A Last-Step Regression Algorithm for Non-Stationary Online Learning

Edward Moroshko
Department of Electrical Engineering,
The Technion, Haifa, Israel

Koby Crammer
Department of Electrical Engineering,
The Technion, Haifa, Israel

Abstract

The goal of a learner in standard online learning is to maintain an average loss close to the loss of the best-performing *single* function in some class. In many real-world problems, such as rating or ranking items, there is no single best target function during the runtime of the algorithm, instead the best (local) target function is drifting over time. We develop a novel last-step min-max optimal algorithm 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. In some situations, our bound improves over existing bounds, and additionally the algorithm suffers logarithmic regret when there is no drift. We also build on the H_∞ filter and its bound, and develop and analyze a second algorithm for drifting setting. Synthetic simulations demonstrate the advantages of our algorithms in a worst-case constant drift setting.

1 Introduction

We consider the on-line learning problems, in which a learning algorithm predicts real numbers given inputs in a sequence of trials. An example of such a problem is to predict a stock's prices given input about the current state of the stock-market. In general, the goal of the algorithm is to achieve an average loss that is not much larger compared to the loss one suffers if it had always chosen to predict according to the best-performing single function from some class of functions.

In the past half a century, many algorithms were proposed (a review can be found in a comprehensive book on the topic [10]) for this problem, some of which are able to achieve an average loss arbitrarily close to that of the best function in retrospect. Furthermore, such guarantees hold even if the input and output pairs are chosen in a fully adversarial manner with no distributional assumptions.

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. 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.

With such properties in mind, we develop new learning algorithms designed to work with target drift. The goal of an algorithm is to maintain an average loss close to that of the best slowly changing sequence of functions, rather than compete well with a single function. We focus on problems for which this sequence consists only of linear functions. Some previous algorithms [27, 1, 22, 25] designed for this problem are based on gradient descent, with additional control on the norm (or Bregman divergence) of the weight-vector used for prediction [25], or the number of inputs used to define it [7].

We take a different route and derive an algorithm based on the last-step min-max approach proposed by Forster [17] and later used [34] for online density estimation. On each iteration the algorithm makes the optimal min-max prediction with respect to a quantity called regret, assuming it is the last iteration. Yet, unlike previous work, it is optimal when a drift is allowed. As opposed to the derivation of the last-step min-max predictor for a fixed vector, the resulting optimization problem is not straightforward to solve. We develop a dynamic program (a recursion) to solve this problem, which allows to compute the optimal last-step min-max predictor. We 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. Specifically, we show that if the total amount of drift is $T\nu$ (for $\nu = o(1)$) the cumulative regret is bounded by $T\nu^{1/3} + \log(T)$. When the in-

Appearing in Proceedings of the 16^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2013, Scottsdale, AZ, USA. Volume 31 of JMLR: W&CP 31.

stantaneous drift is close to constant, this improves over a previous bound of Vaits and Crammer [35] of an algorithm named ARCOR that showed a bound of $T\nu^{1/4}\log(T)$. Additionally, when no drift is introduced (stationary setting) our algorithm suffers logarithmic regret, as for the algorithm of Forster [17]. We also build on the H_∞ adaptive filter, which is min-max optimal with respect to a *filtering task*, and derive another learning algorithm based on the same min-max principle. We provide a regret bound for this algorithm as well, and relate the two algorithms and their respective bounds. Finally, synthetic simulations show the advantages of our algorithms when a close to constant drift is allowed.

2 Problem Setting

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

On each round, the performance of the algorithm is evaluated using the squared loss, $\ell_t(\text{alg}) = \ell\left(y_t, \hat{y}_t\right) = \left(\hat{y}_t - y_t\right)^2$. The cumulative loss suffered over T iterations is, $L_T(\text{alg}) = \sum_{t=1}^{T} \ell_t(\text{alg})$. The goal of the algorithm is to have low cumulative loss compared to predictors from some class. A large body of work is focused on linear prediction functions of the form $f(\mathbf{x}) = \mathbf{x}^\top \mathbf{u}$ where $\mathbf{u} \in \mathbb{R}^d$ is some weight-vector. We denote by $\ell_t(\mathbf{u}) = \left(\mathbf{x}_t^\top \mathbf{u} - y_t\right)^2$ the instantaneous loss of a weight-vector $\mathbf{u}$.

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 $L_T(\{\mathbf{u}_t\}) = \sum_t^T \ell_t(\mathbf{u}_t)$ and the regret of an algorithm by $R_T(\{\mathbf{u}_t\}) = \sum_t^T (y_t - \hat{y}_t)^2 - L_T(\{\mathbf{u}_t\})$. The goal of the 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 the 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 incorporate the total drift of the weight-vectors defined to be,

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

where ν is the *average drift* . Below we bound the regret with, $R_T(\{\mathbf{u}_t\}) \leq \mathcal{O}\left(T^{\frac{2}{3}}V^{\frac{1}{3}} + \log(T)\right) = \mathcal{O}\left(T\nu^{\frac{1}{3}} + \log(T)\right)$. Next, we develop an explicit form of the last-step min-max algorithm with drift.

3 Algorithm

We define the last-step minmax predictor $\hat{y}_T$ to be[1],

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

where we define

$$Q_t\left(\mathbf{u}_1,\ldots,\mathbf{u}_t\right) = b\left\|\mathbf{u}_1\right\|^2 + c\sum_{s=1}^{t-1}\left\|\mathbf{u}_{s+1} - \mathbf{u}_s\right\|^2 + \sum_{s=1}^{t}\left(y_s - \mathbf{u}_s^\top \mathbf{x}_s\right)^2 \ , \quad (3)$$

for some positive constants b, c. The last optimization problem can also be seen as a game where the algorithm chooses a prediction $\hat{y}_t$ to minimize the last-step regret, while an adversary chooses a target label y_t to maximize it. The first term of (2) is the loss suffered by the algorithm while $Q_t\left(\mathbf{u}_1,\ldots,\mathbf{u}_t\right)$ defined in (3) is a sum of the loss suffered by some sequence of linear functions $(\mathbf{u}_1,\ldots,\mathbf{u}_t)$, a penalty for consecutive pairs that are far from each other, and for the norm of the first to be far from zero.

