New Min-Max Algorithms and Analysis for Online Regression

Edward Moroshko

New Min-Max Algorithms and Analysis for Online Regression

Research Thesis

In Partial Fulfillment of The
Requirements for the Degree of
Master of Science in Electrical Engineering

Edward Moroshko

Submitted to the Senate of
the Technion — Israel Institute of Technology
Tammuz, 5773 Haifa June 2013

The Research Thesis Was Done Under The Supervision of Prof. Koby Crammer in the Faculty of Electrical Engineering.

I would like to thank Koby for his remarkable guidance and support. I also thank Koby for funding my travels to ALT 2012 and AISTATS 2013 conferences.
Thanks to my family for the support.
Special thanks to my fiancee Ephrath for her love, support and patience along the way.

Publications

Parts of this work were published in
Weighted Last-Step Min-Max Algorithm with Improved Sub-Logarithmic Regret.
Edward Moroshko and Koby Crammer. 2012. In ALT.
A Last-Step Regression Algorithm for Non-Stationary Online Learning.
Edward Moroshko and Koby Crammer. 2013. In AISTATS.

Contents

List of Figures

List of Tables

Abstract

In online learning the performance of an algorithm is typically compared to the performance of a fixed function from some class, using a quantity called regret. The purpose of a learning algorithm is to have a regret as small as possible. About a decade ago, the last-step min-max approach has been proposed for online regression, where the algorithm tries to minimize the maximal (worst-case) regret with respect to the best-performing linear function. In fact, to make the min-max optimization problem well defined it has been assumed that the choices of the adversary are bounded, yielding artificially only extreme cases. In this work we introduce a new algorithm that weighs the examples in such a way that the min-max problem will be well defined. We provide analysis of the algorithm and develop new regret bounds that are better in some situations than other state-of-the-art bounds.

In many real-world problems there is no fixed best target function during the runtime of the algorithm. Instead, the best (local) target function is drifting over time. In this work we extend the last-step min-max approach to the drifting setting and introduce a new algorithm which is a last-step min-max optimal in context of a drift. We analyze the algorithm in the worst-case regret framework and show that it maintains an average loss close to that of the best slowly changing sequence of linear functions, as long as the total of drift is sublinear. We show formally that in some situations our bound is better than existing bounds, and synthetic simulations demonstrate the advantages of our algorithm over previous approaches, especially in a worst-case constant drift setting.

Abbreviations and Notations

AA	—	Aggregating Algorithm
AAR	—	Aggregating Algorithm for Regression
$ARCOR$	—	Adaptive Regularization with Covariance Reset
$AROW$	—	Adaptive Regularization Of Weights
$AROWR$	—	Adaptive Regularization Of Weights for Regression
$CR-RLS$	—	Covariance Reset Recursive Least Squares
$LASER$	—	Last-Step Adaptive Regressor
$NLMS$	—	Normalized Least Mean Square
RLS	—	Recursive Least Squares
$WEMM$	—	Weighted Min-Max
$\|\mathbf{u}\|$	—	ℓ_2-norm of the vector $\mathbf{u}$

Chapter 1

Introduction

We consider the online learning regression problem, in which a learning algorithm tries to predict real numbers in a sequence of rounds given some side-information or inputs $\mathbf{x}_t \in \mathbb{R}^d$. Real-world example applications for these algorithms are weather or stock market predictions. The goal of an algorithm is to have a small discrepancy between its predictions and the associated outcomes or labels $y_t \in \mathbb{R}$, which can be chosen by an adversary. This discrepancy is measured with a loss function, such as the square loss. It is common to evaluate algorithms by their regret, the difference between the cumulative loss of an algorithm with the cumulative loss of any function taken from some class.

Many algorithms were proposed for this problem. Gradient descent based algorithms were analysed by Cesa-Bianchi et al. [1993], and it was shown that a regret of $\mathcal{O}(\sqrt{T})$ can be achieved where T is the total number of rounds. Kivinen and Warmuth [1997] proposed the Exponentiated Gradient Algorithm that uses the components of the gradient in the exponents of factors that are used in updating the weight vector multiplicatively. This algorithm achieves also a regret of $\mathcal{O}(\sqrt{T})$, but has a much smaller loss if only few components of the input are relevant for the predictions. Similar regret was shown in the online convex programming framework [Zinkevich, 2003].

In this work we focus on second order algorithms. These algorithms maintain a weight vector used for prediction and a covariance-like matrix which functions like adaptive learning rate. The RLS algorithm [Hayes, 1996], Ridge-Regression [Foster, 1991] and the AAR algorithm [Vovk, 1997, 2001] are examples of second order algorithms, all achieve a regret of $\mathcal{O}(\log T)$, yet with different multiplicative factors of the logarithm. Another similar algorithm with similar regret (AROWR) proposed by Vaits and Crammer [2011] and analysed by Crammer et al. [2012b]. Recently, Orabona et al. [2012] showed that beyond logarithmic regret bound can be achieved when the total best linear model loss is sublinear in T.

Forster [1999] proposed a last-step min-max algorithm for online regression that makes a prediction assuming it is the last example to be observed, and the goal of the algorithm is indeed to minimize the maximal (worst-case) regret with respect to linear functions. Takimoto and Warmuth [2000] used this idea for the online density estimation problem. The resulting optimization problem of Forster [1999] was convex in both choice of the algorithm and the choice of the adversary, yielding an unbounded optimization problem. Forster [1999] circumvented this problem by assuming a bound Y over the choices of the adversary that should be known to the algorithm, yet his analysis is for the version with no bound, leading to a regret of $\mathcal{O}(\log T)$.

We propose a modified last-step min-max algorithm with weights over examples, that are controlled in a way to obtain a problem that is concave over the choices of the adversary and convex over the choices of the algorithm. We analyze our algorithm and show a logarithmic-regret that may have a better multiplicative factor than the analysis of Forster [1999] or other algorithms. We derive additional analysis that is logarithmic in the loss of the reference function, rather than the number of rounds T. This behaviour was recently given by Orabona et al. [2012] for the Online Newton Step algorithm [Hazan et al., 2006]. Yet, their bound [Orabona et al., 2012] has a similar multiplicative factor to that of Forster [1999], while our bound has a potentially better multiplicative factor and it has the same dependency in the cumulative loss of the reference function as Orabona et al. [2012]. Additionally, our algorithm and analysis are totally free of assuming the bound Y or knowing its value.

Competing with the best *fixed* function might not suffice for some problems. In many real-world applications, the true target function is not fixed, but is slowly drifting over time. This setting is called non-stationary setting. Consider a function designed to rate movies for recommender systems given some features. Over time a rate of a movie may change as more movies are released or the season changes. Furthermore, the very own personal-taste of a user may change as well.

These reasons led to the development of algorithms and accompanying analysis for drifting and shifting settings (see for example Auer and Warmuth [2000], Herbster and Warmuth [2001], Kivinen et al. [2001], Cavallanti et al. [2007] and the references therein). The goal of an algorithm in this settings 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. Often such a sequence is either drifting, where each function is close in some sense to its predecessor, or shifting, where conceptually the sequence can be partitioned into few segments, for each there is a single function that performs well on all examples of that segment.

One approach used in previous algorithms for non-stationary setting is based on gradient descent with additional control on the norm of the weight-vector used for prediction. Bounding the weight vector was performed either by projecting it into a bonded set [Herbster and Warmuth, 2001], shrinking it by multiplication [Kivinen et al., 2001, Cavallanti et al., 2007] or subtraction of previously seen examples [Cavallanti et al., 2007], which was motivated from reduction of memory when using kernels [Crammer et al., 2003]. Since the tracking ability is naturally connected to a weakened dependence on the past, memory boundedness could be viewed as a way to obtain a good performance in non-stationary setting. Another approach, which is used in second order algorithms, is based on resetting the covariance matrix, and in this way forgetting the history. The Covariance Reset RLS algorithm (CR-RLS) [Salgado et al., 1988, Goodwin et al., 1983, Chen and Yen, 1999] was designed for adaptive filtering and it makes covariance reset every fixed amount of rounds. The ARCOR algorithm [Vaits and Crammer, 2011] is similar to the CR-RLS algorithm but it resets the covariance matrix each time the smallest eigenvalue of the covariance matrix reaches a certain threshold. Thus, the ARCOR algorithm performs resets based on the actual properties of the data: the eigenspectrum of the covariance matrix. In addition, the ARCOR algorithm projects the weight vector into a bounded set in order to prevent it from going too far in some directions.

We take a different route and derive an algorithm for the non-stationary setting based on the last-step min-max approach of Forster [1999]. Yet, our algorithm is min-max optimal when a drift is allowed. As opposed to the derivation of the last-step min-max predictor for a fixed competitor, 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 analyze the algorithm in the worst-case regret framework and show that the algorithm maintains an average loss close to that of the best slowly changing sequence of functions, as long as the total drift is sublinear in the number of rounds T. Additionally, when no drift is introduced (stationary setting) our algorithm suffers logarithmic regret, as for the algorithm of Forster [1999].

Finally, synthetic simulations show the advantages of our algorithm.

Chapter 2

The online regression setting

In this thesis we work in the online setting for regression evaluated with the squared loss. Online algorithms work in rounds or iterations. On each iteration an online algorithm receives an instance $\mathbf{x}_t \in \mathbb{R}^d$ and predicts a real value $\hat{y}_t \in \mathbb{R}$, it then receives a label $y_t \in \mathbb{R}$, possibly chosen by an adversary, suffers loss $\ell_t(\text{alg}) = \ell\left(y_t, \hat{y}_t\right) = \left(\hat{y}_t - y_t\right)^2$, updates its prediction rule, and proceeds to the next round. The cumulative loss suffered by an algorithm over T iterations is $L_T(\text{alg}) = \sum_{t=1}^{T} \ell_t(\text{alg})$. The goal of an algorithm is to perform well compared to any predictor from some function class.

A common choice, which we adopt as well, is to compare the performance of an algorithm with respect to *a single* function, or specifically a single linear function, $f(\mathbf{x}) = \mathbf{x}^\top \mathbf{u}$, parameterized by a vector $\mathbf{u} \in \mathbb{R}^d$. We call this setting the **stationary** setting. Denote by $\ell_t(\mathbf{u}) = \left(\mathbf{x}_t^\top \mathbf{u} - y_t\right)^2$ the instantaneous loss of a vector $\mathbf{u}$, and by $L_T(\mathbf{u}) = \sum_{t=1}^{T} \ell_t(\mathbf{u})$ the total loss of the competitor $\mathbf{u}$. The regret with respect to $\mathbf{u}$ is defined to be,

$$R_T(\mathbf{u}) \doteq L_T(\text{alg}) - L_T(\mathbf{u}) = \sum_{t=1}^{T} (\hat{y}_t - y_t)^2 - \sum_{t=1}^{T} (\mathbf{x}_t^\top \mathbf{u} - y_t)^2 \; .$$

A desired goal of an algorithm is to have $R_T(\mathbf{u}) = o(T)$, that is, the average loss suffered by an algorithm will converge to the average loss of the best linear function $\mathbf{u}$.

In the **non-stationary** setting we focus on algorithms that are able to compete against sequences of weight-vectors, $(\mathbf{u}_1, \ldots, \mathbf{u}_T) \in \mathbb{R}^d \times \cdots \times \mathbb{R}^d$, where $\mathbf{u}_t$ is used to make a prediction for the tth example $(\mathbf{x}_t, y_t)$. We define the cumulative loss of such set by $L_T(\{\mathbf{u}_t\}) = \sum_{t=1}^{T} \ell_t(\mathbf{u}_t)$ and the regret of an algorithm by

$$R_T(\{\mathbf{u}_t\}) \doteq L_T(\text{alg}) - L_T(\{\mathbf{u}_t\}) = \sum_{t=1}^{T} (\hat{y}_t - y_t)^2 - \sum_{t=1}^{T} (\mathbf{x}_t^\top \mathbf{u}_t - y_t)^2 \ .$$

The goal of an algorithm is to have a low-regret, and formally to have $R_T(\{\mathbf{u}_t\}) = o(T)$, that is, the average loss suffered by an algorithm will converge to the average loss of the best linear function sequence $(\mathbf{u}_1, \ldots, \mathbf{u}_T)$.

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

$$V = V_T(\{\mathbf{u}_t\}) = \sum_{t=1}^{T-1} \|\mathbf{u}_t - \mathbf{u}_{t+1}\|^2 \ .$$

We cannot expect a learning algorithm to have a low-regret when the drift V is linear in T. On the other hand, when $V = o(T)$ (sublinear drift) we expect sublinear regret, $R_T(\{\mathbf{u}_t\}) = o(T)$. In addition, when there is no drift in the data ($V = 0$), an algorithm designed for non-stationary setting should have a regret as small as an algorithm designed for stationary setting.

Chapter 3

Stationary Regression

Forster [1999] proposed the last-step min-max prediction rule for online stationary regression. According to this rule, each round the algorithm assumes that it is the last round, and outputs the best prediction $\hat{y}_T$ assuming the worst choice of the label y_T. Thus, the optimal prediction is[1]

$$\hat{y}_T = \arg\min_{\hat{y}_T} \max_{y_T} \left[\sum_{t=1}^{T} (y_t - \hat{y}_t)^2 - \inf_{\mathbf{u}} \left(b \left\| \mathbf{u} \right\|^2 + L_T(\mathbf{u}) \right) \right] , \tag{3.1}$$

for some positive constant $b > 0$. The first term of Eq. (3.1) is the loss suffered by the algorithm while the second term is a sum of the loss suffered by a competitor $\mathbf{u}$ and a penalty for the norm of $\mathbf{u}$ to be far from zero. As we will see, to make the min-max optimization problem well defined, Forster [1999] assumed an artificial bound on the labels $|y_t| \leq Y$, and formally the bound should be known to the learning algorithm.

In Sec. 3.1 we introduce the notation of *weighted loss*, where we replace $L_T(\mathbf{u})$ in Eq. (3.1) with a weighted loss. This will help us to make the min-max optimization problem well defined (without the need to bound the labels), and our prediction algorithm will be the formal solution of this problem. In Sec. 3.2 we show that our algorithm can be expressed in a recursive form, similar to other algorithms. In Sec. 3.3 we show that the algorithm can be expressed in dual variables, which allows an efficient run of the algorithm in any reproducing kernel Hilbert space. Finally, in Sec. 3.4 we compare our algorithm to other algorithms of the same form.

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

The analysis of the algorithm appears in Chapter 4, where we derive two regret bounds for the algorithm, and compare them to other bounds.

3.1 Weighted Min-Max (WEMM) algorithm

The algorithm we describe below is based on an extension of the last-step min-max rule of Forster [1999]. For this purpose we introduce a weighted cumulative loss using positive input-dependent weights $\{a_t\}_{t=1}^T$,

$$L_T^{\boldsymbol{a}}(\mathbf{u}) = \sum_{t=1}^{T} a_t \left(y_t - \mathbf{u}^\top \mathbf{x}_t\right)^2 .$$

The exact values of the weights a_t will be defined below.

Our variant of the last-step min-max algorithm predicts

$$\hat{y}_T = \arg\min_{\hat{y}_T} \max_{y_T} \left[\sum_{t=1}^{T} (y_t - \hat{y}_t)^2 - \inf_{\mathbf{u}} \left(b \left\|\mathbf{u}\right\|^2 + L_T^{\boldsymbol{a}}(\mathbf{u}) \right) \right] , \tag{3.2}$$

for some positive constant $b > 0$. We next compute the actual prediction based on the optimal last-step min-max solution of Eq. (3.2). We start with additional notation,

$$\mathbf{A}_t = b\mathbf{I} + \sum_{s=1}^{t} a_s \mathbf{x}_s \mathbf{x}_s^\top \qquad \in \mathbb{R}^{d\times d} \tag{3.3}$$

$$\mathbf{b}_t = \sum_{s=1}^{t} a_s y_s \mathbf{x}_s \qquad \in \mathbb{R}^{d} . \tag{3.4}$$

The solution of the internal infimum over $\mathbf{u}$ is summarized in the following lemma.

