizing over $\boldsymbol{y}_T$, and finally, minimizing over $\hat{y}_T$. We start with the inner term for which we define an auxiliary function

$$P_t\left(\boldsymbol{u}_t\right) = \min_{\boldsymbol{u}_1,\dots,\boldsymbol{u}_{t-1}} Q_t\left(\boldsymbol{u}_1, \dots, \boldsymbol{u}_t\right) \ ,$$

which clearly satisfies

$$\min_{\boldsymbol{u}_1,\dots,\boldsymbol{u}_t} Q_t\left(\boldsymbol{u}_1, \dots, \boldsymbol{u}_t\right) = \min_{\boldsymbol{u}_t} P_t(\boldsymbol{u}_t) \ .$$

The following lemma states a recursive form of the function-sequence $P_t(\boldsymbol{u}_t)$.

Lemma 1 *For $t = 2, 3, \dots$*

$$\begin{aligned} &P_1(\boldsymbol{u}_1) = Q_1(\boldsymbol{u}_1) \\ &P_t\left(\boldsymbol{u}_t\right) = \min_{\boldsymbol{u}_{t-1}} \left(P_{t-1}\left(\boldsymbol{u}_{t-1}\right) + c\left\|\boldsymbol{u}_t - \boldsymbol{u}_{t-1}\right\|^2 + \left(y_t - \boldsymbol{u}_t^\top \boldsymbol{x}_t\right)^2 \right) . \end{aligned}$$

The proof appears in Appendix A. Using Lemma 1 we write explicitly the function $P_t(\boldsymbol{u}_t)$.

Lemma 2 *The following equality holds*

$$P_t\left(\boldsymbol{u}_t\right) = \boldsymbol{u}_t^\top D_t \boldsymbol{u}_t - 2\boldsymbol{u}_t^\top \boldsymbol{e}_t + f_t \ ,$$

where

$$\begin{aligned} D_1 &= bI + \boldsymbol{x}_1\boldsymbol{x}_1^\top & D_t &= \left(D_{t-1}^{-1} + c^{-1}I\right)^{-1} + \boldsymbol{x}_t\boldsymbol{x}_t^\top & (11) \\ \boldsymbol{e}_1 &= y_1\boldsymbol{x}_1 & \boldsymbol{e}_t &= \left(I + c^{-1}D_{t-1}\right)^{-1} \boldsymbol{e}_{t-1} + y_t\boldsymbol{x}_t & (12) \\ f_1 &= y_1^2 & f_t &= f_{t-1} - \boldsymbol{e}_{t-1}^\top \left(cI + D_{t-1}\right)^{-1} \boldsymbol{e}_{t-1} + y_t^2 \ . & (13) \end{aligned}$$

Note that $D_t \in \mathbb{R}^{d\times d}$ is a positive definite matrix, $\boldsymbol{e}_t \in \mathbb{R}^{d\times 1}$ and $f_t \in \mathbb{R}$. The proof appears in Appendix B. From Lemma 2 we conclude that

$$\min_{\boldsymbol{u}_1,\dots,\boldsymbol{u}_t} Q_t\left(\boldsymbol{u}_1, \dots, \boldsymbol{u}_t\right) = \min_{\boldsymbol{u}_t} P_t\left(\boldsymbol{u}_t\right) = \min_{\boldsymbol{u}_t} \left(\boldsymbol{u}_t^\top D_t \boldsymbol{u}_t - 2\boldsymbol{u}_t^\top \boldsymbol{e}_t + f_t\right) = -\boldsymbol{e}_t^\top D_t^{-1} \boldsymbol{e}_t + f_t \ . \tag{14}$$

Substituting (14) back in (9) we get that the last-step minmax predictor is given by

$$\hat{y}_T = \arg\min_{\hat{y}_T} \max_{y_T} \left[\sum_{t=1}^{T} \left(y_t - \hat{y}_t\right)^2 + \boldsymbol{e}_T^\top D_T^{-1} \boldsymbol{e}_T - f_T \right] \ . \tag{15}$$

Since $\boldsymbol{e}_T$ depends on y_T we substitute (12) in the second term of (15),

$$\boldsymbol{e}_T^\top D_T^{-1} \boldsymbol{e}_T = \left(\left(I + c^{-1}D_{T-1}\right)^{-1} \boldsymbol{e}_{T-1} + y_T\boldsymbol{x}_T\right)^\top D_T^{-1} \left(\left(I + c^{-1}D_{T-1}\right)^{-1} \boldsymbol{e}_{T-1} + y_T\boldsymbol{x}_T\right) \ . \tag{16}$$

Substituting (16) and (13) in (15) and omitting terms not depending explicitly on y_T and $\hat{y}_T$ we get

$$\begin{aligned}\hat{y}_T &= \arg\min_{\hat{y}_T}\max_{y_T}\left[(y_T-\hat{y}_T)^2 + y_T^2\boldsymbol{x}_T^\top D_T^{-1}\boldsymbol{x}_T + 2y_T\boldsymbol{x}_T^\top D_T^{-1}\left(I+c^{-1}D_{T-1}\right)^{-1}\boldsymbol{e}_{T-1} - y_T^2\right]\\ &= \arg\min_{\hat{y}_T}\max_{y_T}\left[\left(\boldsymbol{x}_T^\top D_T^{-1}\boldsymbol{x}_T\right)y_T^2 + 2y_T\left(\boldsymbol{x}_T^\top D_T^{-1}\left(I+c^{-1}D_{T-1}\right)^{-1}\boldsymbol{e}_{T-1}-\hat{y}_T\right)+\hat{y}_T^2\right] . \end{aligned} \tag{17}$$

The last equation is strictly convex in y_T and thus the optimal solution is not bounded. To solve it, we follow an approach used by Forster (1999) in a different context. In order to make the optimal value bounded, we assume that the adversary can only choose labels from a bounded set $y_T \in [-Y, Y]$. Thus, the optimal solution of (17) over y_T is given by the following equation, since the optimal value is $y_T \in \{+Y, -Y\}$,

$$\hat{y}_T = \arg\min_{\hat{y}_T}\left[\left(\boldsymbol{x}_T^\top D_T^{-1}\boldsymbol{x}_T\right)Y^2 + 2Y\left|\boldsymbol{x}_T^\top D_T^{-1}\left(I+c^{-1}D_{T-1}\right)^{-1}\boldsymbol{e}_{T-1}-\hat{y}_T\right|+\hat{y}_T^2\right] .$$

This problem is of a similar form to the one discussed by Forster (1999), from which we get the optimal solution, $\hat{y}_T = clip\left(\boldsymbol{x}_T^\top D_T^{-1}\left(I+c^{-1}D_{T-1}\right)^{-1}\boldsymbol{e}_{T-1}, Y\right)$.

The optimal solution depends explicitly on the bound Y, and as its value is not known, we thus ignore it, and define the output of the algorithm to be

$$\hat{y}_T = \boldsymbol{x}_T^\top D_T^{-1}\left(I+c^{-1}D_{T-1}\right)^{-1}\boldsymbol{e}_{T-1} = \boldsymbol{x}_T^\top D_T^{-1}D'_{T-1}\boldsymbol{e}_{T-1} , \tag{18}$$

where we define

$$D'_{t-1} = \left(I+c^{-1}D_{t-1}\right)^{-1} . \tag{19}$$

We call the algorithm LASER for last step adaptive regressor algorithm. Clearly, for $c = \infty$ the LASER algorithm reduces to the AAR algorithm. Similar to CR-RLS and ARCOR, this algorithm can be also expressed in terms of weight-vector $\boldsymbol{w}_t$ and a PSD matrix Σ_t, by denoting $\boldsymbol{w}_t = D_t^{-1}\boldsymbol{e}_t$ and $\Sigma_t = D_t^{-1}$. The algorithm is summarized in the middle column of Table 2.

4.3 Discussion

Table 2 enables us to compare the three algorithms head-to-head. All algorithms perform predictions, and then update the prediction vector $\boldsymbol{w}_t$ and the matrix Σ_t. CR-RLS and ARCOR are more similar to each other, both stem from a stationary algorithm, and perform resets from time-to-time. For CR-RLS it is performed every fixed time steps, while for ARCOR it is performed when the eigenvalues of the matrix (or effective learning rate) are too small. ARCOR also performs a projection step, which is motivated to ensure that the weight-vector will not grow to much, and is used explicitly in the analysis below. Note that CR-RLS (as well as RLS) also uses a forgetting factor (if $r < 1$).

Our second algorithm, LASER, controls the covariance matrix in a smoother way. On each iteration it interpolates it with the identity matrix before adding $\boldsymbol{x}_t\boldsymbol{x}_t^\top$. Note that if λ is an eigenvalue of Σ_{t-1}^{-1} then $\lambda \times (c/(\lambda + c)) < \lambda$ is an eigenvalue of $\left(\Sigma_{t-1} + c^{-1}I\right)^{-1}$. Thus the algorithm implicitly reduces the eigenvalues of the inverse covariance (and increases the eigenvalues of the covariance).

Finally, all three algorithms can be combined with Mercer kernels as they employ only sums of inner- and outer-products of its inputs. This allows them to perform non-linear predictions, similar to SVM.

5. Related Work

There is a large body of research in online learning for regression problems. Almost half a century ago, Widrow and Hoff (1960) developed a variant of the least mean squares (LMS) algorithm for adaptive filtering and noise reduction. The algorithm was further developed and analyzed extensively, for example by Feuer and Weinstein (1985). The normalized least mean squares filter (NLMS) (Bershad, 1986; Bitmead and Anderson, 1980) is similar to LMS but it is insensitive to scaling of the input. The recursive least squares (RLS) (Hayes, 1996) is the closest to our algorithms in the signal processing literature and also maintains a weight-vector and a covariance-like matrix, which is positive semi-definite (PSD), that is used to re-weight inputs.

In the machine learning literature the problem of online regression was studied extensively, and clearly we cannot cover all the relevant work. Cesa-Bianchi et al. (1993) studied gradient descent based algorithms for regression with the squared loss. Kivinen and Warmuth (1997) proposed various generalizations for general regularization functions. We refer the reader to a comprehensive book in the subject (Cesa-Bianchi and Lugosi, 2006).

Foster (1991) studied an online version of the ridge regression algorithm in the worst-case setting. Vovk (1990) proposed a related algorithm called the Aggregating Algorithm (AA), which was later applied to the problem of linear regression with square loss (Vovk, 1997, 2001). Forster (1999) simplified the regret analysis for this problem. Both algorithms employ second-order information. ARCOR for the separable case is very similar to these algorithms, although has alternative derivation. Recently, few algorithms were proposed either for classification (Cesa-Bianchi et al., 2005; Dredze et al., 2008; Crammer et al., 2008, 2009) or for general loss functions (Duchi et al., 2010; McMahan and Streeter, 2010) in the online convex programming framework. AROWR (Vaits and Crammer, 2011) shares the same design principles of AROW (Crammer et al., 2009) yet it is aimed for regression. The ARCOR algorithm takes AROWR one step further and it has two important modifications which makes it work in the drifting or shifting settings. These modifications make the analysis more complex than of AROW.

Two of the approaches used in previous algorithms for non-stationary setting are bounding the weight vector and covariance reset. Bounding the weight vector was performed either by projecting it into a bounded set (Herbster and Warmuth, 2001), shrinking it by multiplication (Kivinen et al., 2001), or subtraction of previously seen examples (Cavallanti et al., 2007). These three methods (or at least most of their variants) can be combined with kernel operators, and in fact, the last two approaches were designed and motivated by kernels.

The Covariance Reset RLS algorithm (CR-RLS) (Salgado et al., 1988; Goodwin et al., 83; Chen and Yen, 1999) was designed for adaptive filtering. CR-RLS makes covariance reset every fixed amount of data points, while ARCOR performs restarts based on the actual properties of the data - the eigenspectrum of the covariance matrix. Furthermore, as far as we know, there is no analysis in the mistake bound model for CR-RLS. Both ARCOR and CR-RLS are motivated from the property that the covariance matrix goes to zero and becomes rank deficient. In both algorithms the information encapsulated in the covariance matrix is lost after restarts. In a rapidly varying environments, like a wireless channel, this loss of memory can be beneficial, as previous contributions to the covariance matrix may have little correlation with the current structure. Recent versions of CR-RLS (Goodhart et al., 1991; Song et al., 2002) employ covariance reset to have numerically stable computations.

ARCOR algorithm combines two techniques with second-order algorithm for regression. In this aspect it has the best of all worlds, fast convergence rate due to the usage of second-order information, and the ability to adapt in non-stationary environments due to projection and resets.

LASER is simpler than all these algorithms as it controls the increase of the eigenvalues of the covariance matrix, implicitly rather than explicitly, by "averaging" it with a fixed diagonal matrix (see 11), and it does not involve projection steps. The Kalman filter (Kalman, 1960) and the H_∞ algorithm (e.g. the work of Simon 2006) designed for filtering take a similar approach, yet the exact algebraic form is different.

The derivation of the LASER algorithm in this work shares similarities with the work of Forster (1999) and the work of Moroshko and Crammer (2012). These algorithms are motivated from the last-step min-max predictor. Yet, the algorithms of Forster and Moroshko and Crammer are designed for the stationary setting, while LASER is primarily designed for the non-stationary setting. Moroshko and Crammer (2012) also discussed a weak variant of the non-stationary setting, where the complexity is measured by the total distance from a reference vector $\bar{\mathbf{u}}$, rather than the total distance of consecutive vectors (as in this paper), which is more relevant to non-stationary problems.