We first solve recursively the inner optimization problem $\min_{\mathbf{u}_1,\ldots,\mathbf{u}_t} Q_t\left(\mathbf{u}_1,\ldots,\mathbf{u}_t\right)$, for which we define an auxiliary function,

$$P_t\left(\mathbf{u}_t\right) = \min_{\mathbf{u}_1,\ldots,\mathbf{u}_{t-1}} Q_t\left(\mathbf{u}_1,\ldots,\mathbf{u}_t\right) \ , \quad (4)$$

which clearly satisfies,

$$\min_{\mathbf{u}_1,\ldots,\mathbf{u}_t} Q_t\left(\mathbf{u}_1,\ldots,\mathbf{u}_t\right) = \min_{\mathbf{u}_t} P_t(\mathbf{u}_t) \ . \quad (5)$$

We start the derivation of the algorithm with a lemma, stating a recursive form of the function-sequence $P_t(\mathbf{u}_t)$.

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

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

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

The proof appears in App. B.1. Using Lem. 1 we write explicitly the function $P_t(\mathbf{u}_t)$.

Lemma 2. *The following equality holds*

$$P_t\left(\mathbf{u}_t\right) = \mathbf{u}_t^\top \mathbf{D}_t \mathbf{u}_t - 2\mathbf{u}_t^\top \mathbf{e}_t + f_t \ , \tag{6}$$

where,

$$\mathbf{D}_1 = b\mathbf{I} + \mathbf{x}_1\mathbf{x}_1^\top \ , \ \mathbf{D}_t = \left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)^{-1} + \mathbf{x}_t\mathbf{x}_t^\top \tag{7}$$

$$\mathbf{e}_1 = y_1\mathbf{x}_1 \ , \ \mathbf{e}_t = \left(\mathbf{I} + c^{-1}\mathbf{D}_{t-1}\right)^{-1}\mathbf{e}_{t-1} + y_t\mathbf{x}_t \tag{8}$$

$$f_1 = y_1^2 \ , \ f_t = f_{t-1} - \mathbf{e}_{t-1}^\top \left(c\mathbf{I} + \mathbf{D}_{t-1}\right)^{-1}\mathbf{e}_{t-1} + y_t^2 \ . \tag{9}$$

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

The proof appears in App. B.2. From Lem. 2 we conclude, by substituting (6) in (5), that,

$$\begin{aligned}&\min_{\mathbf{u}_1,\ldots,\mathbf{u}_t} Q_t\left(\mathbf{u}_1,\ldots,\mathbf{u}_t\right) \\ &= \min_{\mathbf{u}_t}\left(\mathbf{u}_t^\top\mathbf{D}_t\mathbf{u}_t - 2\mathbf{u}_t^\top\mathbf{e}_t + f_t\right) = -\mathbf{e}_t^\top\mathbf{D}_t^{-1}\mathbf{e}_t + f_t \ .\end{aligned} \tag{10}$$

Substituting (10) back in (2) we get that the last-step min-max predictor $\hat{y}_T$ is given by,

$$\arg\min_{\hat{y}_T}\max_{y_T}\left[\sum_{t=1}^{T}\left(y_t - \hat{y}_t\right)^2 + \mathbf{e}_T^\top\mathbf{D}_T^{-1}\mathbf{e}_T - f_T\right] \ . \tag{11}$$

Since $\mathbf{e}_T$ depends on y_T we substitute (8) in the second term of (11),

$$\begin{aligned}&\mathbf{e}_T^\top\mathbf{D}_T^{-1}\mathbf{e}_T = \\ &\left(\left(\mathbf{I} + c^{-1}\mathbf{D}_{T-1}\right)^{-1}\mathbf{e}_{T-1} + y_T\mathbf{x}_T\right)^\top\mathbf{D}_T^{-1} \\ &\left(\left(\mathbf{I} + c^{-1}\mathbf{D}_{T-1}\right)^{-1}\mathbf{e}_{T-1} + y_T\mathbf{x}_T\right) \ .\end{aligned} \tag{12}$$

Substituting (12) and (9) in (11) and omitting terms not depending explicitly on y_T and $\hat{y}_T$ we get,

$$\begin{aligned}\hat{y}_T &= \arg\min_{\hat{y}_T}\max_{y_T}\Bigg[\left(y_T - \hat{y}_T\right)^2 + y_T^2\mathbf{x}_T^\top\mathbf{D}_T^{-1}\mathbf{x}_T \\ &\quad + 2y_T\mathbf{x}_T^\top\mathbf{D}_T^{-1}\left(\mathbf{I} + c^{-1}\mathbf{D}_{T-1}\right)^{-1}\mathbf{e}_{T-1} - y_T^2\Bigg] \\ &= \arg\min_{\hat{y}_T}\max_{y_T}\Bigg[\left(\mathbf{x}_T^\top\mathbf{D}_T^{-1}\mathbf{x}_T\right)y_T^2 + \hat{y}_T^2 \\ &\quad + 2y_T\left(\mathbf{x}_T^\top\mathbf{D}_T^{-1}\left(\mathbf{I} + c^{-1}\mathbf{D}_{T-1}\right)^{-1}\mathbf{e}_{T-1} - \hat{y}_T\right)\Bigg] \ .\end{aligned} \tag{13}$$

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 in a different context [17]. 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 (13) over y_T is given by the following equation, since the optimal value is $y_T \in \{+Y, -Y\}$,

$$\begin{aligned}\hat{y}_T = \arg\min_{\hat{y}_T}\Bigg[&\left(\mathbf{x}_T^\top\mathbf{D}_T^{-1}\mathbf{x}_T\right)Y^2 + \hat{y}_T^2 \\ &+ 2Y\left|\mathbf{x}_T^\top\mathbf{D}_T^{-1}\left(\mathbf{I} + c^{-1}\mathbf{D}_{T-1}\right)^{-1}\mathbf{e}_{T-1} - \hat{y}_T\right|\Bigg] \ .\end{aligned}$$

This problem is of a similar form to the one discussed by Forster [17], from which we get the optimal solution, $\hat{y}_T = clip\left(\mathbf{x}_T^\top\mathbf{D}_T^{-1}\left(\mathbf{I} + c^{-1}\mathbf{D}_{T-1}\right)^{-1}\mathbf{e}_{T-1}, Y\right)$, where for $y > 0$ we define $clip(x, y) = \text{sign}(x)\min\{|x|, y\}$. 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,

$$\hat{y}_T = \mathbf{x}_T^\top\mathbf{D}_T^{-1}\left(\mathbf{I} + c^{-1}\mathbf{D}_{T-1}\right)^{-1}\mathbf{e}_{T-1} \ . \tag{14}$$

We call the algorithm LASER for last step adaptive regressor algorithm, and it is summarized in Fig. 1. Clearly, for $c = \infty$ the LASER algorithm reduces to the AAR algorithm of Vovk [36], or the last-step min-max algorithm of Forster [17]. See also the work of Azoury and Warmuth [2]. The algorithm can be combined with Mercer kernels as it employs only sums of inner- and outer-products of its inputs. This algorithm can be seen also as a forward algorithm [2]: The predictor of (14) 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 (8) 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 (10) 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 (14).