Lemma 1 *For all $t \geq 1$, the function $f\left(\mathbf{u}\right) = b\left\|\mathbf{u}\right\|^2 + \sum_{s=1}^{t} a_s \left(y_s - \mathbf{u}^\top \mathbf{x}_s\right)^2$ is minimal at a unique point $\mathbf{u}_t$ given by,*

$$\mathbf{u}_t = \mathbf{A}_t^{-1}\mathbf{b}_t \quad \textit{and} \quad f(\mathbf{u}_t) = \sum_{s=1}^{t} a_s y_s^2 - \mathbf{b}_t^\top \mathbf{A}_t^{-1}\mathbf{b}_t . \tag{3.5}$$

Proof: From

$$\begin{aligned}
f(\mathbf{u}) &= b\|\mathbf{u}\|^2 + \sum_{s=1}^{t} a_s \left(y_s - \mathbf{u}^\top \mathbf{x}_s\right)^2 \\
&= \sum_{s=1}^{t} a_s y_s^2 - 2\sum_{s=1}^{t} \mathbf{u}^\top \left(a_s y_s \mathbf{x}_s\right) + \mathbf{u}^\top \left(b\mathbf{I} + \sum_{s=1}^{t} a_s \mathbf{x}_s \mathbf{x}_s^\top\right)\mathbf{u} \\
&\overset{Eq.~(3.3),Eq.~(3.4)}{=} \sum_{s=1}^{t} a_s y_s^2 - 2\mathbf{u}^\top \mathbf{b}_t + \mathbf{u}^\top \mathbf{A}_t \mathbf{u}
\end{aligned}$$

it follows that $\nabla f(\mathbf{u}) = 2\mathbf{A}_t\mathbf{u} - 2\mathbf{b}_t,\ \triangle f(\mathbf{u}) = 2\mathbf{A}_t \succ 0$. Thus f is convex and it is minimal if $\nabla f(\mathbf{u}) = 0$, i.e. for $\mathbf{u} = \mathbf{A}_t^{-1}\mathbf{b}_t$. This shows that $\mathbf{u}_t = \mathbf{A}_t^{-1}\mathbf{b}_t$ and we obtain

$$f(\mathbf{u}_t) = f\left(\mathbf{A}_t^{-1}\mathbf{b}_t\right) = \sum_{s=1}^{t} a_s y_s^2 - 2\mathbf{b}_t^\top \mathbf{A}_t^{-1}\mathbf{b}_t + \mathbf{b}_t^\top \mathbf{A}_t^{-1}\mathbf{A}_t\mathbf{A}_t^{-1}\mathbf{b}_t = \sum_{s=1}^{t} a_s y_s^2 - \mathbf{b}_t^\top \mathbf{A}_t^{-1}\mathbf{b}_t \,.$$

■

Remark 1 *The minimization problem in Lemma 1 can be interpreted as MAP estimator of* $\mathbf{u}$ *based on the sequence* $\{(\mathbf{x}_s, y_s)\}_{s=1}^{t}$ *in the following generative model:*

$$\begin{aligned}
\mathbf{u} &\sim N\left(0, \sigma_b^2 \mathbf{I}\right) \\
y_s &\sim N\left(\mathbf{x}_s^\top \mathbf{u}, \sigma_s^2\right) ,
\end{aligned}$$

where $\sigma_b^2 = \frac{1}{2b}$ *and* $\sigma_s^2 = \frac{1}{2a_s}$.

Under the model we calculate,

$$\begin{aligned}
\mathbf{u}_{MAP} &= \arg\max_{\mathbf{u}} P\left(\mathbf{u} \mid \{\mathbf{x}_s\}, \{y_s\}\right) \\
&= \arg\max_{\mathbf{u}} \left[P(\mathbf{u}) \prod_{s=1}^{t} P\left(y_s \mid \mathbf{u}, \mathbf{x}_s\right)\right] \\
&= \arg\min_{\mathbf{u}} \left[-\log P(\mathbf{u}) - \sum_{s=1}^{t} \log P\left(y_s \mid \mathbf{u}, \mathbf{x}_s\right)\right] . \qquad (3.6)
\end{aligned}$$

By our gaussian generative model,

$$-\log P\left(\mathbf{u}\right) = \log\left(2\pi\sigma_b^2\right)^{d/2} + \frac{1}{2\sigma_b^2}\left\|\mathbf{u}\right\|^2$$

$$-\log P\left(y_s \mid \mathbf{u}, \mathbf{x}_s\right) = \log\left(2\pi\sigma_s^2\right)^{1/2} + \frac{1}{2\sigma_s^2}\left(y_s - \mathbf{x}_s^\top \mathbf{u}\right)^2 \ .$$

Substituting in Eq. (3.6) we get

$$\mathbf{u}_{MAP} = \arg\min_{\mathbf{u}}\left[\frac{1}{2\sigma_b^2}\left\|\mathbf{u}\right\|^2 + \sum_{s=1}^{t}\frac{1}{2\sigma_s^2}\left(y_s - \mathbf{x}_s^\top \mathbf{u}\right)^2\right] \ ,$$

and by using $\frac{1}{2\sigma_b^2} = b$*,* $\frac{1}{2\sigma_s^2} = a_s$ *we get the minimization problem of Lemma 1.*

Substituting Eq. (3.5) back in Eq. (3.2) we obtain the following form of the min-max problem,

$$\min_{\hat{y}_T}\max_{y_T} G(y_T, \hat{y}_T) \quad \text{for} \quad G(y_T, \hat{y}_T) = \alpha(a_T)y_T^2 + 2\beta(a_T, \hat{y}_T)y_T + \hat{y}_T^2 \ , \tag{3.7}$$

for some functions $\alpha(a_T)$ and $\beta(a_T, \hat{y}_T)$. Clearly, for this problem to be well defined the function G should be convex in $\hat{y}_T$ and concave in y_T.

A previous choice, proposed by Forster [1999], is to have uniform weights and set $a_t = 1$ (for $t = 1, \ldots, T$), which for the particular function $\alpha(a_T)$ yields $\alpha(a_T) > 0$. Thus, $G(y_T, \hat{y}_T)$ is a convex function in y_T, implying that the optimal value of G is not bounded from above. Forster [1999] addressed this problem by restricting y_T to belong to a predefined interval $[-Y, Y]$, known also to the learner. As a consequence, the adversary optimal prediction is in fact either $y_T = Y$ or $y_T = -Y$, which in turn yields an optimal predictor which is clipped at this bound, $\hat{y}_T = \text{clip}\left(\mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1}\mathbf{x}_T, Y\right)$, where for $y > 0$ we define $\text{clip}(x, y) = x$ if $|x| \leq y$ and $\text{clip}(x, y) = y\,\text{sign}(x)$, otherwise.

This phenomena is illustrated in the left panel of Fig. 3.1 (best viewed in color). For the min-max optimization function defined by Forster [1999], fixing some value of $\hat{y}_T$, the function is convex in y_T, and the adversary would achieve a maximal value at the boundary of the feasible values of y_T interval. That is, either $y_T = Y$ or $y_T = -Y$, as indicated by the two magenta lines at $y_T = \pm 10$. The optimal predictor $\hat{y}_T$ is achieved somewhere along the lines $y_T = Y$ or $y_T = -Y$.

We propose an alternative approach to make the min-max optimal solution

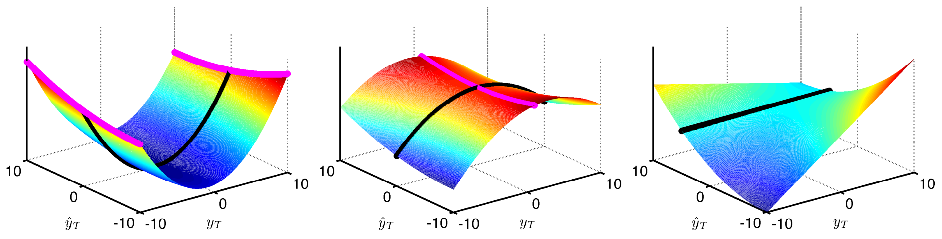

Figure 3.1: An illustration of the min-max objective function $G(y_T, \hat{y}_T)$ Eq. (3.7). The black line is the value of the objective as a function of y_T for the optimal predictor $\hat{y}_T$. Left: Forster's optimization function (convex in y_T). Center: our optimization function (strictly concave in y_T, case 1 in Theorem 2). Right: our optimization function (invariant to y_T, case 2 in Theorem 2).

bounded by appropriately setting the weight a_T such that $G(y_T, \hat{y}_T)$ is concave in y_T for a constant $\hat{y}_T$. We explicitly consider two cases. First, set a_T such that $G(y_T, \hat{y}_T)$ is *strictly concave* in y_T, and thus attains a single maximum with no need to artificially restrict the value of y_T. The optimal predictor $\hat{y}_T$ in this case is achieved in the unique saddle point, as illustrated in the center panel of Fig. 3.1. A second case is to set a_T such that $\alpha(a_T) = 0$ and the min-max function $G(y_T, \hat{y}_T)$ becomes linear in y_T. Here, the optimal prediction is achieved by choosing $\hat{y}_T$ such that $\beta(a_T, \hat{y}_T) = 0$ which turns $G(y_T, \hat{y}_T)$ to be invariant to y_T, as illustrated in the right panel of Fig. 3.1.

Equipped with Lemma 1 we develop the optimal solution of the min-max predictor, summarized in the following theorem.

Theorem 2 *Assume that* $1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T - a_T \leq 0$. *Then the optimal prediction for the last round* T *is*

$$\hat{y}_T = \mathbf{b}_{T-1}^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T \ . \tag{3.8}$$

The proof of the theorem makes use of the following technical lemma.

Lemma 3 *For all* $t = 1, 2, \ldots, T$

$$a_t^2 \mathbf{x}_t^\top \mathbf{A}_t^{-1} \mathbf{x}_t + 1 - a_t = \frac{1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t - a_t}{1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t} \ . \tag{3.9}$$

Proof: Using the Woodbury identity we get

$$\mathbf{A}_t^{-1} = \mathbf{A}_{t-1}^{-1} - \frac{\mathbf{A}_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}}{\frac{1}{a_t} + \mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t} ,$$

therefore the left side of Eq. (3.9) is

$$\begin{aligned} a_t^2\mathbf{x}_t^\top\mathbf{A}_t^{-1}\mathbf{x}_t + 1 - a_t &= a_t^2\mathbf{x}_t^\top\left(\mathbf{A}_{t-1}^{-1} - \frac{\mathbf{A}_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}}{\frac{1}{a_t} + \mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t}\right)\mathbf{x}_t + 1 - a_t \\ &= a_t^2\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t - \frac{a_t^2\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t}{\frac{1}{a_t} + \mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t} + 1 - a_t \\ &= \frac{1 + a_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t - a_t}{1 + a_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t} . \end{aligned}$$

■

We now prove Theorem 2.

Proof: The adversary can choose any y_T, thus the algorithm should predict $\hat{y}_T$ such that the following quantity is minimal,

$$\begin{aligned} &\max_{y_T}\left(\sum_{t=1}^T (y_t - \hat{y}_t)^2 - \inf_{\mathbf{u}\in\mathbb{R}^d}\left(b\left\|\mathbf{u}\right\|^2 + \sum_{t=1}^T a_t\left(y_t - \mathbf{u}^\top\mathbf{x}_t\right)^2\right)\right) \\ \overset{Eq.\ (3.5)}{=}\ &\max_{y_T}\left(\sum_{t=1}^T (y_t - \hat{y}_t)^2 - \sum_{t=1}^T a_t y_t^2 + \mathbf{b}_T^\top\mathbf{A}_T^{-1}\mathbf{b}_T\right) . \end{aligned}$$

That is, we need to solve the following min-max problem

$$\min_{\hat{y}_T}\max_{y_T}\left(\sum_{t=1}^T (y_t - \hat{y}_t)^2 - \sum_{t=1}^T a_t y_t^2 + \mathbf{b}_T^\top\mathbf{A}_T^{-1}\mathbf{b}_T\right) .$$

We use the following relation to re-write the optimization problem,

$$\mathbf{b}_T^\top\mathbf{A}_T^{-1}\mathbf{b}_T = \mathbf{b}_{T-1}^\top\mathbf{A}_T^{-1}\mathbf{b}_{T-1} + 2a_T y_T\mathbf{b}_{T-1}^\top\mathbf{A}_T^{-1}\mathbf{x}_T + a_T^2 y_T^2\mathbf{x}_T^\top\mathbf{A}_T^{-1}\mathbf{x}_T . \tag{3.10}$$

Omitting all terms that are not depending on y_T and $\hat{y}_T$,

$$\min_{\hat{y}_T}\max_{y_T}\left((y_T - \hat{y}_T)^2 - a_T y_T^2 + 2a_T y_T\mathbf{b}_{T-1}^\top\mathbf{A}_T^{-1}\mathbf{x}_T + a_T^2 y_T^2\mathbf{x}_T^\top\mathbf{A}_T^{-1}\mathbf{x}_T\right) .$$

We manipulate the last problem to be of form Eq. (3.7) using Lemma 3,

$$\min_{\hat{y}_T} \max_{y_T} \left(\frac{1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T - a_T}{1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T} y_T^2 + 2 y_T \left(a_T \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} \mathbf{x}_T - \hat{y}_T \right) + \hat{y}_T^2 \right) , \tag{3.11}$$

where

$$\alpha(a_T) = \frac{1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T - a_T}{1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T} \quad \text{and} \quad \beta(a_T, \hat{y}_T) = a_T \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} \mathbf{x}_T - \hat{y}_T \ .$$

We consider two cases: (1) $1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T - a_T < 0$ (corresponding to the middle panel of Fig. 3.1), and (2) $1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T - a_T = 0$ (corresponding to the right panel of Fig. 3.1), starting with the first case,

$$1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T - a_T < 0 \ . \tag{3.12}$$

Denote the inner-maximization problem by,

$$f\left(y_T\right) = \frac{1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T - a_T}{1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T} y_T^2 + 2 y_T \left(a_T \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} \mathbf{x}_T - \hat{y}_T \right) + \hat{y}_T^2 \ .$$

This function is strictly-concave with respect to y_T because of Eq. (3.12). Thus, it has a unique maximal value given by,

$$\begin{aligned} f^{max}(\hat{y}_T) \quad = \quad & -\frac{a_T}{1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T - a_T} \hat{y}_T^2 + \frac{2 a_T \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} \mathbf{x}_T \left(1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T\right)}{1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T - a_T} \hat{y}_T \\ & - \frac{\left(a_T \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} \mathbf{x}_T\right)^2 \left(1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T\right)}{1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T - a_T} \ . \end{aligned}$$

Next, we solve $\min_{\hat{y}_T} f^{max}(\hat{y}_T)$, which is strictly-convex with respect to $\hat{y}_T$ because of Eq. (3.12). Solving this problem we get the optimal last-step min-max predictor,

$$\hat{y}_T = \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} \mathbf{x}_T \left(1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T\right) \ . \tag{3.13}$$

We further derive the last equation. From Eq. (3.3) we have,

$$\mathbf{A}_T^{-1} a_T \mathbf{x}_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} = \mathbf{A}_T^{-1} \left(\mathbf{A}_T - \mathbf{A}_{T-1}\right) \mathbf{A}_{T-1}^{-1} = \mathbf{A}_{T-1}^{-1} - \mathbf{A}_T^{-1} \ . \tag{3.14}$$

Substituting Eq. (3.14) in Eq. (3.13) we have the following equality as desired,

$$\hat{y}_T = \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1}\mathbf{x}_T + \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} a_T \mathbf{x}_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1}\mathbf{x}_T = \mathbf{b}_{T-1}^\top \mathbf{A}_{T-1}^{-1}\mathbf{x}_T \ . \quad (3.15)$$

We now move to the second case for which, $1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1}\mathbf{x}_T - a_T = 0$, which is written equivalently as,

$$a_T = \frac{1}{1 - \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1}\mathbf{x}_T} \ . \quad (3.16)$$

Substituting Eq. (3.16) in Eq. (3.11) we get,

$$\min_{\hat{y}_T} \max_{y_T} \left(2y_T \left(a_T \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1}\mathbf{x}_T - \hat{y}_T\right) + \hat{y}_T^2\right) \ .$$

For $\hat{y}_T \neq a_T \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1}\mathbf{x}_T$, the value of the optimization problem is not-bounded as the adversary may choose $y_T = z^2 \left(a_T \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1}\mathbf{x}_T - \hat{y}_T\right)$ for $z \to \infty$. Thus, the optimal last-step min-max prediction is to set $\hat{y}_T = a_T \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1}\mathbf{x}_T$. Substituting $a_T = 1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1}\mathbf{x}_T$ and following the derivation from Eq. (3.13) to Eq. (3.15) above, yields the desired identity. ■

We conclude by noting that although we did not restrict the form of the predictor $\hat{y}_T$, it turns out that it is a linear predictor defined by $\hat{y}_T = \mathbf{x}_T^\top \mathbf{w}_{T-1}$ for $\mathbf{w}_{T-1} = \mathbf{A}_{T-1}^{-1}\mathbf{b}_{T-1}$. In other words, the functional form of the optimal predictor is the same as the form of the comparison function class - linear functions in our case. We call the algorithm (defined using Eq. (3.3), Eq. (3.4) and Eq. (3.8)) `WEMM` for weighted min-max prediction, and it is summarised in Fig. 3.2. We note that `WEMM` can also be seen as an incremental off-line algorithm [Azoury and Warmuth, 2001] or follow-the-leader, on a weighted sequence: The prediction $\hat{y}_T = \mathbf{x}_T^\top \mathbf{w}_{T-1}$ is with a model that is optimal over a prefix of length $T-1$. The prediction of the optimal predictor defined in Eq. (3.5) is $\mathbf{x}_T^\top \mathbf{u}_{T-1} = \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1}\mathbf{b}_{T-1} = \hat{y}_T$, where $\hat{y}_T$ was defined in Eq. (3.8).

```
Parameter:  $1 < b$
Initialize:  Set $\mathbf{A}_0 = b\,\mathbf{I} \in \mathbb{R}^{d\times d}$ and $\mathbf{b}_0 = \mathbf{0} \in \mathbb{R}^d$
For $t = 1, \ldots, T$ do
    • Receive an instance $\mathbf{x}_t$
    • Output prediction $\hat{y}_t = \mathbf{b}_{t-1}^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t$
    • Receive the correct label $y_t$
    • Set weight:
          $$a_t = \frac{1}{1 - \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t}$$
    • Update:
          $$\mathbf{A}_t = \mathbf{A}_{t-1} + a_t \mathbf{x}_t \mathbf{x}_t^\top$$
          $$\mathbf{b}_t = \mathbf{b}_{t-1} + a_t y_t \mathbf{x}_t$$
Output:  $\mathbf{A}_T$ , $\mathbf{b}_T$
```

Figure 3.2: WEMM: weighted min-max prediction.