6. Regret Bounds

We now analyze our algorithms in the non-stationary case, upper bounding the regret using more than a single comparison vector. Specifically, our goal is to prove bounds that would hold uniformly for all inputs, and are of the form

$$L_T(\text{alg}) \leq L_T(\{\boldsymbol{u}_t\}) + \alpha(T)\left(V^{(P)}\right)^{\gamma} ,$$

for either $P = 1$ or $P = 2$, a constant γ and a function $\alpha(T)$ that may depend implicitly on other quantities of the problem.

Specifically, in the next section we show (Corollary 6) that under a particular choice of $\Lambda_i = \Lambda_i(V^{(1)})$ for the ARCOR algorithm, its regret is bounded by

$$L_T(\text{ARCOR}) \leq L_T(\{\boldsymbol{u}_t\}) + \mathcal{O}\left(T^{\frac{1}{2}}\left(V^{(1)}\right)^{\frac{1}{2}}\log T\right) .$$

Additionally, in Section 6.2, we show (Corollary 12) that under proper choice of the constant $c = c\left(V^{(2)}\right)$, the regret of LASER is bounded by

$$L_T(\text{LASER}) \leq L_T(\{\boldsymbol{u}_t\}) + \mathcal{O}\left(T^{\frac{2}{3}}\left(V^{(2)}\right)^{\frac{1}{3}}\right) .$$

The two bounds are not comparable in general. For example, assume a constant instantaneous drift $\|\boldsymbol{u}_{t+1}-\boldsymbol{u}_t\| = \nu$ for some constant value ν. In this case the variance and squared variance are, $V^{(1)} = T\nu$ and $V^{(2)} = T\nu^2$. The bound of ARCOR becomes asymptotically $\nu^{\frac{1}{2}}T\log T$, while the bound of LASER becomes asymptotically $\nu^{\frac{2}{3}}T$. Hence the bound of LASER is better in this case.

Another example is polynomial decay of the drift, $\|\boldsymbol{u}_{t+1}-\boldsymbol{u}_t\| \le t^{-\kappa}$ for some $\kappa > 0$. In this case, for $\kappa \neq 1$ we get[2] $V^{(1)} \le \sum_{t=1}^{T-1} t^{-\kappa} \le \int_1^{T-1} t^{-\kappa}dt + 1 = \frac{(T-1)^{1-\kappa}-\kappa}{1-\kappa}$. For $\kappa = 1$ we get $V^{(1)} \le \log(T-1)+1$. For LASER we have, for $\kappa \neq 0.5$, $V^{(2)} \le \sum_{t=1}^{T-1} t^{-2\kappa} \le \int_1^{T-1} t^{-2\kappa}dt + 1 = \frac{(T-1)^{1-2\kappa}-2\kappa}{1-2\kappa}$. For $\kappa = 0.5$ we get $V^{(2)} \le \log(T-1)+1$. Asymptotically, ARCOR outperforms LASER about when $\kappa \ge 0.7$.

Herbster and Warmuth (2001) developed shifting bounds for general gradient descent algorithms with projection of the weight-vector using the Bregman divergence. In their bounds, there is a factor greater than 1 multiplying the term $L_T(\{\boldsymbol{u}_t\})$ (see also theorem 11.4 in Cesa-Bianchi and Lugosi 2006). However, it is possible to get regret bound similar to ARCOR bound above, as they have an implicit parameter that can be tuned with the prior knowledge of $L_T(\{\boldsymbol{u}_t\})$ and $V^{(1)}$, leading to regret of $\mathcal{O}\left(\sqrt{L_T(\{\boldsymbol{u}_t\})V^{(1)}}\right)$, or just $\mathcal{O}\left(\sqrt{TV^{(1)}}\right)$, assuming only the knowledge of $V^{(1)}$.

Busuttil and Kalnishkan (2007) developed a variant of the Aggregating Algorithm (Vovk, 1990) for the non-stationary setting. However, to have sublinear regret they require a strong assumption on the drift $V^{(2)} = o(1)$, while we require only $V^{(2)} = o(T)$ (for LASER) or $V^{(1)} = o(T)$ (for ARCOR).

6.1 Analysis of the ARCOR Algorithm

Let us define additional notation that we will use in our bounds. We denote by t_i the example index for which a restart was performed for the ith time, that is $\Sigma_{t_i} = I$ for all i. We define by n the total number of restarts, or intervals. We denote by $T_i = t_i - t_{i-1}$ the number of examples between two consecutive restarts. Clearly $T = \sum_{i=1}^n T_i$. Finally, we denote by $\Sigma^{i-1} = \Sigma_{t_i-1}$ just before the ith restart, and we note that it depends on exactly T_i examples (since the last restart).

In what follows we compare the performance of the ARCOR algorithm to the performance of a sequence of weight vectors $\boldsymbol{u}_t \in \mathbb{R}^d$, which are of norm bounded by R_B. In other words, all the vectors $\boldsymbol{u}_t$ belong to B. We break the proof into four steps. In the first step (Theorem 3) we bound the regret when the algorithm is executed with some value of parameters $\{\Lambda_i\}$ and the resulting covariance matrices. In the second step, summarized in Corollary 4, we remove the dependencies in the covariance matrices by taking a worst case bound. In the third step, summarized in Lemma 5, we upper bound the total number of switches n given the parameters $\{\Lambda_i\}$. Finally, in Corollary 6 we provide the regret bound for a specific choice of the parameters. We now move to state the first theorem.

Theorem 3 *Assume that the ARCOR algorithm is run with an input sequence $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_T, y_T)$. Assume that all the inputs are upper bounded by unit norm $\|\boldsymbol{x}_t\| \le 1$ and that the outputs*

[2] This is correct because $f(t) = t^{-\kappa}$ is a monotonically decreasing function for $\kappa > 0$ and thus we can lower bound the integral with the right Riemann sum. In addition $f(1) = 1$.

are bounded by $Y = \max_t |y_t|$. Let $\boldsymbol{u}_t$ be any sequence of bounded weight vectors $\|\boldsymbol{u}_t\| \leq R_B$. Then, the cumulative loss is bounded by

$$\begin{aligned} L_T(ARCOR) \leq\ & L_T(\{\boldsymbol{u}_t\}) + 2R_B r \sum_t \frac{1}{\Lambda_{i(t)}} \|\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\| \\ & + r\boldsymbol{u}_T^\top \Sigma_T^{-1} \boldsymbol{u}_T + 2\left(R_B^2 + Y^2\right) \sum_i^n \log\det\left(\left(\Sigma^i\right)^{-1}\right) \ , \end{aligned}$$

where n is the number of covariance restarts and Σ^{i-1} is the value of the covariance matrix just before the ith restart.

The proof appears in Appendix C. Note that the number of restarts n is not fixed but depends both on the total number of examples T and the scheme used to set the values of the lower bound of the eigenvalues Λ_i. In general, the lower the values of Λ_i are, the smaller number of covariance-restarts occur, yet the larger the value of the last term of the bound is, which scales inversely proportional to Λ_i. A more precise statement is given in the next corollary.

Corollary 4 *Assume that the ARCOR algorithm made n restarts and $\{\Lambda_i\}$ are monotonically decreasing with i (which is satisfied by our choice later). Under the conditions of Theorem 3 we have*

$$\begin{aligned} L_T(ARCOR) \leq\ & L_T(\{\boldsymbol{u}_t\}) + 2R_B r \Lambda_n^{-1} \sum_t \|\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\| \\ & + 2\left(R_B^2 + Y^2\right) dn \log\left(1 + \frac{T}{nrd}\right) + r\boldsymbol{u}_T^\top \Sigma_T^{-1} \boldsymbol{u}_T \ . \end{aligned}$$

Proof By definition we have

$$\left(\Sigma^i\right)^{-1} = I + \frac{1}{r} \sum_{t=t_i}^{T_i+t_i} \boldsymbol{x}_t \boldsymbol{x}_t^\top \ .$$

Denote the eigenvalues of $\sum_{t=t_i}^{T_i+t_i} \boldsymbol{x}_t \boldsymbol{x}_t^\top$ by $\lambda_1, \ldots, \lambda_d$. Since $\|\boldsymbol{x}_t\| \leq 1$ their sum is $\mathrm{Tr}\left(\sum_{t=t_i}^{T_i+t_i} \boldsymbol{x}_t \boldsymbol{x}_t^\top\right) \leq T_i$. We use the concavity of the log function to bound $\log\det\left(\left(\Sigma^i\right)^{-1}\right) = \sum_j^d \log\left(1 + \frac{\lambda_j}{r}\right) \leq d\log\left(1 + \frac{T_i}{rd}\right)$. We use concavity again to bound the sum

$$\sum_i^n \log\det\left(\left(\Sigma^i\right)^{-1}\right) \leq \sum_i^n d\log\left(1 + \frac{T_i}{rd}\right) \leq dn\log\left(1 + \frac{T}{nrd}\right) \ ,$$

where we used the fact that $\sum_i^n T_i = T$. Substituting the last inequality in Theorem 3, as well as using the monotonicity of the coefficients, $\Lambda_i \geq \Lambda_n$ for all $i \leq n$, yields the desired bound. ■

Implicitly, the second and third terms of the bound have opposite dependence on n. The second term is decreasing with n. If n is small it means that the lower bound Λ_n is very low (otherwise we would make many restarts) and thus Λ_n^{-1} is large. The third term is increasing with $n \ll T$. We now make this implicit dependence explicit.

Our goal is to bound the number of restarts n as a function of the number of examples T. This depends on the exact sequence of values Λ_i used. The following lemma provides a bound on n given a specific sequence of Λ_i.

Lemma 5 *Assume that the ARCOR algorithm is run with some sequence of Λ_i. Then, the number of restarts is upper bounded by*

$$n \leq \max_N \left\{ N \ : \ T \geq r \sum_i^N \left(\Lambda_i^{-1} - 1\right) \right\} \ .$$

Proof Since $\sum_{i=1}^n T_i = T$, then the number of restarts is maximized when the number of examples between restarts T_i is minimized. We prove now a lower bound on T_i for all $i = 1 \ldots n$. A restart occurs for the ith time when the smallest eigenvalue of Σ_t is smaller (for the first time) than Λ_i.

As before, by definition, $\left(\Sigma^i\right)^{-1} = I + \frac{1}{r}\sum_{t=t_i}^{T_i+t_i} \boldsymbol{x}_t \boldsymbol{x}_t^\top$. By a result in matrix analysis (Golub and Van Loan, 1996, Theorem 8.1.8) we have that there exists a matrix $A \in \mathbb{R}^{d\times T_i}$ with each column belongs to a bounded convex body that satisfy $a_{k,l} \geq 0$ and $\sum_k a_{k,l} \leq 1$ for $l = 1, \ldots, T_i$, such that the kth eigenvalue λ_k^i of $\left(\Sigma^i\right)^{-1}$ equals to $\lambda_k^i = 1 + \frac{1}{r}\sum_{l=1}^{T_i} a_{k,l}$. The value of T_i is defined when the largest eigenvalue of $\left(\Sigma^i\right)^{-1}$ hits Λ_i^{-1}. Formally, we get the following lower bound on T_i,

$$\begin{aligned} &\arg\min_{\{a_{k,l}\}} \ s \\ &\quad \text{s.t.} \ \max_k \left(1 + \frac{1}{r}\sum_{l=1}^{s} a_{k,l}\right) \geq \Lambda_i^{-1} \\ &\quad\quad a_{k,l} \geq 0 \quad \text{for } k = 1, \ldots, d, l = 1, \ldots, s \\ &\quad\quad \sum_k a_{k,l} \leq 1 \quad \text{for } l = 1, \ldots, s \ . \end{aligned}$$

For a fixed value of s, a maximal value $\max_k \left(1 + \frac{1}{r}\sum_{l=1}^{s} a_{k,l}\right)$ is obtained when each column of A concentrates the "mass" in one value $k = k_0$ and equal to its maximal value $a_{k_0,l} = 1$ for $l = 1, \ldots, s$. That is, we have $a_{k,l} = 1$ for $k = k_0$ and $a_{k,l} = 0$ otherwise. In this case $\max_k \left(1 + \frac{1}{r}\sum_{l=1}^{s} a_{k,l}\right) = 1 + \frac{1}{r}s$ and the lower bound is obtained when $1 + \frac{1}{r}s = \Lambda_i^{-1}$. Solving for s we get that the shortest possible length of the ith interval is bounded by, $T_i \geq r\left(\Lambda_i^{-1} - 1\right)$. Summing over the last equation we get, $T = \sum_i^n T_i \geq r\sum_i^n \left(\Lambda_i^{-1} - 1\right)$. Thus, the number of restarts is upper bounded by the maximal value n that satisfies the last inequality. ■

We now prove a bound for a specific choice of the parameters $\{\Lambda_i\}$, namely polynomial decay, $\Lambda_i^{-1} = i^{q-1} + 1$ for $q > 1$ (note that the thresholds $\{\Lambda_i\}$ are monotonically decreasing with i). This scheme of setting $\{\Lambda_i\}$ balances between the amount of drift (need for many restarts) and the property that using the covariance matrix for updates achieves fast convergence. We note that an exponential scheme $\Lambda_i = 2^{-i}$ will lead to very few restarts, and very small eigenvalues of the covariance matrix. Intuitively, this is because the last segment will be about half the length of the entire sequence. Combining Lemma 5 with Corollary 4 we get,

Corollary 6 *Assume that the ARCOR algorithm is run with a polynomial scheme, that is $\Lambda_i^{-1} = i^{q-1} + 1$ for some $q > 1$. Under the conditions of Theorem 3 we have*

$$\begin{aligned} L_T(ARCOR) \leq \; & L_T(\{\boldsymbol{u}_t\}) + r\boldsymbol{u}_T^\top \Sigma_T^{-1} \boldsymbol{u}_T \\ & + 2\left(R_B^2 + Y^2\right) d \left(qT+1\right)^{\frac{1}{q}} \log\left(1 + \frac{T}{nrd}\right) \qquad (20) \\ & + 2R_B r \left(\left(qT+1\right)^{\frac{q-1}{q}} + 1\right) \sum_t \left\|\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\right\| \; . \qquad (21) \end{aligned}$$

Proof Substituting $\Lambda_i^{-1} = i^{q-1} + 1$ in Lemma 5 we get

$$T \geq r \sum_i^n \left(\Lambda_i^{-1} - 1\right) = r \sum_{i=1}^n i^{q-1} \geq r \int_1^n x^{q-1} dx = \frac{r}{q}\left(n^q - 1\right) \; ,$$

where the middle inequality is correct because $f(x) = x^{q-1}$ for $q > 1$ is a monotonically increasing function and thus we can upper bound the integral with the right Riemann sum. This yields an upper bound on n,

$$n \leq \left(qT+1\right)^{\frac{1}{q}} \quad \Rightarrow \quad \Lambda_n^{-1} \leq \left(qT+1\right)^{\frac{q-1}{q}} + 1 \; .$$

■

Comparing the last two terms of the bound of Corollary 6 we observe a natural tradeoff in the value of q. The third term of (20) is decreasing with large values of q, while the fourth term of (21) is increasing with q.