4 Analysis

We now analyze the performance of the algorithm in the worst-case setting, starting with the following technical lemma.

Lemma 3. *For all t the following statement holds,*

$$\begin{aligned}&\mathbf{D'}_{t-1}\mathbf{D}_t^{-1}\mathbf{x}_t\mathbf{x}_t^\top\mathbf{D}_t^{-1}\mathbf{D'}_{t-1} - \mathbf{D}_{t-1}^{-1} \\ &+ \mathbf{D'}_{t-1}\left(\mathbf{D}_t^{-1}\mathbf{D'}_{t-1} + c^{-1}\mathbf{I}\right) \preceq 0\end{aligned}$$

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

The proof appears in App. B.3. We next bound the cumulative loss of the algorithm,

Theorem 4. *Assume the labels are bounded $\sup_t |y_t| \le Y$ for some $Y \in \mathbb{R}$. Then the following bound holds,*

Parameters: $0 < b < c$
Initialize: Set $\mathbf{D}_0 = (bc)/(c-b)\,\mathbf{I} \in \mathbb{R}^{d\times d}$ and $\mathbf{e}_0 = \mathbf{0} \in \mathbb{R}^d$
For $t = 1, \ldots, T$ do
- Receive an instance $\mathbf{x}_t$
- Compute $\mathbf{D}_t = \left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)^{-1} + \mathbf{x}_t\mathbf{x}_t^\top$ (7)
- Output prediction $\hat{y}_t = \mathbf{x}_t^\top \mathbf{D}_t^{-1}\left(\mathbf{I} + c^{-1}\mathbf{D}_{t-1}\right)^{-1}\mathbf{e}_{t-1}$
- Receive the correct label y_t
- Update: $\mathbf{e}_t = \left(\mathbf{I} + c^{-1}\mathbf{D}_{t-1}\right)^{-1}\mathbf{e}_{t-1} + y_t\mathbf{x}_t$ (8)

Output: $\mathbf{e}_T$, $\mathbf{D}_T$

Figure 1: LASER: last step adaptive regression algorithm.

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

Proof. Fix t. A long algebraic manipulation yields,

$$\begin{aligned}
&\left(y_t - \hat{y}_t\right)^2 + \min_{\mathbf{u}_1,\ldots,\mathbf{u}_{t-1}} Q_{t-1}\left(\mathbf{u}_1,\ldots,\mathbf{u}_{t-1}\right) \\
&- \min_{\mathbf{u}_1,\ldots,\mathbf{u}_t} Q_t\left(\mathbf{u}_1,\ldots,\mathbf{u}_t\right) \\
&= \left(y_t - \hat{y}_t\right)^2 + 2y_t\mathbf{x}_t^\top\mathbf{D}_t^{-1}\mathbf{D'}_{t-1}\mathbf{e}_{t-1} \\
&\quad + \mathbf{e}_{t-1}^\top\Bigg[-\mathbf{D}_{t-1}^{-1} + \mathbf{D'}_{t-1}\left(\mathbf{D}_t^{-1}\mathbf{D'}_{t-1} + c^{-1}\mathbf{I}\right)\Bigg]\mathbf{e}_{t-1} \\
&\quad + y_t^2\mathbf{x}_t^\top\mathbf{D}_t^{-1}\mathbf{x}_t - y_t^2\ . \qquad (15)
\end{aligned}$$

Substituting the specific value of the predictor $\hat{y}_t = \mathbf{x}_t^\top\mathbf{D}_t^{-1}\mathbf{D'}_{t-1}\mathbf{e}_{t-1}$ from (14), we get that (15) equals to,

$$\begin{aligned}
&\hat{y}_t^2 + y_t^2\mathbf{x}_t^\top\mathbf{D}_t^{-1}\mathbf{x}_t + \mathbf{e}_{t-1}^\top\Bigg[-\mathbf{D}_{t-1}^{-1} \\
&\quad + \mathbf{D'}_{t-1}\left(\mathbf{D}_t^{-1}\mathbf{D'}_{t-1} + c^{-1}\mathbf{I}\right)\Bigg]\mathbf{e}_{t-1} \\
&= \mathbf{e}_{t-1}^\top\mathbf{D'}_{t-1}\mathbf{D}_t^{-1}\mathbf{x}_t\mathbf{x}_t^\top\mathbf{D}_t^{-1}\mathbf{D'}_{t-1}\mathbf{e}_{t-1} + \mathbf{e}_{t-1}^\top\Bigg[-\mathbf{D}_{t-1}^{-1} \\
&\quad + \mathbf{D'}_{t-1}\left(\mathbf{D}_t^{-1}\mathbf{D'}_{t-1} + c^{-1}\mathbf{I}\right)\Bigg]\mathbf{e}_{t-1} + y_t^2\mathbf{x}_t^\top\mathbf{D}_t^{-1}\mathbf{x}_t \\
&= \mathbf{e}_{t-1}^\top\Bigg[\mathbf{D'}_{t-1}\mathbf{D}_t^{-1}\mathbf{x}_t\mathbf{x}_t^\top\mathbf{D}_t^{-1}\mathbf{D'}_{t-1} - \mathbf{D}_{t-1}^{-1} \qquad (16) \\
&\quad + \mathbf{D'}_{t-1}\left(\mathbf{D}_t^{-1}\mathbf{D'}_{t-1} + c^{-1}\mathbf{I}\right)\Bigg]\mathbf{e}_{t-1} + y_t^2\mathbf{x}_t^\top\mathbf{D}_t^{-1}\mathbf{x}_t\ .
\end{aligned}$$

Parameters: $1 < a$, $0 < b, c$
Initialize: Set $\mathbf{P}_0 = b^{-1}\mathbf{I} \in \mathbb{R}^{d\times d}$ and $\mathbf{w}_0 = \mathbf{0} \in \mathbb{R}^d$
For $t = 1, \ldots, T$ do
- Receive an instance $\mathbf{x}_t$
- Output prediction $\hat{y}_t = \mathbf{x}_t^\top\mathbf{w}_{t-1}$
- Receive the correct label y_t
- Compute $\tilde{\mathbf{P}}_t = \left(\mathbf{P}_{t-1}^{-1} + (a-1)\mathbf{x}_t\mathbf{x}_t^\top\right)^{-1}$
- Update $\mathbf{w}_t = \mathbf{w}_{t-1} + a\tilde{\mathbf{P}}_t(y_t - \hat{y}_t)\mathbf{x}_t$
- Update $\mathbf{P}_t = \tilde{\mathbf{P}}_t + c^{-1}\mathbf{I}$

Output: $\mathbf{w}_T$, $\mathbf{P}_T$

Figure 2: An H_∞ algorithm for online regression.