3.2 Recursive form

Although Theorem 2 is correct for $1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T - a_T \leq 0$, for the rest of the work we will assume an equality, that is

$$a_t = \frac{1}{1 - \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t} \quad , \quad t = 1 \ldots T\ . \tag{3.17}$$

For this case, the `WEMM` algorithm can be expressed in a recursive form in terms of weight vector $\mathbf{w}_t$ and a covariance-like matrix $\mathbf{\Sigma}_t$. We denote $\mathbf{w}_t = \mathbf{A}_t^{-1}\mathbf{b}_t$ and $\mathbf{\Sigma}_t = \mathbf{A}_t^{-1}$, and develop recursive update rules for $\mathbf{w}_t$ and $\mathbf{\Sigma}_t$:

$$\begin{aligned}
\mathbf{w}_t &= \mathbf{A}_t^{-1}\mathbf{b}_t \\
&= \left(\mathbf{A}_{t-1} + a_t \mathbf{x}_t \mathbf{x}_t^\top\right)^{-1} (\mathbf{b}_{t-1} + a_t y_t \mathbf{x}_t) \\
&= \left(\mathbf{A}_{t-1}^{-1} - \frac{\mathbf{A}_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}}{a_t^{-1} + \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1}\mathbf{x}_t}\right) (\mathbf{b}_{t-1} + a_t y_t \mathbf{x}_t)
\end{aligned}$$

$$
\begin{aligned}
&= \quad \mathbf{w}_{t-1} - \frac{\mathbf{A}_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^\top\mathbf{w}_{t-1}}{a_t^{-1} + \mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t} + a_t y_t \mathbf{A}_{t-1}^{-1}\mathbf{x}_t - \frac{a_t y_t \mathbf{A}_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t}{a_t^{-1} + \mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t} \\
&= \quad \mathbf{w}_{t-1} + \frac{y_t\mathbf{A}_{t-1}^{-1}\mathbf{x}_t - \mathbf{A}_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^\top\mathbf{w}_{t-1}}{a_t^{-1} + \mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t} \\
&\overset{Eq.\ (3.17)}{=} \quad \mathbf{w}_{t-1} + \left(y_t - \mathbf{x}_t^\top\mathbf{w}_{t-1}\right)\mathbf{A}_{t-1}^{-1}\mathbf{x}_t \\
&= \quad \mathbf{w}_{t-1} + \left(y_t - \mathbf{x}_t^\top\mathbf{w}_{t-1}\right)\boldsymbol{\Sigma}_{t-1}\mathbf{x}_t \ , \qquad (3.18)
\end{aligned}
$$

and

$$
\begin{aligned}
\boldsymbol{\Sigma}_t^{-1} &= \quad \mathbf{A}_t = \mathbf{A}_{t-1} + a_t\mathbf{x}_t\mathbf{x}_t^\top \\
&\overset{Eq.\ (3.17)}{=} \quad \mathbf{A}_{t-1} + \frac{\mathbf{x}_t\mathbf{x}_t^\top}{1 - \mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t} \\
&= \quad \boldsymbol{\Sigma}_{t-1}^{-1} + \frac{\mathbf{x}_t\mathbf{x}_t^\top}{1 - \mathbf{x}_t^\top\boldsymbol{\Sigma}_{t-1}\mathbf{x}_t}
\end{aligned}
$$

or

$$
\boldsymbol{\Sigma}_t = \boldsymbol{\Sigma}_{t-1} - \boldsymbol{\Sigma}_{t-1}\mathbf{x}_t\mathbf{x}_t^\top\boldsymbol{\Sigma}_{t-1} \ . \qquad (3.19)
$$

A summary of the algorithm in a recursive form appears in the right column of Table 3.1.

3.3 Kernel version of the algorithm

In this section we show that the WEMM algorithm can be expressed in dual variables, which allows an efficient run of the algorithm in any reproducing kernel Hilbert space. We show by induction that the weight-vector $\mathbf{w}_t$ and the covariance matrix $\boldsymbol{\Sigma}_t$ computed by the WEMM algorithm in the right column of Table 3.1 can be written in the form

$$
\begin{aligned}
\mathbf{w}_t &= \sum_{i=1}^{t} \alpha_i^{(t)}\mathbf{x}_i \\
\boldsymbol{\Sigma}_t &= \sum_{j=1}^{t}\sum_{k=1}^{t} \beta_{j,k}^{(t)}\mathbf{x}_j\mathbf{x}_k^\top + b^{-1}\mathbf{I} \ ,
\end{aligned}
$$

where the coefficients α_i and $\beta_{j,k}$ depend only on inner products of the input vectors.

For the initial step we have $\mathbf{w}_0 = \mathbf{0}$ and $\mathbf{\Sigma}_0 = b^{-1}\mathbf{I}$ which are trivially written in the desired form by setting $\alpha^{(0)} = 0$ and $\beta^{(0)} = 0$. We proceed to the induction step. From the weight-vector update rule Eq. (3.18) we get

$$\begin{aligned}
\mathbf{w}_t &= \mathbf{w}_{t-1} + \left(y_t - \mathbf{x}_t^\top \mathbf{w}_{t-1}\right) \mathbf{\Sigma}_{t-1}\mathbf{x}_t = \\
&= \sum_{i=1}^{t-1} \alpha_i^{(t-1)}\mathbf{x}_i + \left(y_t - \mathbf{x}_t^\top \sum_{i=1}^{t-1} \alpha_i^{(t-1)}\mathbf{x}_i\right) \left(\sum_{j=1}^{t-1}\sum_{k=1}^{t-1} \beta_{j,k}^{(t-1)}\mathbf{x}_j\mathbf{x}_k^\top + b^{-1}\mathbf{I}\right)\mathbf{x}_t \\
&= \sum_{i=1}^{t-1} \alpha_i^{(t-1)}\mathbf{x}_i + \left(y_t - \sum_{i=1}^{t-1} \alpha_i^{(t-1)}\left(\mathbf{x}_t^\top\mathbf{x}_i\right)\right) \left(\sum_{j=1}^{t-1}\sum_{k=1}^{t-1} \beta_{j,k}^{(t-1)}\left(\mathbf{x}_k^\top\mathbf{x}_t\right)\mathbf{x}_j + b^{-1}\mathbf{x}_t\right) \\
&= \sum_{i=1}^{t-1} \left[\alpha_i^{(t-1)} + \left(y_t - \sum_{l=1}^{t-1} \alpha_l^{(t-1)}\left(\mathbf{x}_t^\top\mathbf{x}_l\right)\right) \sum_{k=1}^{t-1} \beta_{i,k}^{(t-1)}\left(\mathbf{x}_k^\top\mathbf{x}_t\right)\right]\mathbf{x}_i \\
&\quad + b^{-1}\left(y_t - \sum_{i=1}^{t-1} \alpha_i^{(t-1)}\left(\mathbf{x}_t^\top\mathbf{x}_i\right)\right)\mathbf{x}_t \ ,
\end{aligned}$$

thus

$$\alpha_i^{(t)} = \begin{cases} \alpha_i^{(t-1)} + \left(y_t - \sum_{l=1}^{t-1}\alpha_l^{(t-1)}\left(\mathbf{x}_t^\top\mathbf{x}_l\right)\right)\sum_{k=1}^{t-1}\beta_{i,k}^{(t-1)}\left(\mathbf{x}_k^\top\mathbf{x}_t\right) & i = 1,\ldots,t-1 \\ b^{-1}\left(y_t - \sum_{l=1}^{t-1}\alpha_l^{(t-1)}\left(\mathbf{x}_t^\top\mathbf{x}_l\right)\right) & i = t \end{cases}$$

From the covariance matrix update rule Eq. (3.19) we get

$$\begin{aligned}
\mathbf{\Sigma}_t &= \mathbf{\Sigma}_{t-1} - \mathbf{\Sigma}_{t-1}\mathbf{x}_t\mathbf{x}_t^\top\mathbf{\Sigma}_{t-1} \\
&= \sum_{j=1}^{t-1}\sum_{k=1}^{t-1} \beta_{j,k}^{(t-1)}\mathbf{x}_j\mathbf{x}_k^\top + b^{-1}\mathbf{I} \\
&\quad - \left(\sum_{j=1}^{t-1}\sum_{k=1}^{t-1} \beta_{j,k}^{(t-1)}\mathbf{x}_j\mathbf{x}_k^\top + b^{-1}\mathbf{I}\right)\mathbf{x}_t\mathbf{x}_t^\top\left(\sum_{j=1}^{t-1}\sum_{k=1}^{t-1} \beta_{j,k}^{(t-1)}\mathbf{x}_j\mathbf{x}_k^\top + b^{-1}\mathbf{I}\right) \\
&= \sum_{j=1}^{t-1}\sum_{k=1}^{t-1} \beta_{j,k}^{(t-1)}\mathbf{x}_j\mathbf{x}_k^\top + b^{-1}\mathbf{I}
\end{aligned}$$

$$\begin{aligned}
& -\left(\sum_{j=1}^{t-1}\sum_{k=1}^{t-1}\beta_{j,k}^{(t-1)}\left(\mathbf{x}_k^\top\mathbf{x}_t\right)\mathbf{x}_j + b^{-1}\mathbf{x}_t\right)\left(\sum_{j=1}^{t-1}\sum_{k=1}^{t-1}\beta_{j,k}^{(t-1)}\left(\mathbf{x}_t^\top\mathbf{x}_j\right)\mathbf{x}_k^\top + b^{-1}\mathbf{x}_t^\top\right) \\
= \ & \sum_{j=1}^{t-1}\sum_{k=1}^{t-1}\beta_{j,k}^{(t-1)}\mathbf{x}_j\mathbf{x}_k^\top + b^{-1}\mathbf{I} - \sum_{j=1}^{t-1}\sum_{k=1}^{t-1}\sum_{l=1}^{t-1}\sum_{m=1}^{t-1}\beta_{l,m}^{(t-1)}\beta_{j,k}^{(t-1)}\left(\mathbf{x}_k^\top\mathbf{x}_t\right)\left(\mathbf{x}_t^\top\mathbf{x}_l\right)\mathbf{x}_j\mathbf{x}_m^\top \\
& -b^{-1}\sum_{j=1}^{t-1}\sum_{k=1}^{t-1}\beta_{j,k}^{(t-1)}\left(\mathbf{x}_t^\top\mathbf{x}_j\right)\mathbf{x}_t\mathbf{x}_k^\top - b^{-1}\sum_{j=1}^{t-1}\sum_{k=1}^{t-1}\beta_{j,k}^{(t-1)}\left(\mathbf{x}_k^\top\mathbf{x}_t\right)\mathbf{x}_j\mathbf{x}_t^\top - b^{-2}\mathbf{x}_t\mathbf{x}_t^\top \\
= \ & \sum_{j=1}^{t-1}\sum_{k=1}^{t-1}\left[\beta_{j,k}^{(t-1)} - \sum_{l=1}^{t-1}\sum_{m=1}^{t-1}\beta_{l,k}^{(t-1)}\beta_{j,m}^{(t-1)}\left(\mathbf{x}_m^\top\mathbf{x}_t\right)\left(\mathbf{x}_t^\top\mathbf{x}_l\right)\right]\mathbf{x}_j\mathbf{x}_k^\top + b^{-1}\mathbf{I} \\
& -b^{-1}\sum_{k=1}^{t-1}\sum_{j=1}^{t-1}\beta_{j,k}^{(t-1)}\left(\mathbf{x}_t^\top\mathbf{x}_j\right)\mathbf{x}_t\mathbf{x}_k^\top - b^{-1}\sum_{j=1}^{t-1}\sum_{k=1}^{t-1}\beta_{j,k}^{(t-1)}\left(\mathbf{x}_k^\top\mathbf{x}_t\right)\mathbf{x}_j\mathbf{x}_t^\top - b^{-2}\mathbf{x}_t\mathbf{x}_t^\top \ ,
\end{aligned}$$

thus

$$\beta_{j,k}^{(t)} = \begin{cases} \beta_{j,k}^{(t-1)} - \sum_{l=1}^{t-1}\sum_{m=1}^{t-1}\beta_{l,k}^{(t-1)}\beta_{j,m}^{(t-1)}\left(\mathbf{x}_m^\top\mathbf{x}_t\right)\left(\mathbf{x}_t^\top\mathbf{x}_l\right) & j,k = 1,\ldots,t-1 \\ -b^{-1}\sum_{l=1}^{t-1}\beta_{l,k}^{(t-1)}\left(\mathbf{x}_t^\top\mathbf{x}_l\right) & j = t\ ,\ k = 1,\ldots,t-1 \\ -b^{-1}\sum_{l=1}^{t-1}\beta_{j,l}^{(t-1)}\left(\mathbf{x}_l^\top\mathbf{x}_t\right) & k = t\ ,\ j = 1,\ldots,t-1 \\ -b^{-2} & j = k = t \end{cases}$$

A summary of the kernel version of the `WEMM` algorithm appears in Fig. 3.3.

3.4 Comparison to other algorithms

In this section we compare similar second order online algorithms for stationary regression. The ridge-regression [Foster, 1991], summarized in the third column of Table 3.1, uses the previous examples to generate a weight-vector, which is used to predict current example. On round t it sets a weight-vector to be the solution of the following optimization problem,

$$\mathbf{w}_{t-1} = \arg\min_{\mathbf{w}}\left[\sum_{i=1}^{t-1}\left(y_i - \mathbf{x}_i^\top\mathbf{w}\right)^2 + b\left\|\mathbf{w}\right\|^2\right] \ ,$$

and outputs a prediction $\hat{y}_t = \mathbf{x}_t^\top\mathbf{w}_{t-1}$. The recursive least squares (RLS) [Hayes, 1996] is a similar algorithm, yet it uses a forgetting factor $0 < r \leq 1$, and sets the

```
**Parameter:** $1 < b$, kernel function $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$
**Initialize:** Set $\alpha^{(0)} = 0$ and $\beta^{(0)} = 0$
**For** $t = 1, \ldots, T$ do
    • Receive an instance $\mathbf{x}_t$
    • Output prediction $\hat{y}_t = \sum_{i=1}^{t-1} \alpha_i^{(t-1)} K(\mathbf{x}_t, \mathbf{x}_i)$
    • Receive the correct label $y_t$
    • Update:
```

$$\alpha_i^{(t)} = \begin{cases} \alpha_i^{(t-1)} + \left(y_t - \sum_{l=1}^{t-1} \alpha_l^{(t-1)} K(\mathbf{x}_t, \mathbf{x}_l)\right) \sum_{k=1}^{t-1} \beta_{i,k}^{(t-1)} K(\mathbf{x}_k, \mathbf{x}_t) & i = 1, \ldots, t-1 \\ b^{-1} \left(y_t - \sum_{l=1}^{t-1} \alpha_l^{(t-1)} K(\mathbf{x}_t, \mathbf{x}_l)\right) & i = t \end{cases}$$

$$\beta_{j,k}^{(t)} = \begin{cases} \beta_{j,k}^{(t-1)} - \sum_{l=1}^{t-1} \sum_{m=1}^{t-1} \beta_{l,k}^{(t-1)} \beta_{j,m}^{(t-1)} K(\mathbf{x}_m, \mathbf{x}_t) K(\mathbf{x}_t, \mathbf{x}_l) & j, k = 1, \ldots, t-1 \\ -b^{-1} \sum_{l=1}^{t-1} \beta_{l,k}^{(t-1)} K(\mathbf{x}_t, \mathbf{x}_l) & j = t \ , \ k = 1, \ldots, t-1 \\ -b^{-1} \sum_{l=1}^{t-1} \beta_{j,l}^{(t-1)} K(\mathbf{x}_l, \mathbf{x}_t) & k = t \ , \ j = 1, \ldots, t-1 \\ -b^{-2} & j = k = t \end{cases}$$

Output: $\left\{\alpha_i^{(T)}\right\}_{i=1}^{T}, \left\{\beta_{j,k}^{(T)}\right\}_{j,k=1}^{T}$

Figure 3.3: Kernel WEMM.

weight-vector according to

$$\mathbf{w}_{t-1} = \arg\min_{\mathbf{w}} \left[\sum_{i=1}^{t-1} r^{t-i-1} \left(y_i - \mathbf{x}_i^\top \mathbf{w}\right)^2\right] .$$

The Aggregating Algorithm for Regression (AAR) [Vovk, 1997, 2001], summarized in the second column of Table 3.1, was introduced by Vovk and it is similar to ridge-regression, except it contains additional regularization, which eventually makes it shrink the predictions. It is an application of the Aggregating Algorithm [Vovk, 1990] (a general algorithm for merging prediction strategies) to the problem of linear regression with square loss. On round t, the weight-vector is obtained according to

$$\mathbf{w}_t = \arg\min_{\mathbf{w}} \left[\sum_{i=1}^{t-1} \left(y_i - \mathbf{x}_i^\top \mathbf{w}\right)^2 + \left(\mathbf{x}_t^\top \mathbf{w}\right)^2 + b \left\|\mathbf{w}\right\|^2\right] ,$$