Assuming a bound on the deviation $\sum_t \|\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\| = V_T^{(1)} \leq \mathcal{O}\left(T^{1/p}\right)$, or in other words $p = \left(\log T\right) / \left(\log V^{(1)}\right)$. We set a drift dependent parameter $q = (2p)/(p+1) = (2\log T) / \left(\log T + \log V^{(1)}\right)$ and get that the sum of (20) and (21) is of order $\mathcal{O}\left(T^{\frac{p+1}{2p}} \log(T)\right) = \mathcal{O}\left(\sqrt{V^{(1)} T} \log T\right)$.

Few comments are in order. First, as long as $p > 1$ the sum of (20) and (21) is $o(T)$ and thus vanishing. Second, when the drift is very small, that is $p \approx -(1+\epsilon)$, the algorithm sets $q \approx 2 + (2/\epsilon)$, and thus it will not make any restarts, and the bound of $\mathcal{O}(\log T)$ for the stationary case is retrieved. In other words, for this choice of q the algorithm will have only one interval, and there will be no restarts.

To conclude, we showed that if the algorithm is given an upper bound on the amount of drift, which is sub-linear in T, it can achieve sub-linear regret. Furthermore, if it is known that there is no non-stationarity in the reference vectors, then running the algorithm with large enough q will have a regret logarithmic in T.

6.2 Analysis of the LASER Algorithm

We now analyze the performance of the LASER algorithm in the worst-case setting in six steps. First, state a technical lemma that is used in the second step (Theorem 8), in which we bound the regret with a quantity proportional to $\sum_{t=1}^{T}\boldsymbol{x}_t^\top D_t^{-1}\boldsymbol{x}_t$. Third, in Lemma 9 we bound each of the summands with two terms, one logarithmic and one linear in the eigenvalues of the matrices D_t. In the fourth (Lemma 10) and fifth (Lemma 11) steps we bound the eigenvalues of D_t first for scalars and then extend the results to matrices. Finally, in Corollary 12 we put all these results together and get the desired bound.

Lemma 7 *For all t the following statement holds*

$$D'_{t-1}D_t^{-1}\boldsymbol{x}_t\boldsymbol{x}_t^\top D_t^{-1}D'_{t-1} + D'_{t-1}\left(D_t^{-1}D'_{t-1}+c^{-1}I\right) - D_{t-1}^{-1} \preceq 0 \ ,$$

where as defined in (19) *we have* $D'_{t-1} = \left(I + c^{-1}D_{t-1}\right)^{-1}$.

The proof appears in Appendix D. We next bound the cumulative loss of the algorithm.

Theorem 8 *Assume that the labels are bounded* $\sup_t|y_t| \leq Y$ *for some* $Y \in \mathbb{R}$. *Then the following bound holds*

$$L_T(LASER) \leq \min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_T}\left[L_T(\{\boldsymbol{u}_t\}) + cV_T^{(2)}(\{\boldsymbol{u}_t\}) + b\left\|\boldsymbol{u}_1\right\|^2\right] + Y^2\sum_{t=1}^{T}\boldsymbol{x}_t^\top D_t^{-1}\boldsymbol{x}_t \ . \tag{22}$$

Proof Fix t. A long algebraic manipulation, given in Appendix E, yields

$$\begin{aligned}
&(y_t-\hat{y}_t)^2 + \min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_{t-1}} Q_{t-1}\left(\boldsymbol{u}_1,\ldots,\boldsymbol{u}_{t-1}\right) - \min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_t} Q_t\left(\boldsymbol{u}_1,\ldots,\boldsymbol{u}_t\right)\\
&= (y_t-\hat{y}_t)^2 + 2y_t\boldsymbol{x}_t^\top D_t^{-1}D'_{t-1}\boldsymbol{e}_{t-1} + \boldsymbol{e}_{t-1}^\top\left[-D_{t-1}^{-1} + D'_{t-1}\left(D_t^{-1}D'_{t-1}+c^{-1}I\right)\right]\boldsymbol{e}_{t-1}\\
&\quad + y_t^2\boldsymbol{x}_t^\top D_t^{-1}\boldsymbol{x}_t - y_t^2 \ .
\end{aligned} \tag{23}$$

Substituting the specific value of the predictor $\hat{y}_t = \boldsymbol{x}_t^\top D_t^{-1}D'_{t-1}\boldsymbol{e}_{t-1}$ from (18), we get that (23) equals to

$$\begin{aligned}
&\hat{y}_t^2 + y_t^2\boldsymbol{x}_t^\top D_t^{-1}\boldsymbol{x}_t + \boldsymbol{e}_{t-1}^\top\left[-D_{t-1}^{-1} + D'_{t-1}\left(D_t^{-1}D'_{t-1}+c^{-1}I\right)\right]\boldsymbol{e}_{t-1}\\
= \ &\boldsymbol{e}_{t-1}^\top D'_{t-1}D_t^{-1}\boldsymbol{x}_t\boldsymbol{x}_t^\top D_t^{-1}D'_{t-1}\boldsymbol{e}_{t-1} + y_t^2\boldsymbol{x}_t^\top D_t^{-1}\boldsymbol{x}_t\\
&+ \boldsymbol{e}_{t-1}^\top\left[-D_{t-1}^{-1} + D'_{t-1}\left(D_t^{-1}D'_{t-1}+c^{-1}I\right)\right]\boldsymbol{e}_{t-1}\\
= \ &\boldsymbol{e}_{t-1}^\top\tilde{D}_t\boldsymbol{e}_{t-1} + y_t^2\boldsymbol{x}_t^\top D_t^{-1}\boldsymbol{x}_t \ ,
\end{aligned} \tag{24}$$

where $\tilde{D}_t = D'_{t-1}D_t^{-1}\boldsymbol{x}_t\boldsymbol{x}_t^\top D_t^{-1}D'_{t-1} - D_{t-1}^{-1} + D'_{t-1}\left(D_t^{-1}D'_{t-1}+c^{-1}I\right)$. Using Lemma 7 we upper bound $\tilde{D}_t \preceq 0$ and thus (24) is bounded,

$$y_t^2\boldsymbol{x}_t^\top D_t^{-1}\boldsymbol{x}_t \leq Y^2\boldsymbol{x}_t^\top D_t^{-1}\boldsymbol{x}_t \ .$$

Finally, summing over $t \in \{1, \ldots, T\}$ gives the desired bound,

$$L_T(\text{LASER}) - \min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_T} \left[b \left\| \boldsymbol{u}_1 \right\|^2 + cV_T^{(2)}(\{\boldsymbol{u}_t\}) + L_T(\{\boldsymbol{u}_t\}) \right] \leq Y^2 \sum_{t=1}^{T} \boldsymbol{x}_t^\top D_t^{-1} \boldsymbol{x}_t \ .$$

■

In the next lemma we further bound the right term of (22). This type of bound is based on the usage of the covariance-like matrix D.

Lemma 9

$$\sum_{t=1}^{T} \boldsymbol{x}_t^\top D_t^{-1} \boldsymbol{x}_t \leq \ln \left| \frac{1}{b} D_T \right| + c^{-1} \sum_{t=1}^{T} \operatorname{Tr}\left(D_{t-1}\right) \ . \tag{25}$$

Proof Let $B_t \doteq D_t - \boldsymbol{x}_t \boldsymbol{x}_t^\top = \left(D_{t-1}^{-1} + c^{-1} I\right)^{-1} \succ 0$.

$$\begin{aligned}
\boldsymbol{x}_t^\top D_t^{-1} \boldsymbol{x}_t &= \operatorname{Tr}\left(\boldsymbol{x}_t^\top D_t^{-1} \boldsymbol{x}_t\right) = \operatorname{Tr}\left(D_t^{-1} \boldsymbol{x}_t \boldsymbol{x}_t^\top\right) \\
&= \operatorname{Tr}\left(D_t^{-1}\left(D_t - B_t\right)\right) \\
&= \operatorname{Tr}\left(D_t^{-1/2}\left(D_t - B_t\right) D_t^{-1/2}\right) \\
&= \operatorname{Tr}\left(I - D_t^{-1/2} B_t D_t^{-1/2}\right) \\
&= \sum_{j=1}^{d} \left[1 - \lambda_j\left(D_t^{-1/2} B_t D_t^{-1/2}\right)\right] \ .
\end{aligned}$$

We continue using $1 - x \leq -\ln(x)$ and get

$$\begin{aligned}
\boldsymbol{x}_t^\top D_t^{-1} \boldsymbol{x}_t &\leq -\sum_{j=1}^{d} \ln\left[\lambda_j\left(D_t^{-1/2} B_t D_t^{-1/2}\right)\right] \\
&= -\ln\left[\prod_{j=1}^{d} \lambda_j\left(D_t^{-1/2} B_t D_t^{-1/2}\right)\right] \\
&= -\ln\left|D_t^{-1/2} B_t D_t^{-1/2}\right| \\
&= \ln \frac{|D_t|}{|B_t|} = \ln \frac{|D_t|}{\left|D_t - \boldsymbol{x}_t \boldsymbol{x}_t^\top\right|} \ .
\end{aligned}$$

It follows that

$$\begin{aligned}
\boldsymbol{x}_t^\top D_t^{-1} \boldsymbol{x}_t &\leq \ln \frac{|D_t|}{\left|\left(D_{t-1}^{-1} + c^{-1} I\right)^{-1}\right|} \\
&= \ln \frac{|D_t|}{|D_{t-1}|} \left|\left(I + c^{-1} D_{t-1}\right)\right| \\
&= \ln \frac{|D_t|}{|D_{t-1}|} + \ln\left|\left(I + c^{-1} D_{t-1}\right)\right| \ .
\end{aligned}$$

and because $\ln\left|\frac{1}{b}D_0\right| \geq 0$ we get

$$\sum_{t=1}^{T} \boldsymbol{x}_t^\top D_t^{-1}\boldsymbol{x}_t \leq \ln\left|\frac{1}{b}D_T\right| + \sum_{t=1}^{T} \ln\left|\left(I + c^{-1}D_{t-1}\right)\right| \leq \ln\left|\frac{1}{b}D_T\right| + c^{-1}\sum_{t=1}^{T} \mathrm{Tr}\left(D_{t-1}\right) \ .$$

■

At first sight it seems that the right term of (25) may grow super-linearly with T, as each of the matrices D_t grows with t. The next two lemmas show that this is not the case, and in fact, the right term of (25) is not growing too fast, which will allow us to obtain a sub-linear regret bound. Lemma 10 analyzes the properties of the recursion of D defined in (11) for scalars, that is $d = 1$. In Lemma 11 we extend this analysis to matrices.

Lemma 10 *Define $f(\lambda) = \lambda\beta/\left(\lambda + \beta\right) + x^2$ for $\beta, \lambda \geq 0$ and some $x^2 \leq \gamma^2$. Then:*

1. $f(\lambda) \leq \beta + \gamma^2$
2. $f(\lambda) \leq \lambda + \gamma^2$
3. $f(\lambda) \leq \max\left\{\lambda, \frac{3\gamma^2+\sqrt{\gamma^4+4\gamma^2\beta}}{2}\right\}$

Proof For the first property we have $f(\lambda) = \lambda\beta/\left(\lambda + \beta\right) + x^2 \leq \beta \times 1 + x^2$. The second property follows from the symmetry between β and λ. To prove the third property we decompose the function as, $f(\lambda) = \lambda - \frac{\lambda^2}{\lambda+\beta} + x^2$. Therefore, the function is bounded by its argument $f(\lambda) \leq \lambda$ if, and only if, $-\frac{\lambda^2}{\lambda+\beta} + x^2 \leq 0$. Since we assume $x^2 \leq \gamma^2$, the last inequality holds if, $-\lambda^2 + \gamma^2\lambda + \gamma^2\beta \leq 0$, which holds for $\lambda \geq \frac{\gamma^2+\sqrt{\gamma^4+4\gamma^2\beta}}{2}$.