Using Lem. 3 we upper bound (16) with, $y_t^2\mathbf{x}_t^\top\mathbf{D}_t^{-1}\mathbf{x}_t \le Y^2\mathbf{x}_t^\top\mathbf{D}_t^{-1}\mathbf{x}_t$. Finally, summing over $t \in \{1,\ldots,T\}$ gives the desired bound,

$$\begin{aligned}
&\sum_{t=1}^{T}\left(y_t - \hat{y}_t\right)^2 - \min_{\mathbf{u}_1,\ldots,\mathbf{u}_T}\Bigg[b\left\|\mathbf{u}_1\right\|^2 \\
&\quad + c\sum_{t=1}^{T-1}\left\|\mathbf{u}_{t+1} - \mathbf{u}_t\right\|^2 + \sum_{t=1}^{T}\left(y_t - \mathbf{u}_t^\top\mathbf{x}_t\right)^2\Bigg] \\
&= L_T(\text{LASER}) - \min_{\mathbf{u}_1,\ldots,\mathbf{u}_T}\Bigg[b\left\|\mathbf{u}_1\right\|^2 + cV_T(\{\mathbf{u}_t\}) \\
&\quad + L_T(\{\mathbf{u}_t\})\Bigg] \le Y^2\sum_{t=1}^{T}\mathbf{x}_t^\top\mathbf{D}_t^{-1}\mathbf{x}_t\ .
\end{aligned}$$

□

In the next lemma we further bound the right term of Thm. 4. This type of bound is based on the usage of the covariance-like matrix $\mathbf{D}$.

Lemma 5.

$$\sum_{t=1}^{T}\mathbf{x}_t^\top\mathbf{D}_t^{-1}\mathbf{x}_t \le \ln\left|\frac{1}{b}\mathbf{D}_T\right| + c^{-1}\sum_{t=1}^{T}\mathrm{Tr}\left(\mathbf{D}_{t-1}\right)\ . \quad (17)$$

Proof. Similar to the derivation of Forster [17] (details omitted due to lack of space),

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

and because $\ln\left|\frac{1}{b}\mathbf{D}_0\right| \ge 0$ we get $\sum_{t=1}^{T}\mathbf{x}_t^\top\mathbf{D}_t^{-1}\mathbf{x}_t \le \ln\left|\frac{1}{b}\mathbf{D}_T\right| + \sum_{t=1}^{T}\ln\left|\left(\mathbf{I} + c^{-1}\mathbf{D}_{t-1}\right)\right| \le \ln\left|\frac{1}{b}\mathbf{D}_T\right| + c^{-1}\sum_{t=1}^{T}\mathrm{Tr}\left(\mathbf{D}_{t-1}\right)$. □

At first sight it seems that the right term of (17) may grow super-linearly with T, as each of the matrices $\mathbf{D}_t$ grows with t. The next two lemmas show that this is not the case, and in fact, the right term of (17) is not growing too fast, which will allow us to obtain a sub-linear regret bound. Lem. 6 analyzes the properties of the recursion of $\mathbf{D}$ defined in (7) for scalars, that is $d = 1$. In Lem. 7 we extend this analysis to matrices.

Lemma 6. *Define $f(\lambda) = \lambda\beta/(\lambda+\beta) + x^2$ 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)** $f(\lambda) \leq \max\left\{\lambda, \frac{3\gamma^2+\sqrt{\gamma^4+4\gamma^2\beta}}{2}\right\}$.

The proof appears in App. B.6. We build on Lem. 6 to bound the maximal eigenvalue of the matrices $\mathbf{D}_t$.

Lemma 7. *Assume $\|\mathbf{x}_t\|^2 \leq X^2$ for some X. Then, the eigenvalues of $\mathbf{D}_t$ (for $t \geq 1$), denoted by $\lambda_i(\mathbf{D}_t)$, are upper bounded by $\max_i \lambda_i(\mathbf{D}_t) \leq \max\left\{\frac{3X^2+\sqrt{X^4+4X^2c}}{2}, b + X^2\right\}$.*

Proof. By induction. From (7) we have that $\lambda_i(\mathbf{D}_1) \leq b + X^2$ for $i = 1, \ldots, d$. We proceed with a proof for some t. For simplicity, denote by $\lambda_i = \lambda_i(\mathbf{D}_{t-1})$ the ith eigenvalue of $\mathbf{D}_{t-1}$ with a corresponding eigenvector $\boldsymbol{v}_i$. From (7) we have,

$$\begin{aligned}\mathbf{D}_t &= \left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)^{-1} + \mathbf{x}_t\mathbf{x}_t^\top \\ &\preceq \left(\mathbf{D}_{t-1}^{-1} + c^{-1}\mathbf{I}\right)^{-1} + \mathbf{I}\|\mathbf{x}_t\|^2 \\ &= \sum_i^d \boldsymbol{v}_i\boldsymbol{v}_i^\top\left(\frac{\lambda_i c}{\lambda_i + c} + \|\mathbf{x}_t\|^2\right). \end{aligned} \tag{18}$$

Plugging Lem. 6 in (18) we get, $\mathbf{D}_t \preceq \sum_i^d \boldsymbol{v}_i\boldsymbol{v}_i^\top \max\left\{\frac{3X^2+\sqrt{X^4+4X^2c}}{2}, b + X^2\right\} = \max\left\{\frac{3X^2+\sqrt{X^4+4X^2c}}{2}, b + X^2\right\}\mathbf{I}$. □

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

Corollary 8. *Assume $\|\mathbf{x}_t\|^2 \leq X^2$, $|y_t| \leq Y$. Then,*

$$\begin{aligned}L_T(\textit{LASER}) \leq b\|\mathbf{u}_1\|^2 + L_T(\{\mathbf{u}_t\}) + Y^2\ln\left|\frac{1}{b}\mathbf{D}_T\right| \\ + c^{-1}Y^2\mathrm{Tr}(\mathbf{D}_0) + cV \\ + c^{-1}Y^2Td\max\left\{\frac{3X^2+\sqrt{X^4+4X^2c}}{2}, b + X^2\right\}.\end{aligned} \tag{19}$$