		AROWR	**AAR / Min-Max**	**Ridge-Regression**	**WEMM** (this work)
Parameters		$0 < r, b$	$0 < b$	$0 < b$	$1 < b$
Initialize		$\mathbf{w}_0 = \mathbf{0}\,, \boldsymbol{\Sigma}_0 = b^{-1}\mathbf{I}$			
For $t = 1 \ldots T$		Receive an instance $\mathbf{x}_t$			
	Output prediction:	$\hat{y}_t = \mathbf{x}_t^\top \mathbf{w}_{t-1}$	$\hat{y}_t = \frac{\mathbf{x}_t^\top \mathbf{w}_{t-1}}{1 + \mathbf{x}_t^\top \boldsymbol{\Sigma}_{t-1} \mathbf{x}_t}$	$\hat{y}_t = \mathbf{x}_t^\top \mathbf{w}_{t-1}$	$\hat{y}_t = \mathbf{x}_t^\top \mathbf{w}_{t-1}$
		Receive a correct label y_t			
	Update $\boldsymbol{\Sigma}_t$:	$\boldsymbol{\Sigma}_t^{-1} = \boldsymbol{\Sigma}_{t-1}^{-1} + \frac{1}{r}\mathbf{x}_t\mathbf{x}_t^\top$	$\boldsymbol{\Sigma}_t^{-1} = \boldsymbol{\Sigma}_{t-1}^{-1} + \mathbf{x}_t\mathbf{x}_t^\top$	$\boldsymbol{\Sigma}_t^{-1} = \boldsymbol{\Sigma}_{t-1}^{-1} + \mathbf{x}_t\mathbf{x}_t^\top$	$\boldsymbol{\Sigma}_t^{-1} = \boldsymbol{\Sigma}_{t-1}^{-1} + \frac{\mathbf{x}_t\mathbf{x}_t^\top}{1 - \mathbf{x}_t^\top \boldsymbol{\Sigma}_{t-1} \mathbf{x}_t}$
	Update $\mathbf{w}_t$:	$\mathbf{w}_t = \mathbf{w}_{t-1} + \frac{\left(y_t - \mathbf{x}_t^\top \mathbf{w}_{t-1}\right)\boldsymbol{\Sigma}_{t-1}\mathbf{x}_t}{r + \mathbf{x}_t^\top \boldsymbol{\Sigma}_{t-1}\mathbf{x}_t}$	$\mathbf{w}_t = \mathbf{w}_{t-1} + \frac{\left(y_t - \mathbf{x}_t^\top \mathbf{w}_{t-1}\right)\boldsymbol{\Sigma}_{t-1}\mathbf{x}_t}{1 + \mathbf{x}_t^\top \boldsymbol{\Sigma}_{t-1}\mathbf{x}_t}$	$\mathbf{w}_t = \mathbf{w}_{t-1} + \frac{\left(y_t - \mathbf{x}_t^\top \mathbf{w}_{t-1}\right)\boldsymbol{\Sigma}_{t-1}\mathbf{x}_t}{1 + \mathbf{x}_t^\top \boldsymbol{\Sigma}_{t-1}\mathbf{x}_t}$	$\mathbf{w}_t = \mathbf{w}_{t-1} + \left(y_t - \mathbf{x}_t^\top \mathbf{w}_{t-1}\right) \boldsymbol{\Sigma}_{t-1}\mathbf{x}_t$
Output		$\mathbf{w}_T\ ,\ \boldsymbol{\Sigma}_T$			

Table 3.1: Second order online algorithms for stationary regression

and the algorithm predicts $\hat{y}_t = \mathbf{x}_t^\top \mathbf{w}_t$. Compared to ridge-regression, the AAR algorithm uses an additional input pair $(\mathbf{x}_t, 0)$. The AAR algorithm was shown to be last-step min-max optimal by Forster [1999], that is the predictions can be obtained by solving Eq. (3.1).

The AROWR algorithm [Vaits and Crammer, 2011, Crammer et al., 2012b], summarized in the left column of Table 3.1, is a modification of the AROW algorithm [Crammer et al., 2009] for regression. It maintains a Gaussian distribution parameterized by a mean $\mathbf{w}_t \in \mathbb{R}^d$ and a full covariance matrix $\boldsymbol{\Sigma}_t \in \mathbb{R}^{d \times d}$. Intuitively, the mean $\mathbf{w}_t$ represents a current linear function, while the covariance matrix $\boldsymbol{\Sigma}_t$ captures the uncertainty in the linear function $\mathbf{w}_t$. Given a new example $(\mathbf{x}_t, y_t)$ the algorithm uses its current mean to make a prediction $\hat{y}_t = \mathbf{x}_t^\top \mathbf{w}_{t-1}$. AROWR then sets the new distribution to be the solution of the following optimization problem,

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

Crammer et al. [2012b] derived regret bounds for this algorithm.

Comparing `WEMM` to other algorithms we note two differences. First, for the weight-vector update rule, we do not have the normalization term $1 + \mathbf{x}_t^\top \boldsymbol{\Sigma}_{t-1} \mathbf{x}_t$. Second, for the covariance matrix update rule, our algorithm gives non-constant scale to the increment by $\mathbf{x}_t \mathbf{x}_t^\top$. This scale $1/(1 - \mathbf{x}_t^\top \boldsymbol{\Sigma}_{t-1} \mathbf{x}_t)$ is small when the current instance $\mathbf{x}_t$ lies along the directions spanned by previously observed inputs $\{\mathbf{x}_i\}_{i=1}^{t-1}$, and large when the current instance $\mathbf{x}_t$ lies along previously unobserved directions.

Chapter 4

Analysis of the WEMM algorithm

In this chapter we analyze the `WEMM` algorithm in two steps. First, in Theorem 4 we show that the algorithm suffers a *constant* regret compared with the optimal weight vector $\mathbf{u}$ evaluated using *the weighted* loss, $L^{\boldsymbol{a}}(\mathbf{u})$. Second, in Theorem 5 and Theorem 6 we show that the difference of the weighted-loss $L^{\boldsymbol{a}}(\mathbf{u})$ to the true loss $L(\mathbf{u})$ is only logarithmic in T or in $L_T(\mathbf{u})$.

Theorem 4 *Assume* $1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t - a_t \leq 0$ *for* $t = 1 \ldots T$ *(which is satisfied by our choice later). Then, the loss of* `WEMM`*,* $\hat{y}_t = \mathbf{b}_{t-1}^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t$ *for* $t = 1 \ldots T$*, is upper bounded by,*

$$L_T(\mathit{WEMM}) \leq \inf_{\mathbf{u} \in \mathbb{R}^d} \left(b \left\| \mathbf{u} \right\|^2 + L_T^{\boldsymbol{a}}(\mathbf{u}) \right) .$$

Furthermore, if $1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t - a_t = 0$*, then the last inequality is in fact an equality.*

Proof: Using the Woodbury matrix identity we get

$$\mathbf{A}_t^{-1} = \mathbf{A}_{t-1}^{-1} - \frac{\mathbf{A}_{t-1}^{-1} \mathbf{x}_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1}}{\frac{1}{a_t} + \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t} , \tag{4.1}$$

therefore

$$\mathbf{A}_t^{-1} \mathbf{x}_t = \mathbf{A}_{t-1}^{-1} \mathbf{x}_t - \frac{\mathbf{A}_{t-1}^{-1} \mathbf{x}_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t}{\frac{1}{a_t} + \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t} = \frac{\mathbf{A}_{t-1}^{-1} \mathbf{x}_t}{1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t} . \tag{4.2}$$

For $t = 1 \ldots T$ we have

$$\begin{aligned}
& \ell_t(\texttt{WEMM}) + \inf_{\mathbf{u}\in\mathbb{R}^d} \left(b \left\|\mathbf{u}\right\|^2 + L^{\boldsymbol{a}}_{t-1}(\mathbf{u}) \right) - \inf_{\mathbf{u}\in\mathbb{R}^d} \left(b \left\|\mathbf{u}\right\|^2 + L^{\boldsymbol{a}}_{t}(\mathbf{u}) \right) \\
= \quad & (y_t - \hat{y}_t)^2 + \inf_{\mathbf{u}\in\mathbb{R}^d} \left(b \left\|\mathbf{u}\right\|^2 + \sum_{s=1}^{t-1} a_s \left(y_s - \mathbf{u}^\top \mathbf{x}_s \right)^2 \right) \\
& - \inf_{\mathbf{u}\in\mathbb{R}^d} \left(b \left\|\mathbf{u}\right\|^2 + \sum_{s=1}^{t} a_s \left(y_s - \mathbf{u}^\top \mathbf{x}_s \right)^2 \right) \\
\overset{Eq.\ (3.5)}{=} \quad & (y_t - \hat{y}_t)^2 + \sum_{s=1}^{t-1} a_s y_s^2 - \mathbf{b}_{t-1}^\top \mathbf{A}_{t-1}^{-1} \mathbf{b}_{t-1} - \sum_{s=1}^{t} a_s y_s^2 + \mathbf{b}_t^\top \mathbf{A}_t^{-1} \mathbf{b}_t \\
= \quad & (y_t - \hat{y}_t)^2 - a_t y_t^2 - \mathbf{b}_{t-1}^\top \mathbf{A}_{t-1}^{-1} \mathbf{b}_{t-1} + \mathbf{b}_t^\top \mathbf{A}_t^{-1} \mathbf{b}_t \\
\overset{Eq.\ (3.10)}{=} \quad & (y_t - \hat{y}_t)^2 - a_t y_t^2 - \mathbf{b}_{t-1}^\top \mathbf{A}_{t-1}^{-1} \mathbf{b}_{t-1} + \mathbf{b}_{t-1}^\top \mathbf{A}_t^{-1} \mathbf{b}_{t-1} + 2 a_t y_t \mathbf{b}_{t-1}^\top \mathbf{A}_t^{-1} \mathbf{x}_t + a_t^2 y_t^2 \mathbf{x}_t^\top \mathbf{A}_t^{-1} \mathbf{x}_t \\
= \quad & (y_t - \hat{y}_t)^2 - a_t y_t^2 - \mathbf{b}_{t-1}^\top \left(\mathbf{A}_{t-1}^{-1} - \mathbf{A}_t^{-1} \right) \mathbf{b}_{t-1} + 2 a_t y_t \mathbf{b}_{t-1}^\top \mathbf{A}_t^{-1} \mathbf{x}_t + a_t^2 y_t^2 \mathbf{x}_t^\top \mathbf{A}_t^{-1} \mathbf{x}_t \\
\overset{Eq.\ (3.14)}{=} \quad & (y_t - \hat{y}_t)^2 - a_t y_t^2 - \mathbf{b}_{t-1}^\top \mathbf{A}_t^{-1} a_t \mathbf{x}_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{b}_{t-1} + 2 a_t y_t \mathbf{b}_{t-1}^\top \mathbf{A}_t^{-1} \mathbf{x}_t + a_t^2 y_t^2 \mathbf{x}_t^\top \mathbf{A}_t^{-1} \mathbf{x}_t \\
= \quad & (y_t - \hat{y}_t)^2 - a_t y_t^2 + a_t \left(-\hat{y}_t \mathbf{b}_{t-1}^\top + 2 y_t \mathbf{b}_{t-1}^\top + a_t y_t^2 \mathbf{x}_t^\top \right) \mathbf{A}_t^{-1} \mathbf{x}_t \\
\overset{Eq.\ (4.2)}{=} \quad & (y_t - \hat{y}_t)^2 - a_t y_t^2 + a_t \left(-\hat{y}_t \mathbf{b}_{t-1}^\top + 2 y_t \mathbf{b}_{t-1}^\top + a_t y_t^2 \mathbf{x}_t^\top \right) \frac{\mathbf{A}_{t-1}^{-1} \mathbf{x}_t}{1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t} \\
= \quad & (y_t - \hat{y}_t)^2 + a_t \frac{-y_t^2 - y_t^2 a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t - \hat{y}_t^2 + 2 y_t \hat{y}_t + a_t y_t^2 \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t}{1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t} \\
= \quad & (y_t - \hat{y}_t)^2 - a_t \frac{(y_t - \hat{y}_t)^2}{1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t} \\
= \quad & \frac{1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t - a_t}{1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t} (y_t - \hat{y}_t)^2 \le 0 \ .
\end{aligned}$$

Summing over $t \in \{1, \ldots, T\}$ yields

$$L_T(\texttt{WEMM}) - \inf_{\mathbf{u}\in\mathbb{R}^d} \left(b \left\|\mathbf{u}\right\|^2 + L^{\boldsymbol{a}}_{T}(\mathbf{u}) \right) \le 0 \ .$$

■

Next we decompose the weighted loss $L^{\boldsymbol{a}}_T(\mathbf{u})$ into a sum of the actual loss $L_T(\mathbf{u})$ and a logarithmic term. We give two bounds - one is logarithmic in T

(Theorem 5), and the second is logarithmic in $L_T(\mathbf{u})$ (Theorem 6). We use the following notation of the loss suffered by $\mathbf{u}$ over the worst example,

$$S = S(\mathbf{u}) = \sup_{1 \le t \le T} \ell_t(\mathbf{u}),$$

where clearly S depends explicitly on $\mathbf{u}$, which is omitted for simplicity. We now turn to state our first result.

Theorem 5 *Assume* $\|\mathbf{x}_t\| \le 1$ *for* $t = 1 \ldots T$, *and* $b > 1$. *Assume further that* $a_t = \frac{1}{1 - \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t}$ *for* $t = 1 \ldots T$. *Then*

$$L_T^{\boldsymbol{a}}(\mathbf{u}) \le L_T(\mathbf{u}) + \frac{b}{b-1} S \ln \left| \frac{1}{b} \mathbf{A}_T \right| .$$

Proof: We decompose the weighted loss,

$$L_T^{\boldsymbol{a}}(\mathbf{u}) = L_T(\mathbf{u}) + \sum_t (a_t - 1)\ell_t(\mathbf{u}) \le L_T(\mathbf{u}) + S \sum_t (a_t - 1) . \tag{4.3}$$

Next we bound the term $\sum_t (a_t - 1)$. From Eq. (4.1) we see that $\mathbf{A}_t^{-1} \prec \mathbf{A}_{t-1}^{-1}$ and because $\mathbf{A}_0 = b\mathbf{I}$ we get

$$\mathbf{x}_t^\top \mathbf{A}_t^{-1} \mathbf{x}_t < \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t < \mathbf{x}_t^\top \mathbf{A}_{t-2}^{-1} \mathbf{x}_t < \ldots < \mathbf{x}_t^\top \mathbf{A}_0^{-1} \mathbf{x}_t = \frac{1}{b} \|\mathbf{x}_t\|^2 \le \frac{1}{b} ,$$

therefore $1 \le a_t \le \frac{1}{1 - \frac{1}{b}} = \frac{b}{b-1}$. From Eq. (4.2) we have

$$\begin{aligned} \mathbf{x}_t^\top \mathbf{A}_t^{-1} \mathbf{x}_t &= \frac{\mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t}{1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t} \\ &= \frac{1 - \frac{1}{a_t}}{1 + a_t \left(1 - \frac{1}{a_t}\right)} = \frac{a_t - 1}{a_t^2} , \end{aligned}$$

so we can bound the term $a_t - 1$ as following

$$a_t - 1 = a_t^2 \mathbf{x}_t^\top \mathbf{A}_t^{-1} \mathbf{x}_t \le \frac{b}{b-1} a_t \mathbf{x}_t^\top \mathbf{A}_t^{-1} \mathbf{x}_t . \tag{4.4}$$

With an argument similar to Forster [1999] we have,

$$a_t \mathbf{x}_t^\top \mathbf{A}_t^{-1} \mathbf{x}_t \le \ln \frac{|\mathbf{A}_t|}{\left|\mathbf{A}_t - a_t \mathbf{x}_t \mathbf{x}_t^\top\right|} = \ln \frac{|\mathbf{A}_t|}{|\mathbf{A}_{t-1}|} \ .$$

Summing the last inequality over t and using the initial value $\ln\left|\frac{1}{b}\mathbf{A}_0\right| = 0$ we get

$$\sum_{t=1}^{T} a_t \mathbf{x}_t^\top \mathbf{A}_t^{-1} \mathbf{x}_t \le \ln\left|\frac{1}{b}\mathbf{A}_T\right| \ .$$

Substituting the last inequality in Eq. (4.4) we get the logarithmic bound

$$\sum_{t=1}^{T} (a_t - 1) \le \frac{b}{b-1} \ln\left|\frac{1}{b}\mathbf{A}_T\right| \ ,$$

as required. ■

Next we show a bound that may be sub-logarithmic if the comparison vector $\mathbf{u}$ suffers sublinear amount of loss. Such a bound was previously proposed by Orabona et al. [2012]. We defer the discussion about the bound after providing the proof below.