To conclude. If $\lambda \geq \frac{\gamma^2+\sqrt{\gamma^4+4\gamma^2\beta}}{2}$, then $f(\lambda) \leq \lambda$. Otherwise, by the second property, we have

$$f(\lambda) \leq \lambda + \gamma^2 \leq \frac{\gamma^2 + \sqrt{\gamma^4 + 4\gamma^2\beta}}{2} + \gamma^2 = \frac{3\gamma^2 + \sqrt{\gamma^4 + 4\gamma^2\beta}}{2},$$

as required. ■

We build on Lemma 10 to bound the maximal eigenvalue of the matrices D_t.

Lemma 11 *Assume $\left\|\boldsymbol{x}_t\right\|^2 \leq X^2$ for some X. Then, the eigenvalues of D_t (for $t \geq 1$), denoted by $\lambda_i\left(D_t\right)$, are upper bounded by*

$$\max_i \lambda_i\left(D_t\right) \leq \max\left\{\frac{3X^2 + \sqrt{X^4 + 4X^2c}}{2}, b + X^2\right\} \ .$$

Proof By induction. From (11) we have that $\lambda_i(D_1) \leq b + X^2$ for $i = 1, \ldots, d$. We proceed with a proof for some t. For simplicity, denote by $\lambda_i = \lambda_i(D_{t-1})$ the ith eigenvalue of D_{t-1}

with a corresponding eigenvector $\boldsymbol{v}_i$. From (11) we have

$$\begin{aligned}D_t &= \left(D_{t-1}^{-1} + c^{-1} I\right)^{-1} + \boldsymbol{x}_t \boldsymbol{x}_t^\top \\ &\preceq \left(D_{t-1}^{-1} + c^{-1} I\right)^{-1} + I \left\|\boldsymbol{x}_t\right\|^2 \\ &= \sum_i^d \boldsymbol{v}_i \boldsymbol{v}_i^\top \left(\left(\lambda_i^{-1} + c^{-1}\right)^{-1} + \left\|\boldsymbol{x}_t\right\|^2\right) \\ &= \sum_i^d \boldsymbol{v}_i \boldsymbol{v}_i^\top \left(\frac{\lambda_i c}{\lambda_i + c} + \left\|\boldsymbol{x}_t\right\|^2\right) \ . \end{aligned} \tag{26}$$

Plugging Lemma 10 in (26) we get

$$\begin{aligned}D_t &\preceq \sum_i^d \boldsymbol{v}_i \boldsymbol{v}_i^\top \max\left\{\frac{3X^2 + \sqrt{X^4 + 4X^2 c}}{2}, b + X^2\right\} \\ &= \max\left\{\frac{3X^2 + \sqrt{X^4 + 4X^2 c}}{2}, b + X^2\right\} I \ .\end{aligned}$$

■

Finally, equipped with the above lemmas we are able to prove the main result of this section.

Corollary 12 *Assume $\left\|\boldsymbol{x}_t\right\|^2 \leq X^2$, $|y_t| \leq Y$. Then*

$$\begin{aligned}L_T(LASER) \leq\ & b\left\|\mathbf{u}_1\right\|^2 + L_T(\{\boldsymbol{u}_t\}) + Y^2 \ln\left|\frac{1}{b} D_T\right| + c^{-1} Y^2 \mathrm{Tr}\left(D_0\right) + c V^{(2)} \\ & + c^{-1} Y^2 T d \max\left\{\frac{3X^2 + \sqrt{X^4 + 4X^2 c}}{2}, b + X^2\right\} \ .\end{aligned} \tag{27}$$

Furthermore, set $b = \varepsilon c$ for some $0 < \varepsilon < 1$. Denote by $\mu = \max\left\{9/8X^2, \frac{\left(b+X^2\right)^2}{8X^2}\right\}$ and $M = \max\left\{3X^2, b + X^2\right\}$. If $V^{(2)} \leq T\frac{\sqrt{2}Y^2 dX}{\mu^{3/2}}$ (low drift) then by setting

$$c = \frac{\sqrt{2} T Y^2 d X}{\left(V^{(2)}\right)^{2/3}} \tag{28}$$

we have

$$\begin{aligned}L_T(LASER) \leq\ & b\left\|\mathbf{u}_1\right\|^2 + 3\left(\sqrt{2} Y^2 d X\right)^{2/3} T^{2/3} \left(V^{(2)}\right)^{1/3} + \frac{\varepsilon}{1-\varepsilon} Y^2 d + L_T(\{\boldsymbol{u}_t\}) \\ & + Y^2 \ln\left|\frac{1}{b} D_T\right| \ .\end{aligned} \tag{29}$$

The proof appears in Appendix F. Note that if $V^{(2)} \geq T\frac{Y^2 dM}{\mu^2}$ then by setting $c = \sqrt{Y^2 dMT/V^{(2)}}$ we have

$$L_T(\text{LASER}) \leq b \left\|\mathbf{u}_1\right\|^2 + 2\sqrt{Y^2 dTMV^{(2)}} + \frac{\varepsilon}{1-\varepsilon}Y^2 d + L_T(\{\boldsymbol{u}_t\}) + Y^2 \ln\left|\frac{1}{b}D_T\right| \ . \quad (30)$$

(see Appendix G for details). The last bound is linear in T and can be obtained also by a naive algorithm that outputs $\hat{y}_t = 0$ for all t.

A few remarks are in order. When the variance $V^{(2)} = 0$ goes to zero, we set $c = \infty$ and thus we have $D_t = bI + \sum_{s=1}^{t} \boldsymbol{x}_s \boldsymbol{x}_s^\top$ used in recent algorithms (Vovk, 2001; Forster, 1999; Hayes, 1996; Cesa-Bianchi et al., 2005). In this case the algorithm reduces to the algorithm by Forster (1999) (which is also the AAR algorithm of Vovk 2001), with the same logarithmic regret bound (note that the term $\ln\left|\frac{1}{b}D_T\right|$ in the bounds is logarithmic in T, see the proof of Forster 1999). See also the work of Azoury and Warmuth (2001).

7. Simulations

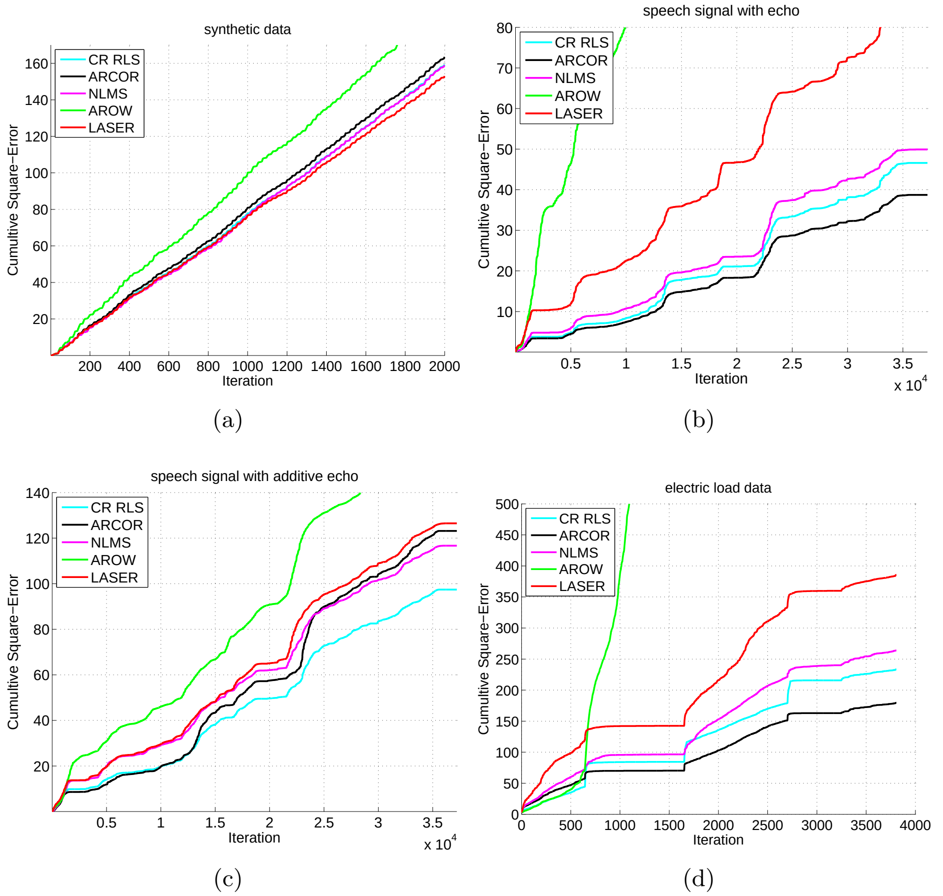
We evaluated our algorithms on four data sets, one synthetic and three real-world. The synthetic data set contains $2,000$ points $\boldsymbol{x}_t \in \mathbb{R}^{20}$, where the first ten coordinates were grouped into five groups of size two. Each such pair was drawn from a 45° rotated Gaussian distribution with standard deviations 10 and 1. The remaining 10 coordinates of $\boldsymbol{x}_t$ were drawn from independent Gaussian distributions $\mathcal{N}(0,2)$. The data set was generated using a sequence of vectors $\boldsymbol{u}_t \in \mathbb{R}^{20}$ for which the only non-zero coordinates are the first two, where their values are the coordinates of a unit vector that is rotating with a constant rate. Specifically, we have $\|\boldsymbol{u}_t\| = 1$ and the instantaneous drift $\|\boldsymbol{u}_t - \boldsymbol{u}_{t-1}\|$ is constant. The labels were set according to $y_t = \boldsymbol{x}_t^\top \boldsymbol{u}_t$.

The first two real-world data sets were generated from echoed speech signal. The first speech echoed signal was generated using FIR filter with k delays and varying attenuated amplitude. This effect imitates acoustic echo reflections from large, distant and dynamic obstacles. The difference equation $y(n) = x(n) + \sum_{D=1}^{k} A(n)x(n-D) + v(n)$ was used, where D is a delay in samples, the coefficient $A(n)$ describes the changing attenuation related to object reflection and $v(n) \sim \mathcal{N}\left(0, 10^{-3}\right)$ is a white noise. The second speech echoed signal was generated using a flange IIR filter, where the delay is not constant, but changing with time. This effect imitates time stretching of audio signal caused by moving and changing objects in the room. The difference equation $y(n) = x(n) + Ay\left(n - D(n)\right) + v(n)$ was used.

The last real-world data set was taken from the Kaggle competition "Global Energy Forecasting Competition 2012 - Load Forecasting".[3] This data set includes hourly demand for four and a half years from 20 different geographic regions, and similar hourly temperature readings from 11 zones, which we used as features $\boldsymbol{x}_t \in \mathbb{R}^{11}$. Based on this data set, we generated drifting and shifting data as follows: we predict the load 3 times a day (thus there is a drift between day and night), and every half a year there is a switch in the region where the load is predicted.

Five algorithms were evaluated: NLMS (normalized least mean square) (Bershad, 1986; Bitmead and Anderson, 1980) which is a state-of-the-art first-order algorithm, AROWR

[3] The data set was taken from
`http://www.kaggle.com/c/global-energy-forecasting-competition-2012-load-forecasting` .



(a)

(b)

(c)

(d)

Figure 1: Cumulative squared loss for AROWR, ARCOR, LASER, NLMS and CR-RLS vs iteration. (a) Results for synthetic data set with drift. (b) Results for a problem of acoustic echo cancellation on speech signal generated using FIR filter and (c) IIR filter. (d) Results for a problem of electric load prediction (best shown in color).

(AROW for Regression) with no restarts nor projection, ARCOR, LASER and CR-RLS. We note that AAR (Vovk, 2001) is a special case of LASER and RLS is a special case of CR-RLS, for a specific choice of their respective parameters ($c = \infty$ for LASER and $T_0 = \infty$ for CR-RLS) . Additionally, the performance of AROWR, AAR and RLS is similar, and thus only the performance of AROWR is shown. For the synthetic data set the algorithms' parameters were tuned using a single random sequence. For the speech signal the algorithms' parameters were tuned on 10% of the signal, then the best parameter choices for each algorithm were used to evaluate the performance on the remaining signal. Similarly, for the load data set the algorithms' parameters were tuned on 20% of the signal.

The results are summarized in Figure 1. AROWR performs the worst on all data sets as it converges very fast and thus not able to track the changes in the data. Focusing on Figure 1(a), showing the results for the synthetic signal, we observe that ARCOR performs relatively bad as suggested by our analysis for constant, yet not too large, drift. Both CR-RLS and NLMS perform better, where CR-RLS is slightly better as it is a second-order algorithm, and allows to converge faster between switches. On the other hand, NLMS is not converging and is able to adapt to the drift. Finally, LASER performs the best, as hinted by its analysis, for which the bound is lower where there is a constant drift.