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

$$c = \left(\sqrt{2}TY^2dX/V\right)^{2/3} \tag{20}$$

we have,

$$\begin{aligned}L_T(\textit{LASER}) \leq \\ b\|\mathbf{u}_1\|^2 + 3\left(\sqrt{2}Y^2dX\right)^{2/3}T^{2/3}V^{1/3} \\ + \frac{\varepsilon}{1-\varepsilon}Y^2d + L_T(\{\mathbf{u}_t\}) + Y^2\ln\left|\frac{1}{b}\mathbf{D}_T\right|.\end{aligned} \tag{21}$$

The proof appears in Sec. A.1. A few remarks are in order. First, when the total drift $V = 0$ goes to zero, we set $c = \infty$ and thus we have $\mathbf{D}_t = b\mathbf{I} + \sum_{s=1}^t \mathbf{x}_s\mathbf{x}_s^\top$ used in recent algorithms [36, 17, 21, 9]. In this case the algorithm reduces to the algorithm by Forster [17] (which is also the Aggregating Algorithm for Regression of Vovk [36]), with the same logarithmic regret bound (note that the last term of (21) is logarithmic in T, see the proof of Forster [17]). See also the work of Azoury and Warmuth [2]. Second, substituting $V = T\nu$ we get that the bound depends on the average drift as $T^{2/3}(T\nu)^{1/3} = T\nu^{1/3}$. Clearly, to have a sublinear regret we must have $\nu = o(1)$. Third, Vaits and Crammer [35] recently proposed an algorithm, called ARCOR, for the same setting. The regret of ARCOR depends on the total drift as $\sqrt{TV'}\log(T)$, where their definition of total drift is a sum of the Euclidean differences $V' = \sum_t^{T-1}\|\mathbf{u}_{t+1} - \mathbf{u}_t\|$, rather than the squared norm. When the instantaneous drift $\|\mathbf{u}_{t+1} - \mathbf{u}_t\|$ is constant, this notion of total drift is related to our average drift, $V' = T\sqrt{\nu}$. Therefore, in this case the bound of ARCOR [35] is $\nu^{1/4}T\log(T)$ which is worse than our bound, both since it has an additional $\log(T)$ factor (as opposed to our additive log term) and since $\nu = o(1)$. Therefore we expect that our algorithm will perform better than ARCOR [35] when the instantaneous drift is approximately constant. Indeed, the synthetic simulations described in Sec. 6 further support this conclusion. Fourth, Herbster and Warmuth [22] 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. Yet, their bounds have better dependency on d, the dimension of the inputs x. Busuttil and Kalnishkan [6] developed a variant of the Aggregating Algorithm [20] 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 $V = o(T)$. Fifth, if $V \geq T\frac{Y^2dM}{\mu^2}$ then by setting $c = \sqrt{Y^2dMT/V}$ we have,

$$\begin{aligned}L_T(\text{LASER}) \leq b\|\mathbf{u}_1\|^2 + 2\sqrt{Y^2dTMV} \\ + \frac{\varepsilon}{1-\varepsilon}Y^2d + L_T(\{\mathbf{u}_t\}) + Y^2\ln\left|\frac{1}{b}\mathbf{D}_T\right|\end{aligned} \tag{22}$$

(See App. B.5 for details). The last bound is linear in T and can be obtained also by a naive algorithm that outputs $\hat{y}_t = 0$ for all t.

5 An H_∞ Algorithm for Online Regression

Adaptive filtering is an active and well established area of research in signal processing. Formally, it is equivalent to online learning. On each iteration t the filter receives an input $\mathbf{x}_t \in \mathbb{R}^d$ and predicts a corresponding output $\hat{y}_t$. It then receives the true desired output y_t and updates its internal model. Many adaptive filtering algorithms employ linear models, that is, at time t they output $\hat{y}_t = \boldsymbol{w}_t^\top \mathbf{x}_t$. For example, a well known online learning algorithm [37] for regression, which is basically a gradient-descent algorithm with the squared-loss, is known as the *least mean-square (LMS)* algorithm in the adaptive filtering literature [31].

One possible difference between adaptive filtering and online learning can be viewed in the interpretation of algorithms, and as a consequence, of their analysis. In online learning, the goal of an algorithm is to make *predictions* $\hat{y}_t$, and the predictions are compared to the predictions of some function from a known class (e.g. linear, parameteized by $\mathbf{u}$). Thus, a typical online performance bound relates the quality of the algorithm's predictions with the quality of some function's $g(\mathbf{x}) = \mathbf{u}^\top \mathbf{x}$ predictions, using some non-negative loss measure $\ell(\boldsymbol{w}_t^\top \mathbf{x}_t, y_t)$. Such bounds often have the following shape,

$$\overbrace{\sum_t \ell(\boldsymbol{w}_t^\top \mathbf{x}_t, y_t)}^{\text{algorithm loss with respect to observation}} \leq A \overbrace{\sum_t \ell(\mathbf{u}^\top \mathbf{x}_t, y_t)}^{\text{function } \mathbf{u} \text{ loss}} + B,$$

for some multiplicative-factor A and an additive factor B.

Adaptive filtering is similar to the realizable setting in machine learning, where it is assumed the existence of some filter and the goal is to recover it using *noisy* observations. Often it is assumed that the output is a corrupted version of the output of some function, $y = f(\mathbf{x}) + n$, with some noise n. Thus a typical bound relates the quality of an algorithm's predictions *with respect to the target filter* $\mathbf{u}$ and the amount of noise in the problem,

$$\overbrace{\sum_t \ell(\boldsymbol{w}_t^\top \mathbf{x}_t, \mathbf{u}^\top \mathbf{x}_t)}^{\text{algorithm loss with respect to a reference}} \leq A \overbrace{\sum_t \ell(\mathbf{u}^\top \mathbf{x}_t, y_t)}^{\text{amount of noise}} + B \ .$$

The H_∞ filters (see e.g. papers by Simon [33, 32]) are a family of (robust) linear filters developed based on a min-max approach, like LASER, and analyzed in the worst case setting. These filters are reminiscent of the celebrated Kalman filter [23], which was motivated and analyzed in a stochastic setting with Gaussian noise. A pseudocode of one such filter we *modified* to online linear regression appears in Fig. 2. Theory of H_∞ filters states [33, Section 11.3] the following bound on its performance as a filter.