Theorem 6 *Assume* $\|\mathbf{x}_t\| \le 1$ *for* $t = 1 \ldots T$*, and* $b > 1$*. Assume further that*

$$a_t = \frac{1}{1 - \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t} \tag{4.5}$$

for $t = 1 \ldots T$*. Then,*

$$L_T^{\boldsymbol{a}}(\mathbf{u}) \le L_T(\mathbf{u}) + \frac{b}{b-1} S d \left[1 + \ln\left(1 + \frac{L_T(\mathbf{u})}{Sd}\right)\right] \ .$$

We prove the theorem with a refined bound on the sum $\sum_t (a_t - 1)\ell_t(\mathbf{u})$ of Eq. (4.3) using the following two lemmas. In Theorem 5 we bound the loss of all examples with S and then bound the remaining term. Here, instead we show a relation to a subsequence "pretending" all examples of it as suffering a loss S, yet with the same cumulative loss, yielding an effective shorter sequence, which we then bound. In the next lemma we show how to find this subsequence, and in the following one bound the performance.

Lemma 7 *Let $I \subset \{1 \ldots T\}$ be the indices of the $T' = \left\lceil \sum_{t=1}^{T} \ell_t(\mathbf{u})/S \right\rceil$ largest elements of a_t, that is $|I| = T'$ and $\min_{t \in I} a_t \geq a_\tau$ for all $\tau \in \{1 \ldots T\}/I$. Then,*

$$\sum_{t=1}^{T} \ell_t(\mathbf{u})(a_t - 1) \leq S \sum_{t \in I} (a_t - 1) \ .$$

Proof: For a vector $\boldsymbol{v} \in \mathbb{R}^T$ define by $I(\boldsymbol{v})$ the set of indicies of the T' maximal absolute-valued elements of $\boldsymbol{v}$, and define $f(\boldsymbol{v}) = \sum_{t \in I(\boldsymbol{v})} |\boldsymbol{v}_t|$. The function $f(\boldsymbol{v})$ is a norm [Dekel et al., 2007] with a dual norm $g(\boldsymbol{h}) = \max\left\{ \|\boldsymbol{h}\|_\infty, \frac{\|\boldsymbol{h}\|_1}{T'} \right\}$. From the property of dual norms we have $\boldsymbol{v} \cdot \boldsymbol{h} \leq f(\boldsymbol{v}) g(\boldsymbol{h})$. Applying this inequality to $\boldsymbol{v} = (a_1 - 1, \ldots, a_T - 1)$ and $\boldsymbol{h} = (\ell_1(\mathbf{u}), \ldots, \ell_T(\mathbf{u}))$ we get,

$$\sum_{t=1}^{T} \ell_t(\mathbf{u})(a_t - 1) \leq \max\left\{ S, \frac{\sum_{t=1}^{T} \ell_t(\mathbf{u})}{T'} \right\} \sum_{t \in I} (a_t - 1) \ .$$

Combining with

$$ST' = S \left\lceil \sum_{t=1}^{T} \ell_t(\mathbf{u})/S \right\rceil \geq \sum_{t=1}^{T} \ell_t(\mathbf{u}) \ ,$$

completes the proof. ■

Note that the quantity $\sum_{t \in I} a_t$ is based only on T' examples, yet was generated using all T examples. In fact by running the algorithm with only these T' examples the corresponding sum cannot get smaller. Specifically, assume that the algorithm is run with inputs $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ and generated a corresponding sequence $(a_1, \ldots, a_T)$. Let I be the set of indices with maximal values of a_t as before. Assume that the algorithm is run with the subsequence of examples from I (with the same order) and generated $\alpha_1, \ldots, \alpha_T$ (where we set $\alpha_t = 0$ for $t \notin I$). Then, $\alpha_t \geq a_t$ for all $t \in I$. This statement follows from Eq. (3.3) from which we get that the matrix $\mathbf{A}_t$ is monotonically increasing in t. Thus, by removing examples we get another smaller matrix which leads to a larger value of α_t.

We continue the analysis with a sequence of length T' rather than a subsequence of the original sequence of length T being analyzed. The next lemma upper bounds the sum $\sum_t^{T'} a_t$ over T' inputs with another sum of same length, yet using orthonormal set of vectors of size d.

Lemma 8 *Let $\mathbf{x}_1, \ldots, \mathbf{x}_\tau$ be any τ inputs with unit-norm. Assume the algorithm is performing updates using Eq. (4.5) for some $\mathbf{A}_0$ resulting in a sequence $a_1, \ldots, a_\tau$. Let $E = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d\} \subset \mathbb{R}^d$ be an eigen-decomposition of $\mathbf{A}_0$ with corresponding eigenvalues $\lambda_1, \ldots, \lambda_d$. Then there exists a sequence of indices $j_1, \ldots, j_\tau$, where $j_i \in \{1, \ldots, d\}$, such that $\sum_t a_t \leq \sum_t \alpha_t$, where α_t are generated using Eq. (4.5) on the sequence $\boldsymbol{v}_{j_1}, \ldots, \boldsymbol{v}_{j_\tau}$.*

Additionally, let n_s be the number of times eigenvector $\boldsymbol{v}_s$ is used ($s = 1, \ldots, d$), that is $n_s = |\{j_t \ : \ j_t = s\}|$ (and $\sum_s n_s = \tau$), then,

$$\sum_t \alpha_t \leq \tau + \sum_{s=1}^{d} \sum_{r=1}^{n_s} \frac{1}{\lambda_s + r - 2} \ .$$

Proof: By induction over τ. For $\tau = 1$ we want to upper bound $a_1 = 1/(1 - \mathbf{x}_1^\top \mathbf{A}_0^{-1} \mathbf{x}_1)$ which is maximized when $\mathbf{x}_1 = \boldsymbol{v}_d$ the eigenvector with minimal eigenvalue λ_d. In this case we have $\alpha_1 = 1/(1 - 1/\lambda_d) = 1 + 1/(\lambda_d - 1)$, as desired.

Next we assume that the lemma holds for some $\tau - 1$ and show it for τ. Let $\mathbf{x}_1$ be the first input, and let $\{\gamma_s\}$ and $\{\mathbf{u}_s\}$ be the eigen-values and eigen-vectors of $\mathbf{A}_1 = \mathbf{A}_0 + a_1 \mathbf{x}_1 \mathbf{x}_1^\top$. The assumption of induction implies that $\sum_{t=2}^{\tau} \alpha_t \leq (\tau - 1) + \sum_{s=1}^{d} \sum_{r=1}^{n_s} \frac{1}{\gamma_s + r - 2}$. From Theorem 8.1.8 of Golub and Van Loan [1996] we know that the eigenvalues of $\mathbf{A}_1$ satisfy $\gamma_s = \lambda_s + m_s$ for some $m_s \geq 0$ and $\sum_s m_s = 1$. We thus conclude that

$$\sum_t a_t \leq 1 + 1/(\lambda_d - 1) + (\tau - 1) + \sum_{s=1}^{d} \sum_{r=1}^{n_s} \frac{1}{\lambda_s + m_s + r - 2} \ .$$

The last term is convex in $m_1, \ldots, m_d$ and thus is maximized over a vertex of the simplex, that is when $m_k = 1$ for some k and zero otherwise. In this case, the eigen-vectors $\{\mathbf{u}_s\}$ of $\mathbf{A}_1$ are in fact the eigenvectors $\{\boldsymbol{v}_s\}$ of $\mathbf{A}_0$, and the proof is completed. ∎

Equipped with these lemmas we now prove Theorem 6.

Proof: Let $T^{'} = \left\lceil \sum_{t=1}^{T} \ell_t\left(\mathbf{u}\right)/S \right\rceil$. Our starting point is the equality $L_T^{\boldsymbol{a}}(\mathbf{u}) = L_T(\mathbf{u}) + \sum_{t=1}^{T} \ell_t\left(\mathbf{u}\right)\left(a_t - 1\right)$ stated in Eq. (4.3). From Lemma 7 we get,

$$\sum_{t=1}^{T} \ell_t\left(\mathbf{u}\right)\left(a_t - 1\right) \leq S \sum_{t \in I} \left(a_t - 1\right) \leq S \sum_{t}^{T'} \left(\alpha_t - 1\right) \ , \tag{4.6}$$

where I is the subset of T' indices for which a_t are maximal, and α_t are the resulting coefficients computed with Eq. (4.5) using only the sub-sequence of examples $\mathbf{x}_t$ with $t \in I$.

By definition $\mathbf{A}_0 = b\mathbf{I}$ and thus from Lemma 8 we further bound Eq. (4.6) with,

$$\sum_{t=1}^{T} \ell_t\left(\mathbf{u}\right)\left(a_t - 1\right) \leq S \sum_{s=1}^{d} \sum_{r=1}^{n_s} \frac{1}{b + r - 2} \ , \tag{4.7}$$

for some n_s such that $\sum_s n_s = T'$. The last equation is maximized when all the counts n_s are about (as d may not divide T') the same, and thus we further bound Eq. (4.7) with,

$$\begin{aligned} \sum_{t=1}^{T} \ell_t\left(\mathbf{u}\right)\left(a_t - 1\right) \leq& \ S \sum_{s=1}^{d} \sum_{r=1}^{\lceil T'/d \rceil} \frac{1}{b + r - 2} \leq Sd \sum_{r=1}^{\lceil T'/d \rceil} \frac{b}{b-1}\frac{1}{r} \\ \leq& \ Sd\frac{b}{b-1}\left(1 + \ln\left(\left\lceil \frac{T'}{d} \right\rceil\right)\right) \\ \leq& \ Sd\frac{b}{b-1}\left(1 + \ln\left(1 + \frac{L_T\left(\mathbf{u}\right)}{Sd}\right)\right) \ , \end{aligned}$$

which completes the proof. ■

4.1 Comparison to other bounds

In this section we compare bounds of similar algorithms, summarized in Table 4.1. Our first bound[1] of Theorem 5 is most similar to the bounds of Forster [1999], Vovk [2001] and Crammer et al. [2012b]. Forster [1999] and Vovk [2001] have a multiplicative factor Y^2 of the logarithm, Crammer et al. [2012b] have the factor

[1]The bound in the table is obtained by noting that $\log\det$ is a concave function of the eigenvalues of the matrix, upper bounded when all the eigenvalues are equal (with the same trace).

Algorithm	Bound on Regret $R_T(\mathbf{u})$
[Vovk, 2001]	$b\left\|\mathbf{u}\right\|^2 + dY^2 \ln\left(1+\frac{T}{db}\right)$
[Forster, 1999]	$b\left\|\mathbf{u}\right\|^2 + dY^2 \ln\left(1+\frac{T}{db}\right)$
[Crammer et al., 2012b]	$rb\left\|\mathbf{u}\right\|^2 + dA \ln\left(1+\frac{T}{drb}\right)$
[Orabona et al., 2012]	$2\left\|\mathbf{u}\right\|^2 + d(U+Y)^2 \ln\left(1+\frac{2\left\|\mathbf{u}\right\|^2+\sum_t \ell_t(\mathbf{u})}{d(U+Y)^2}\right)$
Theorem 5	$b\left\|\mathbf{u}\right\|^2 + Sd\frac{b}{b-1} \ln\left(1+\frac{T}{d(b-1)}\right)$
Theorem 6	$b\left\|\mathbf{u}\right\|^2 + Sd\frac{b}{b-1} \ln\left(1+\frac{L_T(\mathbf{u})}{Sd}\right)$

Table 4.1: Comparison of regret bounds for online stationary regression

$A = \sup_{1\leq t\leq T} \ell_t(\text{alg})$, and we have the worst-loss of $\mathbf{u}$ over all examples (denoted by S). Thus, our first bound is better than the bound of Crammer et al. [2012b] (as often $S < A$), and better than the bounds of Forster [1999] and Vovk [2001] on problems that are approximately linear $y_t \approx \mathbf{u}\cdot\mathbf{x}_t$ for $t = 1,\ldots,T$ and Y is large, while their bound may be better if Y is small. Specifically, consider the case $y_t = \mathbf{u}\cdot\mathbf{x}_t$ for $t = 1,\ldots,T$. In this case we get that $S = 0$ and our bound of Theorem 5 is $\mathcal{O}(1)$ while the bounds of Forster [1999], Vovk [2001] and Crammer et al. [2012b] are $\mathcal{O}(\log T)$. Note that the analysis of Forster [1999] assumes that the labels y_t are bounded, and formally the algorithm should know this bound, while Crammer et al. [2012b] assume that the inputs are bounded, as we do.

Our second bound of Theorem 6 is similar to the bound of Orabona et al. [2012]. Both bounds have potentially sub-logarithmic regret as the cumulative loss $L(\mathbf{u})$ may be sublinear in T. Yet, their bound has a multiplicative factor of $(U+Y)^2$, while our bound has only the maximal loss S, which, as before, can be much smaller. Additionally, their analysis assumes that both the inputs $\mathbf{x}_t$ and the labels y_t are bounded, while we only assume that the inputs are bounded, and furthermore, our algorithm does not need to assume and know a compact set which contains $\mathbf{u}$ ($\left\|\mathbf{u}\right\| \leq U$), as opposed to their algorithm.

Chapter 5

Non-Stationary Regression

Consider the problem of prediction of life expectancy in some country. As the medicine advances and anti-aging techniques develop, there is a drift in the correct model that should be used to describe life expectancy. This is an example of prediction in non-stationary environment. Learning in such environments requires prediction algorithms to have the ability to track changes in the data.

For the stationary algorithms in the previous chapters (see Table 3.1), the update rule of $\mathbf{\Sigma}_t^{-1}$ forces the eigenvalues of $\mathbf{\Sigma}_t^{-1}$ to increase, and thus the eigenvalues of $\mathbf{\Sigma}_t$ go to zero with time. Thus, the effective learning rate goes to zero. As a consequence, these algorithms will gradually stop updating using instances which lie in the subspace of examples that were previously observed numerous times. This property leads to fast convergence in the stationary case, but at the same time to poor performance in the non-stationary case. It might happen that there is a need to update the prediction rule using an instance, yet the learning rate for this specific update is too small, and no useful update can be performed.

In this chapter we extend the last-step min-max rule of Forster [1999] to the non-stationary setting. In Sec. 5.1 we formally define the min-max problem for non-stationary setting, and our prediction algorithm will be the solution of this problem. In Sec. 5.2 we show that our algorithm can be expressed in a recursive form, similar to other algorithms. Finally, in Sec. 5.3 we compare our algorithm to other algorithms of the same form. The analysis of the algorithm appears in Chapter 6, where we derive a regret bound for the algorithm, and compare it to other bounds.

5.1 LASER algorithm

We extend the rule given in Eq. (3.1) to the non-stationary setting and re-define the last-step min-max predictor $\hat{y}_T$ to be,

$$\arg\min_{\hat{y}_T}\max_{y_T}\left[\sum_{t=1}^{T}\left(y_t-\hat{y}_t\right)^2-\min_{\mathbf{u}_1,..,\mathbf{u}_T}Q_T\left(\mathbf{u}_1,...,\mathbf{u}_T\right)\right], \tag{5.1}$$

where,

$$Q_t\left(\mathbf{u}_1,\ldots,\mathbf{u}_t\right)=b\left\|\mathbf{u}_1\right\|^2+c\sum_{s=1}^{t-1}\left\|\mathbf{u}_{s+1}-\mathbf{u}_s\right\|^2+\sum_{s=1}^{t}\left(y_s-\mathbf{u}_s^\top\mathbf{x}_s\right)^2 , \tag{5.2}$$

for some positive constants b, c. The first term of Eq. (5.1) is the loss suffered by the algorithm while $Q_t\left(\mathbf{u}_1,\ldots,\mathbf{u}_t\right)$ defined in Eq. (5.2) is a sum of the loss suffered by some sequence of linear functions $\left(\mathbf{u}_1,\ldots,\mathbf{u}_t\right)$ and a penalty for consecutive pairs that are far from each other, and for the norm of the first to be far from zero.

We develop the algorithm by solving the three optimization problems in Eq. (5.1), first, minimizing the inner term, $\min_{\mathbf{u}_1,..,\mathbf{u}_T}Q_T\left(\mathbf{u}_1,...,\mathbf{u}_T\right)$, second 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,

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

which clearly satisfies,

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

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

Lemma 9 *For* $t=2,3,\ldots$

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

Proof: We calculate

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

■

Using Lemma 9 we write explicitly the function $P_t(\mathbf{u}_t)$.

Lemma 10 *The following equality holds*

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

where,

$$D_1 = b\mathbf{I} + \mathbf{x}_1\mathbf{x}_1^\top , \quad D_t = \left(D_{t-1}^{-1} + c^{-1}\mathbf{I}\right)^{-1} + \mathbf{x}_t\mathbf{x}_t^\top \tag{5.3}$$

$$\mathbf{e}_1 = y_1\mathbf{x}_1 , \quad \mathbf{e}_t = \left(\mathbf{I} + c^{-1}D_{t-1}\right)^{-1}\mathbf{e}_{t-1} + y_t\mathbf{x}_t \tag{5.4}$$

$$f_1 = y_1^2 , \quad f_t = f_{t-1} - \mathbf{e}_{t-1}^\top\left(c\mathbf{I} + D_{t-1}\right)^{-1}\mathbf{e}_{t-1} + y_t^2 \tag{5.5}$$

Note that $D_t \in \mathbb{R}^{d\times d}$ is a positive definite matrix, $\mathbf{e}_t \in \mathbb{R}^{d\times 1}$ and $f_t \in \mathbb{R}$.

Proof: By definition,

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

and indeed $D_1 = b\mathbf{I} + \mathbf{x}_1\mathbf{x}_1^\top$, $\mathbf{e}_1 = y_1\mathbf{x}_1$, and $f_1 = y_1^2$.