Moving to Figure 1(b), showing the results for first echoed speech signal with varying amplitude, we observe that LASER is the worst among all algorithms except AROWR. Indeed, it prevents the convergence by keeping the learning rate far from zero, yet it is a min-max algorithm designed for the worst-case, which is not the case for real-world speech data. However, speech data is highly regular and the instantaneous drift vary. NLMS performs better as it does not converge, yet both CR-RLS and ARCOR perform even better, as they both do not converge due to covariance resets on the one hand, and second-order updates on the other hand. ARCOR outperforms CR-RLS as the former adapts the resets to actual data, and does not use pre-defined scheduling as the later.

Figure 1(c) summarizes the results for evaluations on the second echoed speech signal. Note that the amount of drift grows since the data is generated using flange filter. Both LASER and ARCOR are outperformed as both assume drift that is sublinear or at most linear, which is not the case. CR-RLS outperforms NLMS. The later is first order, so is able to adapt to changes, yet has slower convergence rate. The former is able to cope with drift due to resets.

Finally, Figure 1(d) summarizes the results for the electric load data set. ARCOR outperforms other algorithms, as the drift is sublinear and it has the ability to adapt resets to the data. Again, LASER is a min-max algorithm designed for the worst case, which is usually not the case for real-world data.

Interestingly, in all experiments, NLMS was not performing the best nor the worst. There is no clear winner among the three algorithms that are both second-order (ARCOR, LASER, CR-RLS), and designed to adapt to drifts. Intuitively, if the drift suits the assumptions of an algorithm, that algorithm would perform the best, and otherwise, its performance may even be worse than of NLMS.

We have seen above that ARCOR performs a projection step, which partially was motivated from the analysis. We now evaluate its need and affect in practice on two speech problems. We test two modifications of ARCOR, resulting in four variants altogether. First, we replace the polynomial thresholds scheme to the constant thresholds scheme, that is, all thresholds are equal. Second, we omit the projection step. The results are summarized in Figure 2. The line corresponding to the original algorithm, is called “proj, poly” as it performs a projection step and uses polynomial scheme for the lower-bound on eigenvalues. The version that omits projection and uses constant scheme, called “no proj, const”, is most similar to CR-RLS. Both resets the covariance matrix, CR-RLS after fixed amount of iterations, while “ARCOR-no proj, const” when the eigenvalues meets a specified fixed lower bound. The difference between the two plots is the amount of drift used: the left plot shows results for sublinear drift, and the right plot shows results with increasing per-instance drift. The original version, as hinted by the analysis, is designed to work with sub-linear drift,

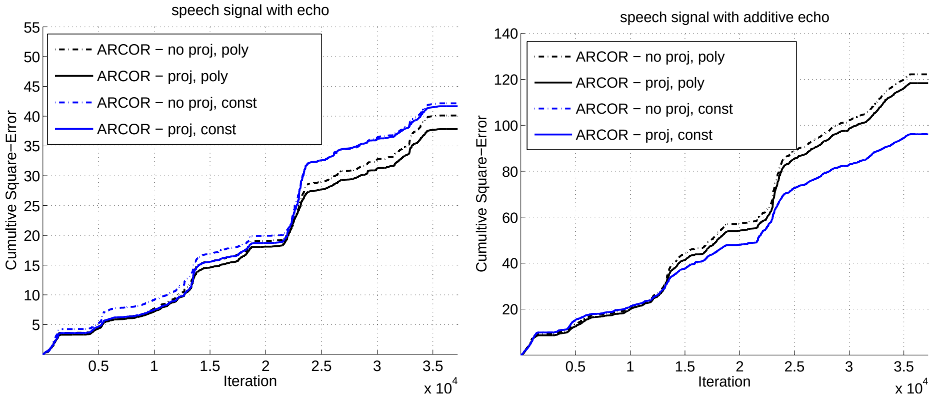


Figure 2: Cumulative squared loss of four variants of ARCOR vs iteration.

and performs the best in this case. However, when this assumption over the amount of drift breaks, this version is not optimal anymore, and constant scheme performs better, as it allows the algorithm to adapt to non-vanishing drift. Finally, in both data sets, the algorithm that performs the best performs a projection step after each iteration, providing some empirical evidence for its need.

8. Summary and Conclusions

We proposed and analyzed two novel algorithms for non-stationary online regression designed and analyzed with the squared loss in the worst-case regret framework. The ARCOR algorithm was built on AROWR. It employs second-order information, yet performs data-dependent covariance resets, which provides it the ability to track drifts. The LASER algorithm was built on the last-step minmax predictor with the proper modifications for non-stationary problems. Our algorithms require some prior knowledge of the drift to get optimal performance, and each algorithm works best in other drift level. The optimal setting depends on the actual drift in the data and the optimality of our bounds is an open issue.

Few open directions are possible. First, extension of these algorithms to other loss functions rather than the squared loss. Second, currently, direct implementation of both algorithms requires either matrix inversion or eigenvector decomposition. A possible direction is to design a more efficient version of these algorithms. Third, an interesting direction is to design algorithms that automatically detect the level of drift, or do not need this information before run-time.

Acknowledgments: This research was funded in part by the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI) and in part by an Israeli Science Foundation grant ISF- 1567/10.

Appendix A. Proof of Lemma 1

Proof We calculate

$$\begin{aligned}
P_t\left(\boldsymbol{u}_t\right) &= \min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_{t-1}}\left(b\left\|\boldsymbol{u}_1\right\|^2 + c\sum_{s=1}^{t-1}\left\|\boldsymbol{u}_{s+1}-\boldsymbol{u}_s\right\|^2 + \sum_{s=1}^{t}\left(y_s - \boldsymbol{u}_s^\top\boldsymbol{x}_s\right)^2\right)\\
&= \min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_{t-1}}\left(b\left\|\boldsymbol{u}_1\right\|^2 + c\sum_{s=1}^{t-2}\left\|\boldsymbol{u}_{s+1}-\boldsymbol{u}_s\right\|^2 + \sum_{s=1}^{t-1}\left(y_s - \boldsymbol{u}_s^\top\boldsymbol{x}_s\right)^2 + c\left\|\boldsymbol{u}_t-\boldsymbol{u}_{t-1}\right\|^2\right.\\
&\qquad\left. + \left(y_t - \boldsymbol{u}_t^\top\boldsymbol{x}_t\right)^2\right)\\
&= \min_{\boldsymbol{u}_{t-1}}\min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_{t-2}}\left(b\left\|\boldsymbol{u}_1\right\|^2 + c\sum_{s=1}^{t-2}\left\|\boldsymbol{u}_{s+1}-\boldsymbol{u}_s\right\|^2 + \sum_{s=1}^{t-1}\left(y_s - \boldsymbol{u}_s^\top\boldsymbol{x}_s\right)^2 + c\left\|\boldsymbol{u}_t-\boldsymbol{u}_{t-1}\right\|^2\right.\\
&\qquad\left. + \left(y_t - \boldsymbol{u}_t^\top\boldsymbol{x}_t\right)^2\right)\\
&= \min_{\boldsymbol{u}_{t-1}}\left[\min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_{t-2}}\left(b\left\|\boldsymbol{u}_1\right\|^2 + c\sum_{s=1}^{t-2}\left\|\boldsymbol{u}_{s+1}-\boldsymbol{u}_s\right\|^2 + \sum_{s=1}^{t-1}\left(y_s - \boldsymbol{u}_s^\top\boldsymbol{x}_s\right)^2\right)\right.\\
&\qquad\left. + c\left\|\boldsymbol{u}_t-\boldsymbol{u}_{t-1}\right\|^2 + \left(y_t - \boldsymbol{u}_t^\top\boldsymbol{x}_t\right)^2\right]\\
&= \min_{\boldsymbol{u}_{t-1}}\left(P_{t-1}\left(\boldsymbol{u}_{t-1}\right) + c\left\|\boldsymbol{u}_t-\boldsymbol{u}_{t-1}\right\|^2 + \left(y_t - \boldsymbol{u}_t^\top\boldsymbol{x}_t\right)^2\right)\ .
\end{aligned}$$

■

Appendix B. Proof of Lemma 2

Proof By definition

$$P_1\left(\boldsymbol{u}_1\right) = Q_1\left(\boldsymbol{u}_1\right) = b\left\|\boldsymbol{u}_1\right\|^2 + \left(y_1 - \boldsymbol{u}_1^\top\boldsymbol{x}_1\right)^2 = \boldsymbol{u}_1^\top\left(bI + \boldsymbol{x}_1\boldsymbol{x}_1^\top\right)\boldsymbol{u}_1 - 2y_1\boldsymbol{u}_1^\top\boldsymbol{x}_1 + y_1^2\ ,$$

and indeed $D_1 = bI + \boldsymbol{x}_1\boldsymbol{x}_1^\top$, $\boldsymbol{e}_1 = y_1\boldsymbol{x}_1$, and $f_1 = y_1^2$.

We proceed by induction, assume that, $P_{t-1}\left(\boldsymbol{u}_{t-1}\right)=\boldsymbol{u}_{t-1}^{\top}D_{t-1}\boldsymbol{u}_{t-1}-2\boldsymbol{u}_{t-1}^{\top}\boldsymbol{e}_{t-1}+f_{t-1}$. Applying Lemma 1 we get

$$
\begin{aligned}
P_t\left(\boldsymbol{u}_t\right) &= \min_{\boldsymbol{u}_{t-1}}\left(\boldsymbol{u}_{t-1}^{\top}D_{t-1}\boldsymbol{u}_{t-1}-2\boldsymbol{u}_{t-1}^{\top}\boldsymbol{e}_{t-1}+f_{t-1}+c\left\|\boldsymbol{u}_t-\boldsymbol{u}_{t-1}\right\|^2+\left(y_t-\boldsymbol{u}_t^{\top}\boldsymbol{x}_t\right)^2\right)\\
&= \min_{\boldsymbol{u}_{t-1}}\Bigg(\boldsymbol{u}_{t-1}^{\top}\left(cI+D_{t-1}\right)\boldsymbol{u}_{t-1}-2\boldsymbol{u}_{t-1}^{\top}\left(c\boldsymbol{u}_t+\boldsymbol{e}_{t-1}\right)+f_{t-1}+c\left\|\boldsymbol{u}_t\right\|^2\\
&\qquad+\left(y_t-\boldsymbol{u}_t^{\top}\boldsymbol{x}_t\right)^2\Bigg)\\
&= -\left(c\boldsymbol{u}_t+\boldsymbol{e}_{t-1}\right)^{\top}\left(cI+D_{t-1}\right)^{-1}\left(c\boldsymbol{u}_t+\boldsymbol{e}_{t-1}\right)+f_{t-1}+c\left\|\boldsymbol{u}_t\right\|^2+\left(y_t-\boldsymbol{u}_t^{\top}\boldsymbol{x}_t\right)^2\\
&= \boldsymbol{u}_t^{\top}\left(cI+\boldsymbol{x}_t\boldsymbol{x}_t^{\top}-c^2\left(cI+D_{t-1}\right)^{-1}\right)\boldsymbol{u}_t-2\boldsymbol{u}_t^{\top}\left[c\left(cI+D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1}+y_t\boldsymbol{x}_t\right]\\
&\quad-\boldsymbol{e}_{t-1}^{\top}\left(cI+D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1}+f_{t-1}+y_t^2\ .
\end{aligned}
$$

Using the Woodbury identity we continue to develop the last equation,

$$
\begin{aligned}
&= \boldsymbol{u}_t^{\top}\left(cI+\boldsymbol{x}_t\boldsymbol{x}_t^{\top}-c^2\left[c^{-1}I-c^{-2}\left(D_{t-1}^{-1}+c^{-1}I\right)^{-1}\right]\right)\boldsymbol{u}_t\\
&\quad-2\boldsymbol{u}_t^{\top}\left[\left(I+c^{-1}D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1}+y_t\boldsymbol{x}_t\right]\\
&\quad-\boldsymbol{e}_{t-1}^{\top}\left(cI+D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1}+f_{t-1}+y_t^2\\
&= \boldsymbol{u}_t^{\top}\left(\left(D_{t-1}^{-1}+c^{-1}I\right)^{-1}+\boldsymbol{x}_t\boldsymbol{x}_t^{\top}\right)\boldsymbol{u}_t-2\boldsymbol{u}_t^{\top}\left[\left(I+c^{-1}D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1}+y_t\boldsymbol{x}_t\right]\\
&\quad-\boldsymbol{e}_{t-1}^{\top}\left(cI+D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1}+f_{t-1}+y_t^2\ ,
\end{aligned}
$$

and indeed $D_t=\left(D_{t-1}^{-1}+c^{-1}I\right)^{-1}+\boldsymbol{x}_t\boldsymbol{x}_t^{\top}$, $\boldsymbol{e}_t=\left(I+c^{-1}D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1}+y_t\boldsymbol{x}_t$ and, $f_t=f_{t-1}-\boldsymbol{e}_{t-1}^{\top}\left(cI+D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1}+y_t^2$, as desired. ■

Appendix C. Proof of Theorem 3

We prove the theorem in four steps. First, we state a technical lemma, for which we define the following notation

$$
\begin{aligned}
&d_t\left(\boldsymbol{z},\boldsymbol{v}\right)=(\boldsymbol{z}-\boldsymbol{v})^{\top}\Sigma_t^{-1}(\boldsymbol{z}-\boldsymbol{v})\ ,\\
&d_{\tilde{t}}\left(\boldsymbol{z},\boldsymbol{v}\right)=(\boldsymbol{z}-\boldsymbol{v})^{\top}\tilde{\Sigma}_t^{-1}(\boldsymbol{z}-\boldsymbol{v})\ ,\\
&\chi_t=\boldsymbol{x}_t^{\top}\Sigma_{t-1}\boldsymbol{x}_t\ .
\end{aligned}
$$

Second, we define a telescopic sum and in Lemma 14 prove a lower bound for each element. Third, in Lemma 15 we upper bound one term of the telescopic sum, and finally, in the fourth step we combine all these parts to conclude the proof. Let us start with the technical lemma.