Theorem 9. *Assume the filter is executed with parameters $a > 1$ and $b, c > 0$. Then, for all input-output pairs $(\mathbf{x}_t, y_t)$ and for all reference vectors $\mathbf{u}_t$ the following bound holds on the filter's performance, $\sum_{t=1}^T \left(\mathbf{x}_t^\top \mathbf{w}_t - \mathbf{x}_t^\top \mathbf{u}_t\right)^2 \leq a L_T(\{\mathbf{u}_t\}) + b \left\|\mathbf{u}_1\right\|^2 + c V_T\left(\{\mathbf{u}_t\}\right)$.*

From the theorem we establish a regret bound for the H_∞ algorithm to online learning.

Corollary 10. *Fix $\alpha > 0$. The total squared-loss suffered by the algorithm is bounded by*

$$\begin{aligned} L_T(H_\infty) \quad \leq \quad & \left(1 + 1/\alpha + (1+\alpha)\, a\right) L_T(\{\mathbf{u}_t\}) \\ & + (1+\alpha)\, b \left\|\mathbf{u}_1\right\|^2 + (1+\alpha)\, c V_T\left(\{\mathbf{u}_t\}\right) \ . \end{aligned} \tag{23}$$

Proof. Using a bound of Hassibi and Kailath [4, Lemma 4] we have that for all $\alpha > 0$, $\left(y_t - \mathbf{x}_t^\top \mathbf{w}_t\right)^2 \leq \left(1 + \frac{1}{\alpha}\right)\left(y_t - \mathbf{x}_t^\top \mathbf{u}_t\right)^2 + (1+\alpha)\left[\mathbf{x}_t^\top \left(\mathbf{w}_t - \mathbf{u}_t\right)\right]^2$. Plugging back into the theorem and collecting the terms we get the desired bound. □

The bound holds for any $\alpha > 0$. We plug $\alpha = \sqrt{L_T\left(\{\mathbf{u}_t\}\right) / \left(a L_T\left(\{\mathbf{u}_t\}\right) + cV + b\left\|\mathbf{u}_1\right\|^2\right)}$ in (23) to get,

$$\begin{aligned} L_T(H_\infty) &\leq (1+a) L_T\left(\{\mathbf{u}_t\}\right) + cV + b\left\|\mathbf{u}_1\right\|^2 \\ &\quad + 2\sqrt{\left(a L_T\left(\{\mathbf{u}_t\}\right) + cV + b\left\|\mathbf{u}_1\right\|^2\right) L_T\left(\{\mathbf{u}_t\}\right)} \\ &\leq (1 + a + 2\sqrt{a}) L_T\left(\{\mathbf{u}_t\}\right) + cV + b\left\|\mathbf{u}_1\right\|^2 \\ &\quad + 2\sqrt{\left(cV + b\left\|\mathbf{u}_1\right\|^2\right) L_T\left(\{\mathbf{u}_t\}\right)} \ . \end{aligned}$$

Intuitively, we expect the H_∞ algorithm to perform better when the data is close to linear, that is when $L_T\left(\{\mathbf{u}_t\}\right)$ is small, as, conceptually, it was designed to minimize a loss with respect to weights $\{\mathbf{u}_t\}$. On the other hand, LASER is expected to perform better when the data is hard to predict with linear models, as it is not motivated from this assumption. Indeed, the bounds reflect these observations.

Comparing the last bound with (21) we note a few differences. First, the factor $(1 + a + 2\sqrt{a}) \geq 4$ of $L_T\left(\{\mathbf{u}_t\}\right)$ is worse for H_∞ than for LASER, which is a unit. Second, LASER has worse dependency in the drift $T^{2/3} V^{1/3}$, while for H_∞ it is about $cV + 2\sqrt{cV L_T\left(\{\mathbf{u}_t\}\right)}$. Third, the H_∞ has an additive factor $\sim \sqrt{L_T\left(\{\mathbf{u}_t\}\right)}$, while LASER has an additive logarithmic factor, at most.

Hence, the bound of the H_∞ based algorithm is better when the cumulative loss $L_T\left(\{\mathbf{u}_t\}\right)$ is small. In this case, $4 L_T\left(\{\mathbf{u}_t\}\right)$ is not a large quantity, and as all the other quantities behave like $\sqrt{L_T\left(\{\mathbf{u}_t\}\right)}$, they are small as well. On the other hand, if $L_T\left(\{\mathbf{u}_t\}\right)$ is large, and is linear in T, the first term of the bound becomes dominant, and thus the factor of 4 for the H_∞ algorithm makes its bound

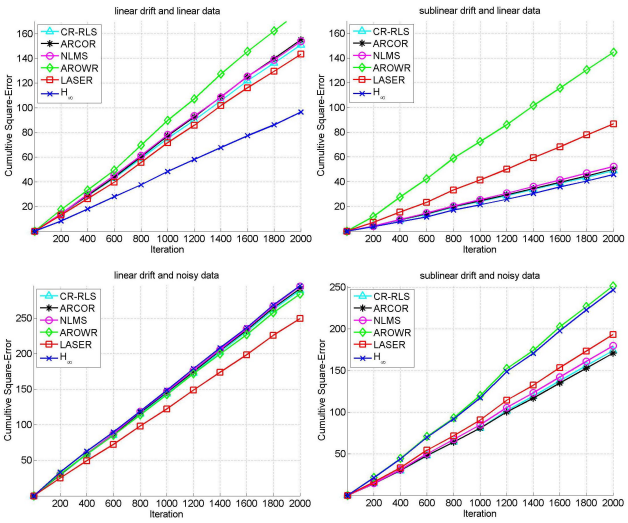

Figure 3: Cumulative squared loss for AROWR, ARCOR, NLMS, CR-RLS, LASER and H_∞ vs iteration. Top left - linear drift and linear data, top right - sublinear drift and linear data, bottom left - linear drift and noisy data, bottom right - sublinear drift and noisy data.

higher than that of LASER. Both bounds were obtained from a min-max approach, either directly (LASER) or via-reduction from filtering (H_∞). The bound of the former is lower in hard problems. Kivinen et al. [26] proposed another approach for filtering with a bound depending on $\sum_t \|\mathbf{u}_t - \mathbf{u}_{t-1}\|$ and not the sum of squares as we have both for LASER and the H_∞-based algorithm.

6 Simulations

We evaluate the LASER and H_∞ algorithms 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 six algorithms: NLMS (normalized least mean square) [3, 5] which is a state-of-the-art first-order algorithm, AROWR (AROW for Regression) [14], ARCOR [35], CR-RLS [11, 30], LASER and H_∞. The algorithms' parameters were tuned using a single random sequence. We repeat each experiment 100 times reporting the mean cumulative square-loss. The results are summarized in Fig. 3 (best viewed in color).

For the first and third datasets (left plots of Fig. 3) we observe the superior performance of the LASER algorithm over previous approaches. LASER has a good tracking ability, fast learning rate and it is designed to perform well in severe conditions like linear drift.