We proceed by induction, assume that, $P_{t-1}(\mathbf{u}_{t-1}) = \mathbf{u}_{t-1}^\top D_{t-1}\mathbf{u}_{t-1} - 2\mathbf{u}_{t-1}^\top\mathbf{e}_{t-1} + f_{t-1}$. Applying Lemma 9 we get,

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

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

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

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

From Lemma 10 we conclude that,

$$\min_{\mathbf{u}_1,\ldots,\mathbf{u}_t} Q_t(\mathbf{u}_1,\ldots,\mathbf{u}_t) = \min_{\mathbf{u}_t} P_t(\mathbf{u}_t)$$

$$= \min_{\mathbf{u}_t} \left(\mathbf{u}_t^\top D_t \mathbf{u}_t - 2\mathbf{u}_t^\top \mathbf{e}_t + f_t \right) = -\mathbf{e}_t^\top D_t^{-1} \mathbf{e}_t + f_t \; . \tag{5.6}$$

Substituting Eq. (5.6) back in Eq. (5.1) we get that the last-step min-max predictor is given by,

$$\hat{y}_T = \arg\min_{\hat{y}_T} \max_{y_T} \left[\sum_{t=1}^{T} (y_t - \hat{y}_t)^2 + \mathbf{e}_T^\top D_T^{-1} \mathbf{e}_T - f_T \right] \; . \tag{5.7}$$

Since $\mathbf{e}_T$ depends on y_T we substitute Eq. (5.4) in the second term of Eq. (5.7),

$$\mathbf{e}_T^\top D_T^{-1} \mathbf{e}_T = \left(\left(\mathbf{I} + c^{-1} D_{T-1} \right)^{-1} \mathbf{e}_{T-1} + y_T \mathbf{x}_T \right)^\top D_T^{-1} \left(\left(\mathbf{I} + c^{-1} D_{T-1} \right)^{-1} \mathbf{e}_{T-1} + y_T \mathbf{x}_T \right) \; . \tag{5.8}$$

Substituting Eq. (5.8) and Eq. (5.5) in Eq. (5.7) and omitting terms not depending explicitly on y_T and $\hat{y}_T$ we get,

$$\begin{aligned} \hat{y}_T &= \arg\min_{\hat{y}_T} \max_{y_T} \left[(y_T - \hat{y}_T)^2 + y_T^2 \mathbf{x}_T^\top D_T^{-1} \mathbf{x}_T + 2 y_T \mathbf{x}_T^\top D_T^{-1} \left(\mathbf{I} + c^{-1} D_{T-1} \right)^{-1} \mathbf{e}_{T-1} - y_T^2 \right] \\ &= \arg\min_{\hat{y}_T} \max_{y_T} \left[\left(\mathbf{x}_T^\top D_T^{-1} \mathbf{x}_T \right) y_T^2 + 2 y_T \left(\mathbf{x}_T^\top D_T^{-1} \left(\mathbf{I} + c^{-1} D_{T-1} \right)^{-1} \mathbf{e}_{T-1} - \hat{y}_T \right) + \hat{y}_T^2 \right] \; . \end{aligned} \tag{5.9}$$

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 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 Eq. (5.9) over y_T is given by the following equation, since the optimal value is $y_T \in \{+Y, -Y\}$,

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

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

$$\hat{y}_T = \mathbf{x}_T^\top D_T^{-1} \left(\mathbf{I} + c^{-1} D_{T-1} \right)^{-1} \mathbf{e}_{T-1} = \mathbf{x}_T^\top D_T^{-1} D'_{T-1} \mathbf{e}_{T-1} \; , \tag{5.10}$$

```
Parameters: $0 < b < c$
Initialize: Set $D_0 = (bc)/(c-b)\,\mathbf{I} \in \mathbb{R}^{d\times d}$ and $\mathbf{e}_0 = \mathbf{0} \in \mathbb{R}^d$
For $t = 1, \ldots, T$ do
    • Receive an instance $\mathbf{x}_t$
    • Compute
          $D_t = \left(D_{t-1}^{-1} + c^{-1}\mathbf{I}\right)^{-1} + \mathbf{x}_t\mathbf{x}_t^\top$
    • Output prediction $\hat{y}_t = \mathbf{x}_t^\top D_t^{-1}\left(\mathbf{I} + c^{-1}D_{t-1}\right)^{-1}\mathbf{e}_{t-1}$
    • Receive the correct label $y_t$
    • Update:
          $\mathbf{e}_t = \left(\mathbf{I} + c^{-1}D_{t-1}\right)^{-1}\mathbf{e}_{t-1} + y_t\mathbf{x}_t$
Output: $\mathbf{e}_T$ , $D_T$
```

Figure 5.1: LASER: last-step adaptive regression algorithm.

where we define

$$D'_{t-1} = \left(\mathbf{I} + c^{-1}D_{t-1}\right)^{-1} \ . \tag{5.11}$$

We call the algorithm LASER for last-step adaptive regressor algorithm, and it is summarized in Fig. 5.1. This algorithm can be seen also as a forward algorithm [Azoury and Warmuth, 2001]: The predictor of Eq. (5.10) can be seen as the optimal linear *model* obtained over the same prefix of length $T-1$ and the new input $\mathbf{x}_T$ with fictional-label $y_T = 0$. Specifically, from Eq. (5.4) we get that if $y_T = 0$, then $\mathbf{e}_T = \left(\mathbf{I} + c^{-1}\mathbf{D}_{T-1}\right)^{-1}\mathbf{e}_{T-1}$. The prediction of the optimal predictor defined in Eq. (5.6) is $\mathbf{x}_T^\top\mathbf{u}_T = \mathbf{x}_T^\top\mathbf{D}_T^{-1}\mathbf{e}_T = \hat{y}_T$, where $\hat{y}_T$ was defined in Eq. (5.10).

When the variance V goes to zero, we set $c = \infty$ and thus we have $D_t = b\mathbf{I} + \sum_{s=1}^t \mathbf{x}_s\mathbf{x}_s^\top$ used in stationary algorithms ([Vovk, 2001, Forster, 1999, Hayes, 1996, Cesa-Bianchi et al., 2005]). In this case the algorithm reduces to the algorithm by Forster [1999].

5.2 Recursive form

Similar to the `WEMM` algorithm, `LASER` algorithm can also be expressed in a recursive form in terms of weight vector $\mathbf{w}_t$ and a covariance-like matrix $\mathbf{\Sigma}_t$. We denote $\mathbf{w}_t = \mathbf{D}_t^{-1}\mathbf{e}_t$ and $\mathbf{\Sigma}_t = \mathbf{D}_t^{-1}$, and develop recursive update rules for $\mathbf{w}_t$ and $\mathbf{\Sigma}_t$:

$$\mathbf{\Sigma}_t^{-1} = \left(\mathbf{\Sigma}_{t-1} + c^{-1}\mathbf{I}\right)^{-1} + \mathbf{x}_t\mathbf{x}_t^\top \ ,$$

$$\begin{aligned}
\mathbf{w}_t &= \mathbf{D}_t^{-1}\mathbf{e}_t \\
&= \left[\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)^{-1} + \mathbf{x}_t\mathbf{x}_t^\top\right]^{-1} \left[\left(\mathbf{I} + c^{-1}\mathbf{D}_{t-1}\right)^{-1}\mathbf{e}_{t-1} + y_t\mathbf{x}_t\right] \\
&= \left[\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I} - \frac{\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t\mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)}{1 + \mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t}\right] \left[\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)^{-1}\mathbf{D}_{t-1}^{-1}\mathbf{e}_{t-1} + y_t\mathbf{x}_t\right] \\
&= \left[\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I} - \frac{\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t\mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)}{1 + \mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t}\right] \left[\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)^{-1}\mathbf{w}_{t-1} + y_t\mathbf{x}_t\right] \\
&= \mathbf{w}_{t-1} - \frac{\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t\mathbf{x}_t^\top\mathbf{w}_{t-1}}{1 + \mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t} + \left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)y_t\mathbf{x}_t \\
&\quad - \frac{\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t\mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)}{1 + \mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t} y_t\mathbf{x}_t \\
&= \mathbf{w}_{t-1} + \frac{\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)y_t\mathbf{x}_t - \left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t\mathbf{x}_t^\top\mathbf{w}_{t-1}}{1 + \mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t} \\
&= \mathbf{w}_{t-1} + \frac{\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t\left(y_t - \mathbf{x}_t^\top\mathbf{w}_{t-1}\right)}{1 + \mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t} \\
&= \mathbf{w}_{t-1} + \frac{\left(y_t - \mathbf{x}_t^\top\mathbf{w}_{t-1}\right)\left(\mathbf{\Sigma}_{t-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t}{1 + \mathbf{x}_t^\top\left(\mathbf{\Sigma}_{t-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t} \ ,
\end{aligned}$$

and the prediction is

$$\begin{aligned}
\hat{y}_t &= \mathbf{x}_t^\top\mathbf{D}_t^{-1}\left(\mathbf{I} + c^{-1}\mathbf{D}_{t-1}\right)^{-1}\mathbf{e}_{t-1} \\
&= \mathbf{x}_t^\top\left[\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I} - \frac{\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t\mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)}{1 + \mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t}\right]\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)^{-1}\mathbf{D}_{t-1}^{-1}\mathbf{e}_{t-1} \\
&= \mathbf{x}_t^\top\left[\mathbf{I} - \frac{\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t\mathbf{x}_t^\top}{1 + \mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t}\right]\mathbf{D}_{t-1}^{-1}\mathbf{e}_{t-1}
\end{aligned}$$

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

Table 5.1: Second order online algorithms for non-stationary regression

$$
\begin{aligned}
&= \mathbf{x}_t^\top \left[\frac{1 + \mathbf{x}_t^\top \left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t - \left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t\mathbf{x}_t^\top}{1 + \mathbf{x}_t^\top \left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t} \right] \mathbf{w}_{t-1} \\
&= \frac{\mathbf{x}_t^\top \mathbf{w}_{t-1}}{1 + \mathbf{x}_t^\top \left(\mathbf{\Sigma}_{t-1} + c^{-1}\mathbf{I}\right)\mathbf{x}_t} \ .
\end{aligned}
$$

A summary of the algorithm in a recursive form appears in the right column of Table 5.1.

5.3 Comparison to other algorithms

In this section we compare similar second order online algorithms for non-stationary regression. The CR-RLS algorithm [Salgado et al., 1988, Goodwin et al., 1983, Chen and Yen, 1999], summarized in the middle column of Table 5.1, is a variant of the RLS algorithm for non-stationary environment. The algorithm performs a step called covariance-reset, which resets the second-order information every fixed amount of rounds. This step enables the algorithm not to converge or get stuck, which is an essential property in non-stationary environment. The only difference of CR-RLS from RLS is that after updating the matrix $\mathbf{\Sigma}_t$, 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.

The ARCOR algorithm [Vaits and Crammer, 2011], summarized in the left column of Table 5.1, is a variant of the AROWR algorithm [Vaits and Crammer, 2011, Crammer et al., 2012b] for non-stationary environment. The algorithm is similar to CR-RLS algorithm but it resets the covariance matrix each time the smallest eigenvalue of the covariance matrix reaches a certain threshold. Thus, the ARCOR algorithm performs resets based on the actual properties of the data: the eigenspectrum of the covariance matrix.

Table 5.1 enables us to compare the three algorithms head-to-head. All algorithms perform linear predictions, and then update the prediction vector $\mathbf{w}_t$ and the matrix $\mathbf{\Sigma}_t$. 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. CR-RLS (as well as RLS) also uses a forgetting factor (if $r < 1$).

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

Chapter 6

Analysis of the LASER algorithm

In this chapter we analyze the performance of the `LASER` algorithm in the worst-case setting in six steps. First, state a technical lemma (Lemma 11) that is used in the second step (Theorem 12), in which we bound the regret with a quantity proportional to $\sum_{t=1}^{T} \mathbf{x}_t^\top D_t^{-1} \mathbf{x}_t$. Third, in Lemma 13 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 14) and fifth (Lemma 15) steps we bound the eigenvalues of D_t first for scalars and then extend the results to matrices. Finally, in Corollary 16 we put all these results together and get the desired bound.

Lemma 11 *For $t \geq 1$ the following statement holds,*

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

where as defined in Eq. (5.11) we have $D'_{t-1} = \left(\mathbf{I} + c^{-1} D_{t-1} \right)^{-1}$.

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

$$D_t^{-1} = \left[\left(D_{t-1}^{-1} + c^{-1} \mathbf{I} \right)^{-1} + \mathbf{x}_t \mathbf{x}_t^\top \right]^{-1} = D_{t-1}^{-1} + c^{-1} \mathbf{I} - \frac{\left(D_{t-1}^{-1} + c^{-1} \mathbf{I} \right) \mathbf{x}_t \mathbf{x}_t^\top \left(D_{t-1}^{-1} + c^{-1} \mathbf{I} \right)}{1 + \mathbf{x}_t^\top \left(D_{t-1}^{-1} + c^{-1} \mathbf{I} \right) \mathbf{x}_t}$$

$$\left(\mathbf{I} + c^{-1} D_{t-1} \right)^{-1} = \mathbf{I} - c^{-1} \left(D_{t-1}^{-1} + c^{-1} \mathbf{I} \right)^{-1} .$$

Multiplying both identities with each other we get,

$$D_t^{-1} \left(\mathbf{I} + c^{-1} D_{t-1} \right)^{-1} = D_{t-1}^{-1} - \frac{\left(D_{t-1}^{-1} + c^{-1} \mathbf{I} \right) \mathbf{x}_t \mathbf{x}_t^\top D_{t-1}^{-1}}{1 + \mathbf{x}_t^\top \left(D_{t-1}^{-1} + c^{-1} \mathbf{I} \right) \mathbf{x}_t} , \tag{6.1}$$

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

$$\left(\mathbf{I}+c^{-1}D_{t-1}\right)^{-1} D_t^{-1} = D_{t-1}^{-1} - \frac{D_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^\top\left(D_{t-1}^{-1}+c^{-1}\mathbf{I}\right)}{1+\mathbf{x}_t^\top\left(D_{t-1}^{-1}+c^{-1}\mathbf{I}\right)\mathbf{x}_t} \; . \tag{6.2}$$

Finally, from Eq. (6.1) we get,

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

We further develop the last equality and use Eq. (6.1) and Eq. (6.2) in the second equality below,

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

■

We next bound the cumulative loss of the algorithm.

Theorem 12 *Assume that the labels are bounded* $\sup_t |y_t| \leq Y$ *for some* $Y \in \mathbb{R}$. *Then the following bound holds,*

$$L_T(\texttt{LASER}) \leq \min_{\mathbf{u}_1,\ldots,\mathbf{u}_T}\left[L_T(\{\mathbf{u}_t\}) + cV_T(\{\mathbf{u}_t\}) + b\left\|\mathbf{u}_1\right\|^2\right] + Y^2\sum_{t=1}^{T}\mathbf{x}_t^\top D_t^{-1}\mathbf{x}_t \; . \tag{6.3}$$

Proof: Fix t.

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

where the last equality follows from Eq. (5.4) and Eq. (5.5). We proceed to develop the last equality,

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

Substituting the specific value of the predictor $\hat{y}_t = \mathbf{x}_t^\top D_t^{-1}D'_{t-1}\mathbf{e}_{t-1}$ from Eq. (5.10), we proceed to develop the last equality,

$$\begin{aligned}
= \; & \hat{y}_t^2 + y_t^2\mathbf{x}_t^\top D_t^{-1}\mathbf{x}_t + \mathbf{e}_{t-1}^\top\Bigg[- D_{t-1}^{-1} + D'_{t-1}\left(D_t^{-1}D'_{t-1} + c^{-1}\mathbf{I}\right)\Bigg]\mathbf{e}_{t-1} \\
= \; & \mathbf{e}_{t-1}^\top D'_{t-1}D_t^{-1}\mathbf{x}_t\mathbf{x}_t^\top D_t^{-1}D'_{t-1}\mathbf{e}_{t-1} + y_t^2\mathbf{x}_t^\top D_t^{-1}\mathbf{x}_t \\
& + \mathbf{e}_{t-1}^\top\Bigg[- D_{t-1}^{-1} + D'_{t-1}\left(D_t^{-1}D'_{t-1} + c^{-1}\mathbf{I}\right)\Bigg]\mathbf{e}_{t-1} \\
= \; & \mathbf{e}_{t-1}^\top\tilde{D}_t\mathbf{e}_{t-1} + y_t^2\mathbf{x}_t^\top D_t^{-1}\mathbf{x}_t \;, \qquad (6.4)
\end{aligned}$$

where $\tilde{D}_t = D'_{t-1}D_t^{-1}\mathbf{x}_t\mathbf{x}_t^\top D_t^{-1}D'_{t-1} - D_{t-1}^{-1} + D'_{t-1}\left(D_t^{-1}D'_{t-1} + c^{-1}\mathbf{I}\right)$. Using Lemma 11 we upper bound $\tilde{D}_t \preceq 0$ and thus Eq. (6.4) is bounded,

$$y_t^2\mathbf{x}_t^\top D_t^{-1}\mathbf{x}_t \le Y^2\mathbf{x}_t^\top D_t^{-1}\mathbf{x}_t \ .$$

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

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

■

In the next lemma we further bound the right term of Eq. (6.3). This type of bound is based on the usage of the covariance-like matrix D.