Lemma 13 *Let* $\tilde{\boldsymbol{w}}_t$ *and* $\tilde{\Sigma}_t$ *be defined in* (7) *and* (8), *then*

$$d_{t-1}\left(\boldsymbol{w}_{t-1},\boldsymbol{u}_{t-1}\right)-d_{\tilde{t}}\left(\tilde{\boldsymbol{w}}_t,\boldsymbol{u}_{t-1}\right)=\frac{1}{r}\ell_t-\frac{1}{r}g_t-\frac{\ell_t\chi_t}{r\left(r+\chi_t\right)}\ ,$$

where $\ell_t=\left(y_t-\boldsymbol{w}_{t-1}^\top\boldsymbol{x}_t\right)^2$ *and* $g_t=\left(y_t-\boldsymbol{u}_{t-1}^\top\boldsymbol{x}_t\right)^2$.

Proof We start by writing the distances explicitly,

$$\begin{aligned}&d_{t-1}\left(\boldsymbol{w}_{t-1},\boldsymbol{u}_{t-1}\right)-d_{\tilde{t}}\left(\tilde{\boldsymbol{w}}_t,\boldsymbol{u}_{t-1}\right)\\=&-\left(\boldsymbol{u}_{t-1}-\tilde{\boldsymbol{w}}_t\right)^\top\tilde{\Sigma}_t^{-1}\left(\boldsymbol{u}_{t-1}-\tilde{\boldsymbol{w}}_t\right)+\left(\boldsymbol{u}_{t-1}-\boldsymbol{w}_{t-1}\right)^\top\Sigma_{t-1}^{-1}\left(\boldsymbol{u}_{t-1}-\boldsymbol{w}_{t-1}\right)\ .\end{aligned}$$

Substituting $\tilde{\boldsymbol{w}}_t$ as appears in (8) the last equation becomes

$$\begin{aligned}&-\left(\boldsymbol{u}_{t-1}-\boldsymbol{w}_{t-1}\right)^\top\tilde{\Sigma}_t^{-1}\left(\boldsymbol{u}_{t-1}-\boldsymbol{w}_{t-1}\right)+2(\boldsymbol{u}_{t-1}-\boldsymbol{w}_{t-1})\tilde{\Sigma}_t^{-1}\Sigma_{t-1}\boldsymbol{x}_t\frac{(y_t-\boldsymbol{x}_t^\top\boldsymbol{w}_{t-1})}{r+\boldsymbol{x}_t^\top\Sigma_{t-1}\boldsymbol{x}_t}\\&-\left(\frac{(y_t-\boldsymbol{x}_t^\top\boldsymbol{w}_{t-1})}{r+\boldsymbol{x}_t^\top\Sigma_{t-1}\boldsymbol{x}_t}\right)^2\boldsymbol{x}_t^\top\Sigma_{t-1}\tilde{\Sigma}_t^{-1}\Sigma_{t-1}\boldsymbol{x}_t+\left(\boldsymbol{u}_{t-1}-\boldsymbol{w}_{t-1}\right)^\top\Sigma_{t-1}^{-1}\left(\boldsymbol{u}_{t-1}-\boldsymbol{w}_{t-1}\right)\ .\end{aligned}$$

Plugging $\tilde{\Sigma}_t$ as appears in (7) we get

$$\begin{aligned}&d_{t-1}\left(\boldsymbol{w}_{t-1},\boldsymbol{u}_{t-1}\right)-d_{\tilde{t}}\left(\tilde{\boldsymbol{w}}_t,\boldsymbol{u}_{t-1}\right)\\=&-\left(\boldsymbol{u}_{t-1}-\boldsymbol{w}_{t-1}\right)^\top\left(\Sigma_{t-1}^{-1}+\frac{1}{r}\boldsymbol{x}_t\boldsymbol{x}_t^\top\right)\left(\boldsymbol{u}_{t-1}-\boldsymbol{w}_{t-1}\right)\\&+2(\boldsymbol{u}_{t-1}-\boldsymbol{w}_{t-1})^\top\left(\Sigma_{t-1}^{-1}+\frac{1}{r}\boldsymbol{x}_t\boldsymbol{x}_t^\top\right)\Sigma_{t-1}\boldsymbol{x}_t\frac{(y_t-\boldsymbol{x}_t^\top\boldsymbol{w}_{t-1})}{r+\boldsymbol{x}_t^\top\Sigma_{t-1}\boldsymbol{x}_t}\\&-\frac{(y_t-\boldsymbol{x}_t^\top\boldsymbol{w}_{t-1})^2}{\left(r+\boldsymbol{x}_t^\top\Sigma_{t-1}\boldsymbol{x}_t\right)^2}\boldsymbol{x}_t^\top\Sigma_{t-1}\left(\Sigma_{t-1}^{-1}+\frac{1}{r}\boldsymbol{x}_t\boldsymbol{x}_t^\top\right)\Sigma_{t-1}\boldsymbol{x}_t\\&+\left(\boldsymbol{u}_{t-1}-\boldsymbol{w}_{t-1}\right)^\top\Sigma_{t-1}^{-1}\left(\boldsymbol{u}_{t-1}-\boldsymbol{w}_{t-1}\right)\ .\end{aligned}$$

Finally, we substitute $\ell_t = \left(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}\right)^2$, $g_t = \left(y_t - \boldsymbol{x}_t^\top \boldsymbol{u}_{t-1}\right)^2$ and $\chi_t = \boldsymbol{x}_t^\top \Sigma_{t-1} \boldsymbol{x}_t$. Rearranging the terms,

$$\begin{aligned}
& d_{t-1}\left(\boldsymbol{w}_{t-1}, \boldsymbol{u}_{t-1}\right) - d_{\tilde{t}}\left(\tilde{\boldsymbol{w}}_t, \boldsymbol{u}_{t-1}\right) \\
=& -\frac{1}{r}\left(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1} - \left(y_t - \boldsymbol{x}_t^\top \boldsymbol{u}_{t-1}\right)\right)^2 \\
& - \frac{2\left(y_t - \boldsymbol{x}_t^\top \boldsymbol{u}_{t-1} - \left(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}\right)\right)\left(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}\right)}{r + \chi_t}\left(1 + \frac{\chi_t}{r}\right) \\
& - \frac{\ell_t \chi_t}{\left(r + \chi_t\right)^2}\left(1 + \frac{\chi_t}{r}\right) \\
=& -\frac{1}{r}\ell_t + 2\left(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}\right)\left(y_t - \boldsymbol{x}_t^\top \boldsymbol{u}_{t-1}\right)\frac{1}{r} - \frac{1}{r}g_t \\
& + \frac{2\ell_t}{r + \chi_t}\left(1 + \frac{\chi_t}{r}\right) - \frac{\ell_t \chi_t}{r\left(r + \chi_t\right)} \\
& - 2\frac{\left(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}\right)\left(y_t - \boldsymbol{x}_t^\top \boldsymbol{u}_{t-1}\right)}{r + \chi_t}\left(1 + \frac{\chi_t}{r}\right) \\
=& \frac{1}{r}\ell_t - \frac{1}{r}g_t - \frac{\ell_t \chi_t}{r\left(r + \chi_t\right)} \ ,
\end{aligned}$$

which completes the proof. ■

We now define one element of the telescopic sum and lower bound it.

Lemma 14 *Denote*

$$\Delta_t = d_{t-1}\left(\boldsymbol{w}_{t-1}, \boldsymbol{u}_{t-1}\right) - d_t\left(\boldsymbol{w}_t, \boldsymbol{u}_t\right)$$

then

$$\Delta_t \geq \frac{1}{r}\left(\ell_t - g_t\right) - \ell_t \frac{\chi_t}{r(r + \chi_t)} + \boldsymbol{u}_{t-1}^\top \Sigma_{t-1}^{-1} \boldsymbol{u}_{t-1} - \boldsymbol{u}_t^\top \Sigma_t^{-1} \boldsymbol{u}_t - 2R_B \Lambda_i^{-1} \|\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\| \ ,$$

where $i-1$ is the number of restarts occurring before example t.

Proof We write Δ_t as a telescopic sum of four terms as follows

$$\begin{aligned}
\Delta_{t,1} &= d_{t-1}\left(\boldsymbol{w}_{t-1}, \boldsymbol{u}_{t-1}\right) - d_{\tilde{t}}\left(\tilde{\boldsymbol{w}}_t, \boldsymbol{u}_{t-1}\right) \\
\Delta_{t,2} &= d_{\tilde{t}}\left(\tilde{\boldsymbol{w}}_t, \boldsymbol{u}_{t-1}\right) - d_t\left(\tilde{\boldsymbol{w}}_t, \boldsymbol{u}_{t-1}\right) \\
\Delta_{t,3} &= d_t\left(\tilde{\boldsymbol{w}}_t, \boldsymbol{u}_{t-1}\right) - d_t\left(\boldsymbol{w}_t, \boldsymbol{u}_{t-1}\right) \\
\Delta_{t,4} &= d_t\left(\boldsymbol{w}_t, \boldsymbol{u}_{t-1}\right) - d_t\left(\boldsymbol{w}_t, \boldsymbol{u}_t\right) \ .
\end{aligned}$$

We lower bound each of the four terms. Since the value of $\Delta_{t,1}$ was computed in Lemma 13, we start with the second term. If no reset occurs then $\Sigma_t = \tilde{\Sigma}_t$ and $\Delta_{t,2} = 0$. Otherwise, we use the facts that $0 \preceq \tilde{\Sigma}_t \preceq I$ and $\Sigma_t = I$, and get

$$\begin{aligned}
\Delta_{t,2} &= \left(\tilde{\boldsymbol{w}}_t - \boldsymbol{u}_{t-1}\right)^\top \tilde{\Sigma}_t^{-1}\left(\tilde{\boldsymbol{w}}_t - \boldsymbol{u}_{t-1}\right) - \left(\tilde{\boldsymbol{w}}_t - \boldsymbol{u}_{t-1}\right)^\top \Sigma_t^{-1}\left(\tilde{\boldsymbol{w}}_t - \boldsymbol{u}_{t-1}\right) \\
&\geq \operatorname{Tr}\left(\left(\tilde{\boldsymbol{w}}_t - \boldsymbol{u}_{t-1}\right)\left(\tilde{\boldsymbol{w}}_t - \boldsymbol{u}_{t-1}\right)^\top (I - I)\right) = 0 \ .
\end{aligned}$$

To summarize, $\Delta_{t,2} \geq 0$. We can lower bound $\Delta_{t,3} \geq 0$ by using the fact that $\boldsymbol{w}_t$ is a projection of $\tilde{\boldsymbol{w}}_t$ onto a closed set (a ball of radius R_B around the origin), which by our assumption contains $\boldsymbol{u}_t$. Employing Corollary 3 of Herbster and Warmuth (2001) we get, $d_t\left(\tilde{\boldsymbol{w}}_t, \boldsymbol{u}_{t-1}\right) \geq d_t\left(\boldsymbol{w}_t, \boldsymbol{u}_{t-1}\right)$ and thus $\Delta_{t,3} \geq 0$.

Finally, we lower bound the fourth term $\Delta_{t,4}$,

$$\begin{aligned}\Delta_{t,4} &= \left(\boldsymbol{w}_t - \boldsymbol{u}_{t-1}\right)^\top \Sigma_t^{-1}\left(\boldsymbol{w}_t - \boldsymbol{u}_{t-1}\right) - \left(\boldsymbol{w}_t - \boldsymbol{u}_t\right)^\top \Sigma_t^{-1}\left(\boldsymbol{w}_t - \boldsymbol{u}_t\right)\\ &= \boldsymbol{u}_{t-1}^\top \Sigma_t^{-1} \boldsymbol{u}_{t-1} - \boldsymbol{u}_t^\top \Sigma_t^{-1} \boldsymbol{u}_t - 2\boldsymbol{w}_t^\top \Sigma_t^{-1}\left(\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\right) \ . \end{aligned} \tag{31}$$

We use the Hölder inequality and then the Cauchy-Schwartz inequality to get the following lower bound

$$\begin{aligned}-2\boldsymbol{w}_t^\top \Sigma_t^{-1}\left(\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\right) &= -2\mathrm{Tr}\left(\Sigma_t^{-1}\left(\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\right)\boldsymbol{w}_t^\top\right)\\ &\geq -2\lambda_{max}\left(\Sigma_t^{-1}\right)\boldsymbol{w}_t^\top\left(\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\right)\\ &\geq -2\lambda_{max}\left(\Sigma_t^{-1}\right)\|\boldsymbol{w}_t\|\|\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\| \ .\end{aligned}$$

Using the facts that $\|\boldsymbol{w}_t\| \leq R_B$ and that $\lambda_{max}\left(\Sigma_t^{-1}\right) = 1/\lambda_{min}\left(\Sigma_t\right) \leq \Lambda_i^{-1}$, where i is the current segment index, we get

$$-2\boldsymbol{w}_t^\top \Sigma_t^{-1}\left(\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\right) \geq -2\Lambda_i^{-1} R_B \|\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\| \ . \tag{32}$$

Substituting (32) in (31) and using $\Sigma_t \preceq \Sigma_{t-1}$ a lower bound is obtained,

$$\begin{aligned}\Delta_{t,4} &\geq \boldsymbol{u}_{t-1}^\top \Sigma_t^{-1} \boldsymbol{u}_{t-1} - \boldsymbol{u}_t^\top \Sigma_t^{-1} \boldsymbol{u}_t - 2R_B\Lambda_i^{-1}\|\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\|\\ &\geq \boldsymbol{u}_{t-1}^\top \Sigma_{t-1}^{-1} \boldsymbol{u}_{t-1} - \boldsymbol{u}_t^\top \Sigma_t^{-1} \boldsymbol{u}_t - 2R_B\Lambda_i^{-1}\|\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\| \ . \end{aligned} \tag{33}$$

Combining (33) with Lemma 13 concludes the proof. ■

Next we state an upper bound that will appear in one of the summands of the telescopic sum.