For the second and fourth datasets (right plots of Fig. 3), where we have sublinear drift level, we get that ARCOR outperforms LASER since it is especially designed for sublinear amount of data drift, yet, H_∞ outperforms ARCOR when there is no noise (top-right plot).

For the third and fourth datasets (bottom plots of Fig. 3), where we added noise to labels, the performance of H_∞ degrades, as expected from our discussion in Sec. 5.

7 Related Work

The problem of performing online regression was studied for more than fifty years in statistics, signal processing and machine learning. We already mentioned the work of Widrow and Hoff [37] who 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) [5, 3] that adapts to the input's scale.

There exists a large body of work on this problem proposed by the machine learning community, which clearly cannot be covered fully here. We refer the reader to a encyclopedic book in the subject [10]. Gradient descent based algorithms for regression with the squared loss were proposed by Cesa-Bianchi et al. [8] about two decades ago. These algorithms were generalized and extended by Kivinen and Warmuth [24] using additional regularization functions.

An online version of the ridge regression algorithm in the worst-case setting was proposed and analyzed by Foster [18]. A related algorithm called Aggregating Algorithm (AA) was studied by Vovk [20], and later applied to the problem of linear regression with square loss [36]. The recursive least squares (RLS) [21] is a similar algorithm proposed for adaptive filtering. Both 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 increase with time t, a property which is used to prove logarithmic regret bounds.

The derivation of our algorithm shares similarities with the work of Forster [17] and the work of Moroshko and Crammer [29]. These algorithms are motivated from the last-step min-max predictor. While the algorithms of Forster [17] and Moroshko and Crammer [29] are designed for the stationary setting, our work is primarily designed for the non-stationary setting. Moroshko and Crammer [29] also discussed a weak variant of the non-stationary setting, where the complexity is measured by the total distance from a reference vector $\bar{\mathbf{u}}$, rather than the total distance of consecutive vectors (as in this paper), which is more relevant to non-stationary problems. Note also that Moroshko and Crammer [29] did not derive algorithms for the non-stationary setting, but just show a bound of the weighted min-max algorithm (designed for the stationary setting) in the weak non-stationary setting.

Our work is mostly close to a recent algorithm [35] called ARCOR. This algorithm is based on the RLS 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) [11, 30, 19] 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 $\mathbf{D}$ implicitly rather than explicitly by "averaging" it with a fixed diagonal matrix (see (7)). The Kalman filter [23] and the H_∞ algorithm (e.g. [33]) designed for filtering take a similar approach, yet the exact algebraic form is different (Fig. 1 vs. Fig. 2).

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 [22]. Other approaches to control its norm are to shrink it multiplicatively [25] or by removing old examples [7]. Some of these algorithms were designed to have sparse functions in the kernel space (e.g. [13, 15]). 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 [9, 14, 12], and later in the online convex programming framework [16, 28].

8 Summary and Conclusions

We proposed a novel algorithm for non-stationary online regression designed and analyzed with the squared loss. The algorithm was developed from the last-step minmax predictor for *non-stationary* problems, and we showed an exact recursive form of its solution. We also described an algorithm based on the H_∞ filter, that is motivated from a min-max approach as well, yet for filtering, and bounded its regret. Simulations showed its superior performance in a worst-case (close to a constant per iteration) drift.

An interesting future direction is to extend the algorithm for general loss functions rather than the squared loss. Currently, to implement the algorithm we need to perform either matrix inversion or eigenvector decomposition, we like to design a more efficient version of the algorithm. Additionally, for the algorithm to perform well, the amount of drift V or a bound over it are used by the algorithm. An interesting direction is to design algorithms that automatically detect the level of drift, or are invariant to it.

A Proofs

A.1 Proof of Corollary 8

Proof. Plugging Lem. 5 in Thm. 4 we have for all $(\mathbf{u}_1 \dots \mathbf{u}_T)$,

$$L_T(\text{LASER}) \leq b \left\|\mathbf{u}_1\right\|^2 + cV + L_T(\{\mathbf{u}_t\}) + Y^2 \ln \left|\frac{1}{b}\mathbf{D}_T\right| + c^{-1}Y^2 \sum_{t=1}^{T} \text{Tr}\left(\mathbf{D}_{t-1}\right) .$$

Using Lem. 7 we bound the RHS and get

$$L_T(\text{LASER}) \leq b \left\|\mathbf{u}_1\right\|^2 + L_T(\{\mathbf{u}_t\}) + Y^2 \ln \left|\frac{1}{b}\mathbf{D}_T\right| + c^{-1}Y^2 \text{Tr}\left(\mathbf{D}_0\right) + cV + c^{-1}Y^2 T d \max\left\{\frac{3X^2 + \sqrt{X^4 + 4X^2 c}}{2}, b + X^2\right\} .$$

The term $c^{-1}Y^2\text{Tr}\left(\mathbf{D}_0\right)$ does not depend on T, because $c^{-1}Y^2\text{Tr}\left(\mathbf{D}_0\right) = c^{-1}Y^2 d\frac{bc}{c-b} = \frac{\varepsilon}{1-\varepsilon}Y^2 d$. To show (21), note that $V \leq T\frac{\sqrt{2}Y^2 dX}{\mu^{3/2}} \Leftrightarrow \mu \leq \left(\frac{\sqrt{2}Y^2 dXT}{V}\right)^{2/3} = c$. We thus have that $\left(3X^2 + \sqrt{X^4 + 4X^2 c}\right)/2 \leq \left(3X^2 + \sqrt{8X^2 c}\right)/2 \leq \sqrt{8X^2 c}$, and we get a bound on the right term of (19),

$$\max\left\{\left(3X^2 + \sqrt{X^4 + 4X^2 c}\right)/2, b + X^2\right\} \leq \max\left\{\sqrt{8X^2 c}, b + X^2\right\} \leq 2X\sqrt{2c} .$$

Using this bound and plugging the value of c from (20) we bound (19) and conclude the proof,

$$\left(\frac{\sqrt{2}TY^2 dX}{V}\right)^{2/3} V + Y^2 T d 2X \sqrt{2\left(\frac{\sqrt{2}TY^2 dX}{V}\right)^{-2/3}} = 3\left(\sqrt{2}TY^2 dX\right)^{2/3} V^{1/3} .$$

□

References

[1] P. Auer and M. Warmuth. Tracking the best disjunction. *Electronic Colloquium on Computational Complexity (ECCC)*, 7(70), 2000.