Lemma 13

$$\sum_{t=1}^T \mathbf{x}_t^\top D_t^{-1}\mathbf{x}_t \le \ln\left|\frac{1}{b}D_T\right| + c^{-1}\sum_{t=1}^T \mathrm{Tr}\left(D_{t-1}\right) \ . \qquad (6.5)$$

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

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

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

$$\begin{aligned}
\mathbf{x}_t^\top D_t^{-1}\mathbf{x}_t &\le -\sum_{j=1}^d \ln\left[\lambda_j\left(D_t^{-1/2}B_tD_t^{-1/2}\right)\right]\\
&= -\ln\left[\prod_{j=1}^d \lambda_j\left(D_t^{-1/2}B_tD_t^{-1/2}\right)\right]
\end{aligned}$$

$$
\begin{aligned}
&= -\ln\left|D_t^{-1/2} B_t D_t^{-1/2}\right| \\
&= \ln\frac{|D_t|}{|B_t|} = \ln\frac{|D_t|}{\left|D_t - \mathbf{x}_t\mathbf{x}_t^\top\right|} \ .
\end{aligned}
$$

It follows that,

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

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

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

■

At first sight it seems that the right term of Eq. (6.5) 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 Eq. (6.5) is not growing too fast, which will allow us to obtain a sublinear regret bound. Lemma 14 analyzes the properties of the recursion of D defined in Eq. (5.3) for scalars, that is $d = 1$. In Lemma 15 we extend this analysis to matrices.

Lemma 14 *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 $f(\lambda) = \lambda\beta / (\lambda+\beta) + x^2 \leq \beta \times 1 + x^2$. 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 14 to bound the maximal eigenvalue of the matrices D_t.

Lemma 15 *Assume $\|\mathbf{x}_t\|^2 \leq X^2$ for some X. Then, the eigenvalues of D_t (for $t \geq 1$), denoted by $\lambda_i(D_t)$, are upper bounded by*

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

Proof: By induction. From Eq. (5.3) 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 Eq. (5.3) we have,

$$\begin{aligned} D_t &= \left(D_{t-1}^{-1} + c^{-1}\mathbf{I}\right)^{-1} + \mathbf{x}_t\mathbf{x}_t^\top \preceq \left(D_{t-1}^{-1} + c^{-1}\mathbf{I}\right)^{-1} + \mathbf{I}\,\|\mathbf{x}_t\|^2 \\ &= \sum_i^d \boldsymbol{v}_i\boldsymbol{v}_i^\top \left(\left(\lambda_i^{-1}+c^{-1}\right)^{-1} + \|\mathbf{x}_t\|^2\right) = \sum_i^d \boldsymbol{v}_i\boldsymbol{v}_i^\top \left(\frac{\lambda_i c}{\lambda_i + c} + \|\mathbf{x}_t\|^2\right) . \end{aligned} \tag{6.6}$$

Plugging Lemma 14 in Eq. (6.6) we get,

$$D_t \preceq \sum_i^d \boldsymbol{v}_i\boldsymbol{v}_i^\top \max\left\{\frac{3X^2+\sqrt{X^4+4X^2c}}{2}, b+X^2\right\}$$

$$= \max\left\{\frac{3X^2+\sqrt{X^4+4X^2c}}{2}, b+X^2\right\}\mathbf{I}\,,$$

as required. ■

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

Corollary 16 *Assume $\|\mathbf{x}_t\|^2 \le X^2$, $|y_t| \le Y$. Then,*

$$\begin{aligned} L_T(\mathit{LASER}) \quad &\le \quad b\|\mathbf{u}_1\|^2 + L_T(\{\mathbf{u}_t\}) + Y^2 \ln\left|\frac{1}{b}\mathbf{D}_T\right| + c^{-1}Y^2\mathrm{Tr}\left(\mathbf{D}_0\right) + cV \\ &\qquad + c^{-1}Y^2Td\max\left\{\frac{3X^2+\sqrt{X^4+4X^2c}}{2}, b+X^2\right\}\ . \qquad (6.7)\end{aligned}$$

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

$$c = \frac{\sqrt{2}TY^2dX}{V^{2/3}} \qquad (6.8)$$

we have,

$$\begin{aligned} L_T(\mathit{LASER}) \quad &\le \quad b\|\mathbf{u}_1\|^2 + 3\left(\sqrt{2}Y^2dX\right)^{2/3}T^{2/3}V^{1/3} + \frac{\varepsilon}{1-\varepsilon}Y^2d + L_T(\{\mathbf{u}_t\}) \\ &\qquad + Y^2\ln\left|\frac{1}{b}D_T\right|\ . \qquad (6.9)\end{aligned}$$

Proof: Plugging Lemma 13 in Theorem 12 we have for all $(\mathbf{u}_1, \ldots, \mathbf{u}_T)$,

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

where the last inequality follows from Lemma 15. The term $c^{-1}Y^2\mathrm{Tr}\left(D_0\right)$ does not depend on T, because

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

Next, note that

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

We thus have that the right term of Eq. (6.7) is upper bounded,

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

Using this bound and plugging the value of c from Eq. (6.8) we bound Eq. (6.7),

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

which concludes the proof. ■

6.1 Comparison to other bounds

In this section we compare bounds of non-stationary regression algorithms. First, note that the regret bound of `LASER` in Eq. (6.9) is of the order of $T^{2/3}V^{1/3}$ and thus the algorithm maintains an average loss close to that of the best slowly changing sequence of linear functions, as long as the total of drift V is sublinear. The regret of the ARCOR algorithm [Vaits and Crammer, 2011] depends on the total drift as $\sqrt{TV'}\log(T)$, where their definition of total drift is a sum of the Euclidean differences $V' = \sum_t^{T-1}\left\|\mathbf{u}_{t+1}-\mathbf{u}_t\right\|$, rather than the squared norm. The two bounds are not comparable in general. However, we can compare the bounds on two concrete analytical examples:

1. Assume a constant instantaneous drift $\|\mathbf{u}_{t+1} - \mathbf{u}_t\| = \nu$ for some constant value ν. In this case the variance and squared variance are, $V' = T\nu$ and $V = T\nu^2$. The bound of ARCOR becomes $\nu^{\frac{1}{2}} T \log T$, while the bound of `LASER` becomes $\nu^{\frac{2}{3}} T$. The bound of ARCOR is larger if $(\log T)^6 > \nu$, and the bound of `LASER` is larger in the opposite case.

2. Assume a polynomial decay of the drift, $\|\mathbf{u}_{t+1} - \mathbf{u}_t\| \leq t^{-\kappa}$ for some $\kappa > 0$. In this case, for $\kappa \neq 1$ we get $V' \leq \sum_{t=1}^{T-1} t^{-\kappa} \leq \int_1^{T-1} t^{-\kappa} dt + 1 = \frac{(T-1)^{1-\kappa} - \kappa}{1-\kappa}$. For $\kappa = 1$ we get $V' \leq \log(T-1) + 1$. For `LASER` we have, for $\kappa \neq 0.5$, $V \leq \sum_{t=1}^{T-1} t^{-2\kappa} \leq \int_1^{T-1} t^{-2\kappa} dt + 1 = \frac{(T-1)^{1-2\kappa} - 2\kappa}{1-2\kappa}$. For $\kappa = 0.5$ we get $V \leq \log(T-1) + 1$. Asymptotically, `LASER` outperforms ARCOR about when $\kappa < 0.7$.

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 $L_T(\{\mathbf{u}_t\})$, leading to a small regret only when the data is close to be realizable with linear models. 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 $V = o(1)$, while we require only $V = o(T)$.

Chapter 7

Simulations

We evaluate the LASER algorithm on four synthetic datasets. We set $T = 2000$ and $d = 20$. For all datasets, the inputs $\mathbf{x}_t \in \mathbb{R}^{20}$ were generated such that 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 were drawn from independent Gaussian distributions $\mathcal{N}(0, 2)$. The first synthetic dataset was generated using a sequence of vectors $\mathbf{u}_t \in \mathbb{R}^{20}$ 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 (linear drift). Specifically, we have $\|\mathbf{u}_t\| = 1$ and the instantaneous drift $\|\mathbf{u}_t - \mathbf{u}_{t-1}\|$ is constant. The second synthetic dataset was generated using a sequence of vectors $\mathbf{u}_t \in \mathbb{R}^{20}$ for which the only non-zero coordinates are the first two. This vector in $\mathbb{R}^2$ is of unit norm $\|\mathbf{u}_t\| = 1$ and rotating in a rate of t^{-1} (sublinear drift). In addition every 50 time-steps the two-dimensional vector defined above was "embedded" in different pair of coordinates of the reference vector $\mathbf{u}_t$, for the first 50 steps it were coordinates $1, 2$, in the next 50 examples, coordinates $3, 4$, and so on. This change causes a switch in the reference vector $\mathbf{u}_t$. For the first two datasets we set $y_t = \mathbf{x}_t^\top \mathbf{u}_t$ (linear data). The third and fourth datasets are the same as first and second except we set $y_t = \mathbf{x}_t^\top \mathbf{u}_t + n_t$ where $n_t \sim \mathcal{N}(0, 0.05)$ (noisy data).

We compared five algorithms: NLMS (normalized least mean square) [Bershad, 1986, Bitmead and Anderson, 1980] which is a state-of-the-art first-order algorithm, AROWR (AROW for Regression) [Crammer et al., 2009], ARCOR [Vaits and Crammer, 2011], CR-RLS [Salgado et al., 1988, Goodwin et al., 1983, Chen and Yen, 1999] and our LASER algorithm. The algorithms' parameters were tuned using a single random sequence. We repeat each experiment 100 times reporting

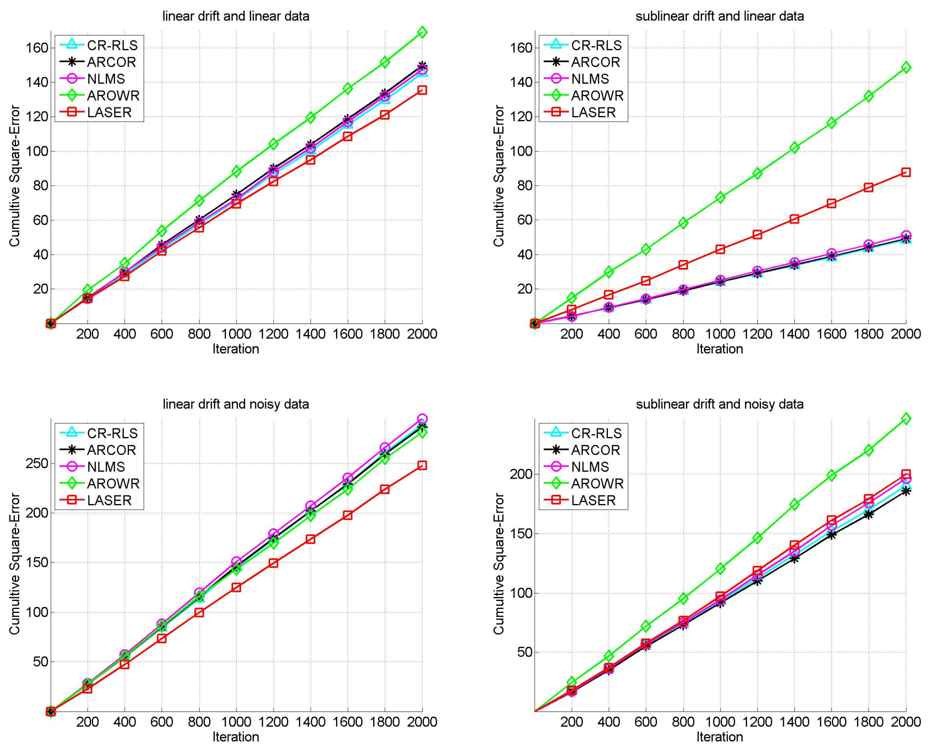

Figure 7.1: Cumulative squared loss for NLMS, AROWR, ARCOR, CR-RLS and LASER vs iteration.

the mean cumulative square-loss.

The results for the first dataset are summarized in the top-left plot of Fig. 7.1. For this dataset AROWR performs the worst, since it converges very fast and does not allow the ability of tracking changes in data. The ARCOR algorithm for this dataset performs relatively bad, this is due to the fact that it is designed for sublinear amount of data drift, while this dataset has linear drift level, which is the worst case assumed by `LASER`. CR-RLS and NLMS show better results, since they both don't make assumptions on the drift level. CR-RLS performs a little bit better than NLMS since it has faster learning rate due to the use of second order information. Finally, LASER performs the best as expected. It has a good tracking ability, fast learning rate and it is designed to perform well in severe conditions like linear non-stationarity of the data. Similar results we get for the third dataset (bottom-left plot of Fig. 7.1).

For the second and fourth datasets (right plots of Fig. 7.1), where we have sublinear drift level, we get that ARCOR outperforms `LASER` since it is especially designed for sublinear amount of data drift.

Chapter 8

Related work

The problem of predicting reals in an online manner was studied for more than five decades. Clearly we cannot cover all previous work here, and the reader is refered to the encyclopedic book of Cesa-Bianchi and Lugosi [2006] for a full survey.

Widrow and Hoff [1960] studied a gradient descent algorithm for the squared loss. Many variants of the algorithm were studied since then. A notable example is the normalized least mean squares algorithm (NLMS) [Bitmead and Anderson, 1980, Bershad, 1986] that adapts to the input's scale. More gradient descent based algorithms and bounds for regression with the squared loss were proposed by Cesa-Bianchi et al. [1993] about two decades ago. These algorithms were generalized and extended by Kivinen and Warmuth [1997] using additional regularization functions.

An online version of the ridge regression algorithm in the worst-case setting was proposed and analyzed by Foster [1991]. A related algorithm called the Aggregating Algorithm (AA) was studied by Vovk [1990], and later applied to the problem of linear regression with square loss [Vovk, 1997, 2001]. See also the work of Azoury and Warmuth [2001].

The recursive least squares (RLS) [Hayes, 1996] is a similar algorithm proposed for adaptive filtering. A variant of the RLS algorithm (AROW for regression [Vaits and Crammer, 2011]) was analysed by Crammer et al. [2012b]. All algorithms make use of second order information, as they maintain a weight-vector and a covariance-like positive semi-definite (PSD) matrix used to re-weight the input. The eigenvalues of this covariance-like matrix grow with time t, a property which is used to prove logarithmic regret bounds. Orabona et al. [2012] showed that beyond logarithmic regret bound can be achieved when the total best linear model loss is sublinear in T. We derive a similar bound, with a multiplicative factor that depends on the worst-loss of the competitor $\mathbf{u}$, rather than a bound Y on the labels.

In the context of online linear optimization, Hazan and Kale [2008] developed regret bounds that depend on the variance of the side information used to define the loss sequence. Formally, they considered the case in which the loss functions have a small variation, defined by $V = \sum_{t=1}^{T} \|f_t - \mu\|^2$, where $\mu = \sum_{t=1}^{T} f_t/T$ is the average of the loss functions. For this, they showed that a regret of $\mathcal{O}(\sqrt{V})$ can be achieved, and they also have an analogous result for the prediction with expert advice problem. Later, Hazan and Kale [2009] considered the portfolio management problem, in which the loss functions are restricted to a specific form, and showed better regret of $\mathcal{O}(\log V)$. A natural question that arises from these bounds is whether it is possible to get tighter variance-based bounds where the variance is $V_1 = \sum_{t=1}^{T} \|f_t - f_{t-1}\|^2$. This question solved recently by Chiang et al. [2012] which showed a regret of $\mathcal{O}(\sqrt{V_1})$. More tighter variance-based bounds for online linear optimization were proved by Rakhlin and Sridharan [2012]. These bounds take advantage of benign (as opposed to worst-case) sequences that can be described well by a known "predictable process". It is important to note that in the regression case that we consider in this thesis, these variance-based bounds correspond to bounds that depend on the variance of the instance vectors $\mathbf{x}_t$, rather than on the loss of the competitor, as the bound of Orabona et al. [2012] and our bound. In addition, our bounds are valid in the worst-case, under no assumptions on the mechanism generating the sequence of instances.

The derivation of the `WEMM` algorithm shares similarities with the work of Forster [1999]. Both algorithms are motivated from the last-step min-max predictor. Yet, the formulation of Forster [1999] yields a convex optimization for which the max operation over y_t is not bounded, and thus he used an artificial clipping operation to avoid unbounded solutions. With a proper tuning of the weight a_t, we are able to obtain a problem that is convex in $\hat{y}_t$ and concave in y_t, and thus well defined.

The derivation of the `LASER` algorithm also shares similarities with the work of Forster [1999]. While the algorithm of Forster [1999] is designed for the stationary setting, `LASER` algorithm is primarily designed for the non-stationary setting. This algorithm is mostly close to a recent algorithm [Vaits and Crammer, 2011] called ARCOR (Adaptive Regularization with Covariance Reset). The ARCOR algorithm is based on the AROWR algorithm with an additional projection step, and it controls the eigenvalues of a covariance-like matrix using scheduled resets.