Lemma 15 *During the runtime of the ARCOR algorithm we have*

$$\sum_{t=t_i}^{t_i+T_i} \frac{\chi_t}{\left(\chi_t + r\right)} \leq \log\left(\det\left(\Sigma_{t_{i+1}-1}^{-1}\right)\right) = \log\left(\det\left(\left(\Sigma^i\right)^{-1}\right)\right) \ .$$

We remind the reader that t_i is the first example index after the ith restart, and T_i is the number of examples observed before the next restart. We also remind the reader the notation $\Sigma^i = \Sigma_{t_{i+1}-1}$ is the covariance matrix just before the next restart.

The proof of the lemma is similar to the proof of Lemma 4 by Crammer et al. (2009) and thus omitted. We now put all the pieces together and prove Theorem 3.

Proof We bound the sum $\sum_t \Delta_t$ from above and below, and start with an upper bound using the property of telescopic sum,

$$\sum_t \Delta_t = \sum_t \left[d_{t-1}\left(\boldsymbol{w}_{t-1}, \boldsymbol{u}_{t-1}\right) - d_t\left(\boldsymbol{w}_t, \boldsymbol{u}_t\right)\right] = d_0\left(\boldsymbol{w}_0, \boldsymbol{u}_0\right) - d_T\left(\boldsymbol{w}_T, \boldsymbol{u}_T\right) \leq d_0\left(\boldsymbol{w}_0, \boldsymbol{u}_0\right) \ . \tag{34}$$

We compute a lower bound by applying Lemma 14,

$$\sum_t \Delta_t \geq \sum_t \Bigg(\frac{1}{r}\left(\ell_t - g_t\right) - \ell_t \frac{\chi_t}{r(r+\chi_t)} \\ + \boldsymbol{u}_{t-1}^\top \Sigma_{t-1}^{-1} \boldsymbol{u}_{t-1} - \boldsymbol{u}_t^\top \Sigma_t^{-1} \boldsymbol{u}_t - 2R_B \Lambda_{i(t)}^{-1} \|\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\| \Bigg) ,$$

where $i(t)$ is the number of restarts occurred before observing the tth example. Continuing to develop the last equation we obtain

$$\begin{aligned} \sum_t \Delta_t \geq & \frac{1}{r}\sum_t \ell_t - \frac{1}{r}\sum_t g_t - \sum_t \ell_t \frac{\chi_t}{r(r+\chi_t)} + \sum_t \left(\boldsymbol{u}_{t-1}^\top \Sigma_{t-1}^{-1} \boldsymbol{u}_{t-1} - \boldsymbol{u}_t^\top \Sigma_t^{-1} \boldsymbol{u}_t \right) \\ & - \sum_t 2R_B \Lambda_{i(t)}^{-1} \|\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\| \\ = & \frac{1}{r}\sum_t \ell_t - \frac{1}{r}\sum_t g_t - \sum_t \ell_t \frac{\chi_t}{r(r+\chi_t)} + \boldsymbol{u}_0^\top \Sigma_0^{-1} \boldsymbol{u}_0 - \boldsymbol{u}_T^\top \Sigma_T^{-1} \boldsymbol{u}_T \\ & - 2R_B \sum_t \Lambda_{i(t)}^{-1} \|\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\| \ . \end{aligned} \tag{35}$$

Combining (34) with (35) and using $d_0\left(\boldsymbol{w}_0, \boldsymbol{u}_0\right) = \boldsymbol{u}_0^\top \Sigma_0^{-1} \boldsymbol{u}_0$ (as $\boldsymbol{w}_0 = \boldsymbol{0}$),

$$\frac{1}{r}\sum_t \ell_t - \frac{1}{r}\sum_t g_t - \sum_t \ell_t \frac{\chi_t}{r(r+\chi_t)} - \boldsymbol{u}_T^\top \Sigma_T^{-1} \boldsymbol{u}_T - 2R_B \sum_t \Lambda_{i(t)}^{-1} \|\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\| \leq 0 \ .$$

Rearranging the terms of the last inequality,

$$\sum_t \ell_t \leq \sum_t g_t + \sum_t \ell_t \frac{\chi_t}{r+\chi_t} + r\boldsymbol{u}_T^\top \Sigma_T^{-1} \boldsymbol{u}_T + 2R_B r \sum_t \frac{1}{\Lambda_{i(t)}} \|\boldsymbol{u}_{t-1} - \boldsymbol{u}_t\| \ . \tag{36}$$

Since $\|\boldsymbol{w}_t\| \leq R_B$ and we assume that $\|\boldsymbol{x}_t\| = 1$ and $\sup_t |y_t| = Y$, we get that $\sup_t \ell_t \leq 2(R_B^2 + Y^2)$. Substituting the last inequality in Lemma 15, we bound the second term in the right-hand-side of (36),

$$\begin{aligned} \sum_t \ell_t \frac{\chi_t}{r+\chi_t} &= \sum_i^n \sum_{t=t_i}^{t_i+T_i} \ell_t \frac{\chi_t}{r+\chi_t} \\ &\leq \sum_i^n \left(\sup_t \ell_t\right) \log\det\left(\left(\Sigma^i\right)^{-1}\right) \\ &\leq 2\left(R_B^2 + Y^2\right) \sum_i^n \log\det\left(\left(\Sigma^i\right)^{-1}\right) \ , \end{aligned}$$

which completes the proof. ■

Appendix D. Proof of Lemma 7

Proof We first use the Woodbury identity to get the following two identities

$$\begin{aligned}
D_t^{-1} &= \left[\left(D_{t-1}^{-1}+c^{-1}I\right)^{-1}+\boldsymbol{x}_t\boldsymbol{x}_t^\top\right]^{-1} \\
&= D_{t-1}^{-1}+c^{-1}I-\frac{\left(D_{t-1}^{-1}+c^{-1}I\right)\boldsymbol{x}_t\boldsymbol{x}_t^\top\left(D_{t-1}^{-1}+c^{-1}I\right)}{1+\boldsymbol{x}_t^\top\left(D_{t-1}^{-1}+c^{-1}I\right)\boldsymbol{x}_t} \\
\left(I+c^{-1}D_{t-1}\right)^{-1} &= I-c^{-1}\left(D_{t-1}^{-1}+c^{-1}I\right)^{-1} \ .
\end{aligned}$$

Multiplying both identities with each other we get

$$\begin{aligned}
& D_t^{-1}\left(I+c^{-1}D_{t-1}\right)^{-1} \\
= & \left[D_{t-1}^{-1}+c^{-1}I-\frac{\left(D_{t-1}^{-1}+c^{-1}I\right)\boldsymbol{x}_t\boldsymbol{x}_t^\top\left(D_{t-1}^{-1}+c^{-1}I\right)}{1+\boldsymbol{x}_t^\top\left(D_{t-1}^{-1}+c^{-1}I\right)\boldsymbol{x}_t}\right]\left[I-c^{-1}\left(D_{t-1}^{-1}+c^{-1}I\right)^{-1}\right] \\
= & D_{t-1}^{-1}-\frac{\left(D_{t-1}^{-1}+c^{-1}I\right)\boldsymbol{x}_t\boldsymbol{x}_t^\top D_{t-1}^{-1}}{1+\boldsymbol{x}_t^\top\left(D_{t-1}^{-1}+c^{-1}I\right)\boldsymbol{x}_t} \ , \qquad (37)
\end{aligned}$$

and, similarly, we multiply the identities in the other order and get

$$\begin{aligned}
& \left(I+c^{-1}D_{t-1}\right)^{-1}D_t^{-1} \\
= & \left[I-c^{-1}\left(D_{t-1}^{-1}+c^{-1}I\right)^{-1}\right]\left[D_{t-1}^{-1}+c^{-1}I-\frac{\left(D_{t-1}^{-1}+c^{-1}I\right)\boldsymbol{x}_t\boldsymbol{x}_t^\top\left(D_{t-1}^{-1}+c^{-1}I\right)}{1+\boldsymbol{x}_t^\top\left(D_{t-1}^{-1}+c^{-1}I\right)\boldsymbol{x}_t}\right] \\
= & D_{t-1}^{-1}-\frac{D_{t-1}^{-1}\boldsymbol{x}_t\boldsymbol{x}_t^\top\left(D_{t-1}^{-1}+c^{-1}I\right)}{1+\boldsymbol{x}_t^\top\left(D_{t-1}^{-1}+c^{-1}I\right)\boldsymbol{x}_t} \ . \qquad (38)
\end{aligned}$$

Finally, from (37) we get

$$\begin{aligned}
& \left(I+c^{-1}D_{t-1}\right)^{-1}D_t^{-1}\boldsymbol{x}_t\boldsymbol{x}_t^\top D_t^{-1}\left(I+c^{-1}D_{t-1}\right)^{-1}-D_{t-1}^{-1} \\
& \quad +\left(I+c^{-1}D_{t-1}\right)^{-1}\left[D_t^{-1}\left(I+c^{-1}D_{t-1}\right)^{-1}+c^{-1}I\right] \\
= & \left(I+c^{-1}D_{t-1}\right)^{-1}D_t^{-1}\boldsymbol{x}_t\boldsymbol{x}_t^\top D_t^{-1}\left(I+c^{-1}D_{t-1}\right)^{-1} \\
& \quad -D_{t-1}^{-1}+\left[I-c^{-1}\left(D_{t-1}^{-1}+c^{-1}I\right)^{-1}\right]\left[D_{t-1}^{-1}+c^{-1}I-\frac{\left(D_{t-1}^{-1}+c^{-1}I\right)\boldsymbol{x}_t\boldsymbol{x}_t^\top D_{t-1}^{-1}}{1+\boldsymbol{x}_t^\top\left(D_{t-1}^{-1}+c^{-1}I\right)\boldsymbol{x}_t}\right] \ .
\end{aligned}$$

We further develop the last equality and use (37) and (38) in the second equality below,

$$\begin{aligned}
&= \left(I + c^{-1}D_{t-1}\right)^{-1} D_t^{-1}\boldsymbol{x}_t\boldsymbol{x}_t^\top D_t^{-1}\left(I + c^{-1}D_{t-1}\right)^{-1} - D_{t-1}^{-1} \\
&\quad + D_{t-1}^{-1} - \frac{D_{t-1}^{-1}\boldsymbol{x}_t\boldsymbol{x}_t^\top D_{t-1}^{-1}}{1 + \boldsymbol{x}_t^\top\left(D_{t-1}^{-1} + c^{-1}I\right)\boldsymbol{x}_t} \\
&= \left[D_{t-1}^{-1} - \frac{D_{t-1}^{-1}\boldsymbol{x}_t\boldsymbol{x}_t^\top\left(D_{t-1}^{-1} + c^{-1}I\right)}{1 + \boldsymbol{x}_t^\top\left(D_{t-1}^{-1} + c^{-1}I\right)\boldsymbol{x}_t}\right]\boldsymbol{x}_t\boldsymbol{x}_t^\top\left[D_{t-1}^{-1} - \frac{\left(D_{t-1}^{-1} + c^{-1}I\right)\boldsymbol{x}_t\boldsymbol{x}_t^\top D_{t-1}^{-1}}{1 + \boldsymbol{x}_t^\top\left(D_{t-1}^{-1} + c^{-1}I\right)\boldsymbol{x}_t}\right] \\
&\quad - \frac{D_{t-1}^{-1}\boldsymbol{x}_t\boldsymbol{x}_t^\top D_{t-1}^{-1}}{1 + \boldsymbol{x}_t^\top\left(D_{t-1}^{-1} + c^{-1}I\right)\boldsymbol{x}_t} \\
&= -\frac{\boldsymbol{x}_t^\top\left(D_{t-1}^{-1} + c^{-1}I\right)\boldsymbol{x}_t D_{t-1}^{-1}\boldsymbol{x}_t\boldsymbol{x}_t^\top D_{t-1}^{-1}}{\left(1 + \boldsymbol{x}_t^\top\left(D_{t-1}^{-1} + c^{-1}I\right)\boldsymbol{x}_t\right)^2} \quad \preceq \quad 0\ .
\end{aligned}$$