The Covariance Reset RLS algorithm (CR-RLS) [Chen and Yen, 1999, Salgado et al., 1988, Goodhart et al., 1991] is another example of an algorithm that resets a covariance matrix but every fixed amount of data points, as opposed to ARCOR that performs these resets adaptively. All of these algorithms that were designed to have numerically stable computations, perform covariance reset from time to time. Our algorithm, `LASER`, is simpler as it does not involve these steps, and it controls the increase of the eigenvalues of the covariance matrix implicitly rather than explicitly by "averaging" it with a fixed diagonal matrix (see Eq. (5.3)). The Kalman filter [Kalman, 1960] and the H_∞ algorithm (e.g. [Simon, 2006]) designed for filtering take a similar approach, yet the exact algebraic form is different.

ARCOR also controls explicitly the norm of the weight vector, which is used for its analysis, by projecting it into a bounded set, as was also proposed by Herbster and Warmuth [2001]. Other approaches to control its norm are to shrink it multiplicatively [Kivinen et al., 2001] or by removing old examples [Cavallanti et al., 2007]. Some of these algorithms were designed to have sparse functions in the kernel space (e.g. [Crammer et al., 2003, Dekel et al., 2005]). Note that our algorithm `LASER` is simpler as it does not perform any of these operation explicitly. Finally, few algorithms that employ second order information were recently proposed for classification [Cesa-Bianchi et al., 2005, Crammer et al., 2009, 2012a], and later in the online convex programming framework [Duchi et al., 2010, McMahan and Streeter, 2010].

Chapter 9

Summary and Conclusions

We proposed a modification of the last-step min-max algorithm [Forster, 1999] using weights over examples (WEMM), and showed how to choose these weights for the problem to be well defined – convex – which enabled us to develop the last-step min-max predictor, without requiring the labels to be bounded. Our algorithmic formulations depend on inner- and outer-products and thus can be employed with kernel functions. Our analysis bounds the regret with quantities that depend only on the loss of the competitor, with no need for any knowledge of the problem.

We also proposed a novel algorithm (LASER) for non-stationary online regression designed and analyzed with the squared loss. The algorithm was developed from the last-step min-max predictor for *non-stationary* problems, and we showed an exact recursive form of its solution. Simulations showed the superior performance of our algorithm, especially in a worst-case (constant per iteration) drift.

An interesting future direction is to extend the algorithms for general loss functions rather than the squared loss, or to classification tasks. Additionally, for the LASER algorithm to perform well, the amount of drift V or a bound over it should be known in advance. An interesting direction is to design algorithms that automatically detect the level of drift, or do not need this information before run-time.

The derivation of the LASER algorithm does not use the notation of weighted loss. An interesting direction is to try to incorporate in the derivation of the LASER algorithm the technique of weighted loss used for the derivation of the WEMM algorithm.

Bibliography

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

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

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

Robert R. Bitmead and Brian 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 *ALT*, 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.

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

Nicolò 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.

Chao-Kai Chiang, Tianbao Yang, Chia-Jung Lee, Mehrdad Mahdavi, Chi-Jen Lu, Rong Jin, and Shenghuo Zhu. Online optimization with gradual variations. *Journal of Machine Learning Research - Proceedings Track*, 23:6.1–6.20, 2012.

Koby Crammer, Jaz S. Kandola, and Yoram Singer. Online classification on a budget. In *NIPS*, 2003.

Koby Crammer, Alex Kulesza, and Mark Dredze. Adaptive regularization of weighted vectors. In *Advances in Neural Information Processing Systems 23*, 2009.

Koby Crammer, Mark Dredze, and Fernando Pereira. Confidence-weighted linear classification for text categorization. *J. Mach. Learn. Res.*, 98888:1891–1926, June 2012a. ISSN 1532-4435.

Koby Crammer, Alex Kulesza, and Mark Dredze. New $\mathcal{H}\infty$ bounds for the recursive least squares algorithm exploiting input structure. In *ICASSP*, pages 2017–2020, 2012b.

Ofer Dekel, Shai Shalev-shwartz, and Yoram Singer. The forgetron: A kernel-based perceptron on a fixed budget. In *In Advances in Neural Information Processing Systems 18*, pages 259–266. MIT Press, 2005.

Ofer Dekel, Philip M. Long, and Yoram Singer. Online learning of multiple tasks with a shared loss. *Journal of Machine Learning Research*, 8:2233–2264, 2007.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. In *COLT*, pages 257–269, 2010.

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.

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

Graham C. Goodwin, Eam K. Teoh, and H. Elliott. Deterministic convergence of a self-tuning regulator with covariance resetting. *Control Theory and App., IEE Proc. D*, 130(1):6 –8, 1983.

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

Elad Hazan and Satyen Kale. Extracting certainty from uncertainty: Regret bounded by variation in costs. In *COLT*, pages 57–68, 2008.

Elad Hazan and Satyen Kale. On stochastic and worst-case models for investing. In *NIPS*, pages 709–717, 2009.

Elad Hazan, Adam Kalai, Satyen Kale, and Amit Agarwal. Logarithmic regret algorithms for online convex optimization. In *COLT*, pages 499–513, 2006.

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

Rudolf E. 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 *NIPS*, pages 785–792, 2001.

Hugh B. McMahan and Matthew J. Streeter. Adaptive bound optimization for online convex optimization. In *COLT*, pages 244–256, 2010.

Francesco Orabona, Nicolò Cesa-Bianchi, and Claudio Gentile. Beyond logarithmic bounds in online learning. In *AISTATS*, 2012.

Alexander Rakhlin and Karthik Sridharan. Online learning with predictable sequences. *CoRR*, abs/1208.3728, 2012.

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.

Eiji Takimoto and Manfred K. Warmuth. The last-step minimax algorithm. In *ALT*, 2000.

Nina Vaits and Koby Crammer. Re-adapting the regularization of weights for non-stationary regression. In *ALT*, 2011.

Volodimir G. 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 *NIPS*, 1997.

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

Bernard Widrow and Marcian E. Hoff. Adaptive switching circuits. 1960.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent, 2003.

כיוון נוסף הוא לבנות אלגוריתמים שלא רגישים לידיעת מידת השינוי בבעיה, או שמסוגלים לזהות אוטומטית את מידת השינוי.

המשקולות בוצעה ע"י הטלתו על קבוצה חסומה [Herbster and Warmuth, 2001], כיווץ באופן כפלי [Kivinen et al., 2001, Cavallanti et al., 2007], או חיסור של דוגמאות קודמות [Cavallanti et al., 2007]. גישה נוספת, שבה השתמשו באלגוריתמים מסדר שני, מבוססת על אתחול מטריצת הקווריאנס, ובצורה זו שוכחים את ההיסטוריה. אלגוריתם ה CR-RLS תוכנן לסינון מסתגל והוא מבצע אתחול של מטריצת הקווריאנס כל מספר קבוע של איטרציות. אלגוריתם דומה ל CR-RLS הוא אלגוריתם ARCOR [Vaits and Crammer, 2011], אבל אלגוריתם זה מבצע אתחול של מטריצת הקווריאנס כל פעם שהערך העצמי של מטריצת הקווריאנס מגיע לסף מסויים, ולא כל מספר קבוע של איטרציות. לכן, אלגוריתם ARCOR מבצע אתחולים בהתבסס על התכונות האמיתיות של הנתונים: הספקטרום של מטריצת הקווריאנס. בנוסף, אלגוריתם ARCOR מבצע הטלה של וקטור המשקולות לקבוצה סגורה כדי למנוע ממנו לגדול יותר מידי בכיוונים מסוימים.

בעבודה זו אנו הולכים בכיוון אחר ומפתחים אלגוריתם לבעיה עם שינוי בהתבסס על עקרון המינימום-מקסימום של Forster [1999]. האלגוריתם שלנו הוא אופטימלי במובן מינימום-מקסימום כאשר יש שינוי. בניגוד לפיתוח של חזאי מינימום-מקסימום עבור מתחרה קבוע ויחיד, בעית האופטימיזציה שמתקבלת במקרה זה אינה פשוטה לפתרון. אנו מפתחים פתרון רקורסיבי לבעיה זו, שמאפשר לחשב את החזאי האופטימלי במובן מינימום-מקסימום. אנו מנתחים את האלגוריתם במסגרת של regret במקרה הגרוע ביותר ומראים, שכל עוד כמות השינוי הממוצע הוא תת-לינארי, השגיאה הממוצעת של האלגוריתם שלנו תתכנס לשגיאה הממוצעת של סדרת הפונקציות הטובה ביותר. בנוסף, כאשר אין שינוי, לאלגוריתם שלנו יש regret לוגריתמי, כמו לאלגוריתם של Forster [1999]. האלגוריתם שלנו הוא פשוט יותר מאלגוריתמים קודמים מכיוון שאינו מבצע אתחולים מפורשים של מטריצת הקווריאנס אלא מבצע "מיצוע" שלה עם מטריצה סקלרית בכל איטרציה.

לסיום, סימולציות סינטטיות מראות את היתרונות של אלגוריתם המינימום-מקסימום שלנו על אלגוריתמים קודמים במקרה הגרוע של שינוי קבוע.

לעבודה זו מספר כיוונים להרחבה אפשרית. כיוון אחד הוא להרחיב את האלגוריתמים לפונקציות שגיאה כלליות ולא רק שגיאה ריבועית. כיוון נוסף להרחבה הוא לבעיות של סיווג במקום רגרסיה או מצב של selective sampling. כמו כן, כדי לממש את האלגוריתמים שלנו יש צורך לבצע היפוך מטריצות; כיוון נוסף להרחבה הוא לבנות גירסה יותר יעילה חישובית של האלגוריתמים.

הגרוע) ביחס לפונקציות לינאריות. Takimoto and Warmuth [2000] השתמשו ברעיון זה לבעית שערוך צפיפות פילוג מקוונת. בעית האופטימיזציה ש Forster [1999] קיבל היתה קמורה גם בבחירת האלגוריתם וגם בבחירת המתחרה, מה שהוביל לבעית אופטימיזציה לא חסומה. Forster [1999] עקף בעיה זו ע"י הנחה של חסם Y על בחירות המתחרה, שצריך להיות ידוע לאלגוריתם. אולם, הניתוח שלו הוא עבור גרסה ללא חסם, שהוביל לregret של $O(\log T)$.

אנו מציעים אלגוריתם מינימום-מקסימום שונה, עם משקולות על דוגמאות, שמבוקרות בצורה כזו כך שהבעיה תהיה קעורה בבחירות המתחרה וקמורה בבחירות האלגוריתם. אנו מנתחים את האלגוריתם ומראים regret לוגריתמי שיכול להיות עם מקדם כפלי טוב יותר מהניתוח של Forster [1999] ושל אלגוריתמים אחרים. אנו מציעים חסם נוסף שהוא לוגריתמי בשגיאה של פונקציית הייחוס, ולא במספר האיטרציות T. התנהגות זו ניתנה לאחרונה ע"י Orabona et al. [2012] עבור אלגוריתם Online Newton Step [Hazan et al., 2006]. אולם, לחסם שלהם יש מקדם כפלי דומה לזה של Forster [1999], ואילו לחסם שלנו יש פוטנציאל למקדם כפלי טוב יותר ויש לו אותה תלות בפונקציית הייחוס כמו של Orabona et al. [2012]. בנוסף, האלגוריתם והניתוח שלנו חופשיים מהנחה של החסם Y או ידיעת ערכו.

תחרות עם הפונקציה הטובה ביותר שהיא יחידה וקבועה זה לא מספיק לכמה בעיות. בהרבה ישומים מעשיים, פונקציית המטרה הנכונה אינה קבועה, אלא נסחפת באופן איטי עם הזמן. למשל נחשוב על פונקציה שמתוכננת לדרג סרטים במערכת המלצה, בהנתן תכונות מסוימות. עם הזמן, הדירוג של סרט יכול להשתנות, מכיוון שיותר ויותר סרטים יוצאים או העונה משתנה.

סיבה זו הובילה לפיתוח של אלגוריתמים למצב של שינוי (Herbster and Warmuth [2001], Kivinen et al. [2001], Cavallanti et al. [2007], Auer and Warmuth [2000]). מטרת האלגוריתם במצב זה היא לשמור על שגיאה ממוצעת קרובה ככל הניתן לזו של סדרת פונקציות הכי טובה שמשתנה לאט, ולא להתחרות היטב עם פונקציה קבועה יחידה. אנו מתמקדים בבעיות שבהן סדרה זו מורכבת רק מפונקציות לינאריות. באופן טיפוסי, סדרה כזו היא נסחפת, כאשר כל פונקציה דומה לקודמתה, או "מוזזת", כאשר עקרונית הסדרה יכולה להתחלק למספר סגמנטים, כך שעבור כל סגמנט ישנה פונקציה אחת בודדת בעלת ביצועים טובים על כל הדוגמאות באותו סגמנט.

גישה אחת שבה השתמשו באלגוריתמים קודמים לסביבת עבודה זו מבוססת על gradient-descent עם בקרה נוספת על הנורמה של וקטור המשקולות שמשמש לחיזוי. חסימת וקטור

תקציר

אנו מתייחסים לבעיה של למידה מקוונת לרגרסיה, שבה האלגוריתם הלומד מנסה לחזות מספרים ממשיים בהנתן קלט מסוים, בסדרה של איטרציות. דוגמאות לישומים מעשיים לבעיה זו הם חיזוי של מזג אויר או מניות. מטרת האלגוריתם היא לשמור על הפרש קטן בין החיזוי שלו לתוצאה האמיתית, שיכולה להבחר ע"י מתחרה. הפרש זה נמדד ע"י פונקציית שגיאה, כגון שגיאה ריבועית. מקובל להעריך אלגוריתמים ע"י גודל שנקרא regret, שהוא ההפרש בין השגיאה המצטברת של האלגוריתם לבין השגיאה המצטברת של פונקציה כלשהי ממחלקה כלשהי.

בחצי המאה האחרונה הרבה אלגוריתמים הוצעו לבעיה זו. חלק מהאלגוריתמים מסוגלים להשיג שגיאה ממוצעת קרובה כרצונינו לשגיאה של הפונקציה הכי טובה בדיעבד. בנוסף, הבטחות אלה תקפות גם כאשר זוג הקלט והפלט נבחר ע"י יריב ללא הנחות על פילוגים. אלגוריתמים שמבוססים על Gradient-descent נותחו ע"י Cesa-Bianchi et al. [1993], והם הראו שניתן לקבל regret של $O\left(\sqrt{T}\right)$ כאשר T הוא מספר האיטרציות. Kivinen and Warmuth [1997] הציעו אלגוריתם דומה אבל שמעדכן את וקטור המשקולות בצורה כפלית. אלגוריתם זה משיג גם regret של $O\left(\sqrt{T}\right)$ אבל יש לו שגיאה קטנה יותר אם רק חלק מרכיבי וקטור הכניסה הם רלוונטיים לחיזוי. בהקשר של תכנות קמור מקוון התקבל regret דומה [Zinkevich, 2003].

בעבודה זו אנו מתמקדים באלגוריתמים מסדר שני. אלגוריתמים אלה מחזיקים וקטור משקולות שמשמש לחיזוי ובנוסף מטריצת קווריאנס שמתפקדת כמו קצב לימוד מסתגל. אלגוריתם ה-RLS [Hayes, 1996], אלגוריתם ridge-regression [Foster, 1991] ואלגוריתם ה-AAR [Vovk, 1997, 2001] הם דוגמאות של אלגוריתמים מסדר שני, כולם משיגים regret של $O\left(\log T\right)$, אולם עם מקדמים כפליים שונים של הלוגריתם. אלגוריתם דומה עם regret דומה (AROWR) הוצע ע"י Vaits and Crammer [2011] ונותח ע"י Crammer et al. [2012]. לאחרונה, Orabona et al. [2012] הראה שניתן להשיג regret תת-לוגריתמי כאשר השגיאה המצטברת של המודל האופטימלי היא תת-לינארית במספר האיטרציות T.

Forster [1999] הציע את אלגוריתם המינימום-מקסימום לרגרסיה מקוונת אשר מבצע חיזוי בהנחה שזו הדוגמה האחרונה, והמטרה של האלגוריתם היא למזער את ה-regret המקסימלי (במקרה

פרסומים

חלקים מעבודה זו פורסמו במאמרים

Weighted Last-Step Min-Max Algorithm with Improved Sub-Logarithmic Regret.
Edward Moroshko and Koby Crammer. 2012. In ALT.

A Last-Step Regression Algorithm for Non-Stationary Online Learning.
Edward Moroshko and Koby Crammer. 2013. In AISTATS.

המחקר נעשה בהנחיית פרופ' קובי קרמר בפקולטה להנדסת חשמל.

אני מודה לקובי על ההדרכה והליווי לאורך כל הדרך. בנוסף אני מודה לקובי על המימון של הנסיעות שלי לכנסים ALT 2012 ו AISTATS 2013.
תודה למשפחתי על התמיכה.
תודה מיוחדת לארוסתי אפרת על אהבתה, תמיכתה וסבלנותה הרבה לאורך כל הדרך.

אלגוריתמי מינימום-מקסימום ואנליזה חדשים לרגרסיה מקוונת

חיבור על מחקר

לשם מילוי חלקי של הדרישות לקבלת התואר
מגיסטר למדעים בהנדסת חשמל

אדוארד מורושקו

הוגש לסנט הטכניון – מכון טכנולוגי לישראל
תמוז תשע"ג חיפה יוני 2013

אלגוריתמי מינימום-מקסימום ואנליזה חדשים לרגרסיה מקוונת

אדוארד מורושקו