■

Appendix E. Derivations for Theorem 8

$$\begin{aligned}
&(y_t - \hat{y}_t)^2 + \min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_{t-1}} Q_{t-1}\left(\boldsymbol{u}_1,\ldots,\boldsymbol{u}_{t-1}\right) - \min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_t} Q_t\left(\boldsymbol{u}_1,\ldots,\boldsymbol{u}_t\right) \\
=\ & (y_t - \hat{y}_t)^2 - \boldsymbol{e}_{t-1}^\top D_{t-1}^{-1}\boldsymbol{e}_{t-1} + f_{t-1} + \boldsymbol{e}_t^\top D_t^{-1}\boldsymbol{e}_t - f_t \\
=\ & (y_t - \hat{y}_t)^2 - \boldsymbol{e}_{t-1}^\top D_{t-1}^{-1}\boldsymbol{e}_{t-1} \\
& + \left(\left(I + c^{-1}D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1} + y_t\boldsymbol{x}_t\right)^\top D_t^{-1}\left(\left(I + c^{-1}D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1} + y_t\boldsymbol{x}_t\right) \\
& + \boldsymbol{e}_{t-1}^\top\left(cI + D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1} - y_t^2\ ,
\end{aligned}$$

where the last equality follows from (12) and (13). We proceed to develop the last equality,

$$\begin{aligned}
=\ & (y_t - \hat{y}_t)^2 - \boldsymbol{e}_{t-1}^\top D_{t-1}^{-1}\boldsymbol{e}_{t-1} + \boldsymbol{e}_{t-1}^\top\left(I + c^{-1}D_{t-1}\right)^{-1} D_t^{-1}\left(I + c^{-1}D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1} \\
& + 2y_t\boldsymbol{x}_t^\top D_t^{-1}\left(I + c^{-1}D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1} + y_t^2\boldsymbol{x}_t^\top D_t^{-1}\boldsymbol{x}_t + \boldsymbol{e}_{t-1}^\top\left(cI + D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1} - y_t^2 \\
=\ & (y_t - \hat{y}_t)^2 + \boldsymbol{e}_{t-1}^\top\Bigg(- D_{t-1}^{-1} + \left(I + c^{-1}D_{t-1}\right)^{-1} D_t^{-1}\Big(I + c^{-1}D_{t-1}\Big)^{-1} \\
& + c^{-1}\left(I + c^{-1}D_{t-1}\right)^{-1}\Bigg)\boldsymbol{e}_{t-1} \\
& + 2y_t\boldsymbol{x}_t^\top D_t^{-1}\left(I + c^{-1}D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1} + y_t^2\boldsymbol{x}_t^\top D_t^{-1}\boldsymbol{x}_t - y_t^2 \\
=\ & (y_t - \hat{y}_t)^2 + \boldsymbol{e}_{t-1}^\top\Bigg(- D_{t-1}^{-1} + \Big(I + c^{-1}D_{t-1}\Big)^{-1}\Big[D_t^{-1}\left(I + c^{-1}D_{t-1}\right)^{-1} + c^{-1}I\Big]\Bigg)\boldsymbol{e}_{t-1} \\
& + 2y_t\boldsymbol{x}_t^\top D_t^{-1}\left(I + c^{-1}D_{t-1}\right)^{-1}\boldsymbol{e}_{t-1} + y_t^2\boldsymbol{x}_t^\top D_t^{-1}\boldsymbol{x}_t - y_t^2\ .
\end{aligned}$$

Appendix F. Proof of Corollary 12

Proof Plugging Lemma 9 in Theorem 8 we have for all $(\boldsymbol{u}_1 \ldots \boldsymbol{u}_T)$,

$$\begin{aligned} L_T(\text{LASER}) \leq\ & b\left\|\boldsymbol{u}_1\right\|^2 + cV^{(2)} + L_T(\{\boldsymbol{u}_t\}) + Y^2 \ln\left|\frac{1}{b}D_T\right| + c^{-1}Y^2\sum_{t=1}^{T}\text{Tr}\left(D_{t-1}\right) \\ \leq\ & b\left\|\boldsymbol{u}_1\right\|^2 + L_T(\{\boldsymbol{u}_t\}) + Y^2 \ln\left|\frac{1}{b}D_T\right| + c^{-1}Y^2\text{Tr}\left(D_0\right) + cV^{(2)} \\ & + c^{-1}Y^2Td\max\left\{\frac{3X^2+\sqrt{X^4+4X^2c}}{2}, b+X^2\right\}\ , \end{aligned}$$

where the last inequality follows from Lemma 11. The term $c^{-1}Y^2\text{Tr}\left(D_0\right)$ does not depend on T, because

$$c^{-1}Y^2\text{Tr}\left(D_0\right) = c^{-1}Y^2d\frac{bc}{c-b} = \frac{\varepsilon}{1-\varepsilon}Y^2d\ .$$

To show (29), note that

$$V^{(2)} \leq T\frac{\sqrt{2}Y^2dX}{\mu^{3/2}} \Leftrightarrow \mu \leq \left(\frac{\sqrt{2}Y^2dXT}{V^{(2)}}\right)^{2/3} = c\ .$$

We thus have that the right term of (27) is upper bounded,

$$\begin{aligned} \max\left\{\frac{3X^2+\sqrt{X^4+4X^2c}}{2}, b+X^2\right\} \leq\ & \max\left\{\frac{3X^2+\sqrt{8X^2c}}{2}, b+X^2\right\} \\ \leq\ \max\left\{\sqrt{8X^2c}, b+X^2\right\} \leq 2X\sqrt{2c}\ . & \end{aligned}$$

Using this bound and plugging the value of c from (28) we bound (27),

$$\begin{aligned} & \left(\frac{\sqrt{2}TY^2dX}{V^{(2)}}\right)^{2/3} V^{(2)} + Y^2Td2X\sqrt{2\left(\frac{\sqrt{2}TY^2dX}{V^{(2)}}\right)^{-2/3}} \\ & \quad = 3\left(\sqrt{2}TY^2dX\right)^{2/3}\left(V^{(2)}\right)^{1/3}\ , \end{aligned}$$

which concludes the proof. ■

Appendix G. Details for the bound (30)

To show the bound (30), note that

$$V^{(2)} \geq T\frac{Y^2dM}{\mu^2} \Leftrightarrow \mu \geq \sqrt{\frac{TY^2dM}{V^{(2)}}} = c\ .$$

We thus have that the right term of (27) is upper bounded as follows

$$
\begin{aligned}
&\max\left\{\frac{3X^2+\sqrt{X^4+4X^2c}}{2}, b+X^2\right\} \leq \max\left\{3X^2, \sqrt{X^4+4X^2c}, b+X^2\right\} \\
\leq\ &\max\left\{3X^2, \sqrt{2}X^2, \sqrt{8X^2c}, b+X^2\right\} = \sqrt{8X^2}\max\left\{\frac{3X^2}{\sqrt{8X^2}}, \sqrt{c}, \frac{b+X^2}{\sqrt{8X^2}}\right\} \\
=\ &\sqrt{8X^2}\sqrt{\max\left\{\frac{(3X^2)^2}{8X^2}, c, \frac{(b+X^2)^2}{8X^2}\right\}} = \sqrt{8X^2}\sqrt{\max\left\{\mu, c\right\}} \leq \sqrt{8X^2}\sqrt{\mu} = M\ .
\end{aligned}
$$

Using this bound and plugging $c=\sqrt{Y^2dMT/V^{(2)}}$ we bound (27),

$$
\sqrt{\frac{Y^2dMT}{V^{(2)}}}V^{(2)} + \frac{1}{\sqrt{\frac{Y^2dMT}{V^{(2)}}}}TdY^2M = 2\sqrt{Y^2dMTV^{(2)}}\ .
$$

References

Dmitry Adamskiy, Wouter M. Koolen, Alexey V. Chernov, and Vladimir Vovk. A closer look at adaptive regret. In *The 23rd International Conference on Algorithmic Learning Theory*, pages 290–304, 2012.

Peter Auer and Manfred K. Warmuth. Tracking the best disjunction. *Electronic Colloquium on Computational Complexity (ECCC)*, 7(70), 2000.

K.S. Azoury and M.W. Warmuth. Relative loss bounds for on-line density estimation with the exponential family of distributions. *Machine Learning*, 43(3):211–246, 2001.

N. J. Bershad. Analysis of the normalized lms algorithm with gaussian inputs. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34(4):793–806, 1986.

R. R. Bitmead and B. D. O. Anderson. Performance of adaptive estimation algorithms in dependent random environments. *IEEE Transactions on Automatic Control*, 25:788–794, 1980.

Steven Busuttil and Yuri Kalnishkan. Online regression competitive with changing predictors. In *The 18th International Conference on Algorithmic Learning Theory*, pages 181–195, 2007.

Giovanni Cavallanti, Nicolò Cesa-Bianchi, and Claudio Gentile. Tracking the best hyperplane with a simple budget perceptron. *Machine Learning*, 69(2-3):143–167, 2007.

Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games.* Cambridge University Press, New York, NY, USA, 2006.

Nicolo Cesa-Bianchi, Philip M. Long, and Manfred K. Warmuth. Worst case quadratic loss bounds for on-line prediction of linear functions by gradient descent. Technical Report IR-418, University of California, Santa Cruz, CA, USA, 1993.

Nicoló Cesa-Bianchi, Alex Conconi, and Claudio Gentile. A second-order perceptron algorithm. *Siam Journal of Commutation*, 34(3):640–668, 2005.

Min-Shin Chen and Jia-Yush Yen. Application of the least squares algorithm to the observer design for linear time-varying systems. *Automatic Control, IEEE Transactions on*, 44(9): 1742 –1745, sep 1999.

K. Crammer, M. Dredze, and F. Pereira. Exact confidence-weighted learning. In *Advances in Neural Information Processing Systems 22*, 2008.

K. Crammer, A. Kulesza, and M. Dredze. Adaptive regularization of weighted vectors. In *Advances in Neural Information Processing Systems 23*, 2009.

M. Dredze, K. Crammer, and F. Pereira. Confidence-weighted linear classification. In *Proceedings of the Twenty-Five International Conference on Machine Learning*, 2008.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. In *Proceedings of the 23rd Annual Conference on Computational Learning Theory*, pages 257–269, 2010.

A. Feuer and E. Weinstein. Convergence analysis of lms filters with uncorrelated gaussian data. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 33(1):222–230, 1985.

Jurgen Forster. On relative loss bounds in generalized linear regression. In *Fundamentals of Computation Theory (FCT)*, 1999. ISBN 3-540-66412-2.

Dean P. Foster. Prediction in the worst case. *The Annals of Statistics*, 19(2):1084–1090, 1991.

Gene H. Golub and Charles F. Van Loan. *Matrix Computations (3rd ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996. ISBN 0-8018-5414-8.

S.G. Goodhart, K.J. Burnham, and D.J.G. James. Logical covariance matrix reset in self-tuning control. *Mechatronics*, 1(3):339 – 351, 1991.

G.C. Goodwin, E.K. Teoh, and H. Elliott. Deterministic convergence of a self-tuning regulator with covariance resetting. *Control Theory and Applications, IEE Proceedings D*, 130(1):6 –8, 83.

Monson H. Hayes. 9.4: Recursive least squares. In *Statistical Digital Signal Processing and Modeling*, page 541, 1996. ISBN 0-471-59431-8.

Mark Herbster and Manfred K. Warmuth. Tracking the best linear predictor. *Journal of Machine Learning Research*, 1:281–309, 2001.

Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

Jyrki Kivinen and Manfred K. Warmuth. Exponential gradient versus gradient descent for linear predictors. *Information and Computation*, 132:132–163, 1997.

Jyrki Kivinen, Alex J. Smola, and Robert C. Williamson. Online learning with kernels. In *Advances in Neural Information Processing Systems 14*, pages 785–792, 2001.

Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Inf. Comput.*, 108(2):212–261, 1994.

H. Brendan McMahan and Matthew J. Streeter. Adaptive bound optimization for online convex optimization. In *Proceedings of the 23rd Annual Conference on Computational Learning Theory*, pages 244–256, 2010.

Edward Moroshko and Koby Crammer. Weighted last-step min-max algorithm with improved sub-logarithmic regret. In *The 23rd International Conference on Algorithmic Learning Theory*, 2012.

Edward Moroshko and Koby Crammer. A last-step regression algorithm for non-stationary online learning. In *The Sixteenth International Conference on Artificial Intelligence and Statistics*, 2013.

Mario E. Salgado, Graham C. Goodwin, and Richard H. Middleton. Modified least squares algorithm incorporating exponential resetting and forgetting. *International Journal of Control*, 47(2):477 –491, 1988.

Dan Simon. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches.* Wiley-Interscience, 2006. ISBN 0471708585.

Hong-Seok Song, Kwanghee Nam, and P. Mutschler. Very fast phase angle estimation algorithm for a single-phase system having sudden phase angle jumps. In *Industry Applications Conference. 37th IAS Annual Meeting*, volume 2, pages 925 – 931, 2002.

Eiji Takimoto and Manfred K. Warmuth. The last-step minimax algorithm. In *The 11th International Conference on Algorithmic Learning Theory*, pages 279–290, 2000.

Nina Vaits and Koby Crammer. Re-adapting the regularization of weights for non-stationary regression. In *The 22nd International Conference on Algorithmic Learning Theory*, 2011.

Volodimir Vovk. Aggregating strategies. In *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 371–383. Morgan Kaufmann, 1990.

Volodya Vovk. Competitive on-line linear regression. In *Advances in Neural Information Processing Systems 10*, 1997.

Volodya Vovk. Competitive on-line statistics. *International Statistical Review*, 69, 2001.

Bernard Widrow and Marcian E. Hoff. Adaptive switching circuits. In *IRE WESCON Convention Record, Part 4*, pages 96–104, 1960.