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

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

[4] B.Hassibi and T.Kailath. h_∞ bounds for least-squares estimators. Technical report, Stanford University, 1997.

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

[6] S. Busuttil and Y. Kalnishkan. Online regression competitive with changing predictors. In *ALT*, pages 181–195, 2007.

[7] G. Cavallanti, N. Cesa-Bianchi, and C. Gentile. Tracking the best hyperplane with a simple budget perceptron. *Machine Learning*, 69(2-3):143–167, 2007.

[8] N. Ceas-Bianchi, P. M. Long, and M. K. Warmuth. Worst case quadratic loss bounds for on-line prediction of linear functions by gradient descent. *IEEE Tran. on NN*, 7(3), 1996.

[9] N. Cesa-Bianchi, A. Conconi, and C. Gentile. A second-order perceptron algorithm. *Siam Journal of Commutation*, 34(3):640–668, 2005.

[10] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, USA, 2006.

[11] M.-S. Chen and J.-Y. Yen. Application of the least squares algorithm to the observer design for linear time-varying systems. *Automatic Control, IEEE Tran. on*, 44(9):1742 –1745, 1999.

[12] K. Crammer, M. Dredze, and F. Pereira. Confidence-weighted linear classification for text categorization. *J. Mach. Learn. Res.*, 98888:1891–1926, June 2012.

[13] K. Crammer, J. Kandola, and Y. Singer. Online classification on a budget. In *NIPS*, 2003.

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

[15] O. Dekel, S. Shalev-shwartz, and Y. Singer. The forgetron: A kernel-based perceptron on a fixed budget. In *NIPS 18*, 2005.

[16] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. In *COLT*, pages 257–269, 2010.

[17] J. Forster. On relative loss bounds in generalized linear regression. In *Fundamentals of Computation Theory (FCT)*, 1999.

[18] D. Foster. Prediction in the worst case. *The Annals of Statistics*, 19(2):1084–1090, 1991.

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

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

[21] M. Hayes. 9.4: Recursive least squares. In *Statistical Digital Signal Processing and Modeling*, 1996.

[22] M. Herbster and M. Warmuth. Tracking the best linear predictor. *JMLR*, 1:281–309, 2001.

[23] R. 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.

[24] J. Kivinen and M. K.Warmuth. Exponential gradient versus gradient descent for linear predictors. *Information and Computation*, 132:132–163, 1997.

[25] J. Kivinen, A. Smola, and R. Williamson. Online learning with kernels. In *NIPS*, 2001.

[26] J. Kivinen, M. K. Warmuth, and B. Hassibi. The p-norm generalization of the lms algorithm for adaptive filtering. In *Proc. 13th IFAC Symposium on System Identification*, 2003.

[27] N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Inf. Comput.*, 108(2):212–261, 1994.

[28] H. B. McMahan and M. J. Streeter. Adaptive bound optimization for online convex optimization. In *COLT*, pages 244–256, 2010.

[29] E. Moroshko and K. Crammer. Weighted last-step min-max algorithm with improved sub-logarithmic regret. In *The 23nd International Conference on Algorithmic Learning Theory*, ALT '12, 2012.

[30] M. Salgado, G. Goodwin, and R. Middleton. Modified least squares algorithm incorporating exponential resetting and forgetting. *International J. of Control*, 47(2), 1988.

[31] A. H. Sayed. *Adaptive Filters*. Wiley-IEEE Press, 2008.

[32] D. Simon. A game theory approach to constrained minimax state estimation. *IEEE Transactions on Signal Processing*, 54(2):405–412, 2006.

[33] D. Simon. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. Wiley-Interscience, 2006.

[34] E. Takimoto and M. Warmuth. The last-step minimax algorithm. In *ALT*, 2000.

[35] N. Vaits and K. Crammer. Re-adapting the regularization of weights for non-stationary regression. In *ALT*, 2011.

[36] V. Vovk. Competitive on-line statistics. *International Statistical Review*, 69, 2001.

[37] B. Widrow and M. E. Hoff. Adaptive switching circuits. In *Institute of Radio Engineers, Western Electronic Show and Convention, Convention Record, Part 4*, pages 96–104, 1960.

B APPENDIX SUPPLEMENTARY MATERIAL

B.1 Proof of Lem. 1

Proof. We calculate

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

□

B.2 Proof of Lem. 2

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 $\mathbf{D}_1 = b\mathbf{I} + \mathbf{x}_1\mathbf{x}_1^\top$, $\mathbf{e}_1 = y_1\mathbf{x}_1$, and $f_1 = y_1^2$. We proceed by induction, assume that, $P_{t-1}(\mathbf{u}_{t-1}) = \mathbf{u}_{t-1}^\top\mathbf{D}_{t-1}\mathbf{u}_{t-1} - 2\mathbf{u}_{t-1}^\top\mathbf{e}_{t-1} + f_{t-1}$. Applying Lem. 1 we get,

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

Using Woodbury identity we continue to develop the last equation,

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

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

B.3 Proof of Lem. 3

Proof. We first use the Woodbury equation to get the following two identities

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

and

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

Multiplying both identities with each other we get,

$$\begin{aligned}&\mathbf{D}_t^{-1}\left(\mathbf{I}+c^{-1}\mathbf{D}_{t-1}\right)^{-1} \\ =\ &\left[\mathbf{D}_{t-1}^{-1}+c^{-1}\mathbf{I}\right. \\ &\left.- \frac{\left(\mathbf{D}_{t-1}^{-1}+c^{-1}\mathbf{I}\right)\mathbf{x}_t\mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1}+c^{-1}\mathbf{I}\right)}{1+\mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1}+c^{-1}\mathbf{I}\right)\mathbf{x}_t}\right]\left[\mathbf{I}\right. \\ &\left.- c^{-1}\left(\mathbf{D}_{t-1}^{-1}+c^{-1}\mathbf{I}\right)^{-1}\right] \\ =\ &\mathbf{D}_{t-1}^{-1}-\frac{\left(\mathbf{D}_{t-1}^{-1}+c^{-1}\mathbf{I}\right)\mathbf{x}_t\mathbf{x}_t^\top\mathbf{D}_{t-1}^{-1}}{1+\mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1}+c^{-1}\mathbf{I}\right)\mathbf{x}_t} \qquad (24)\end{aligned}$$

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

$$\begin{aligned}&\left(\mathbf{I}+c^{-1}\mathbf{D}_{t-1}\right)^{-1}\mathbf{D}_t^{-1} \\ =\ &\mathbf{D}_{t-1}^{-1}-\frac{\mathbf{D}_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1}+c^{-1}\mathbf{I}\right)}{1+\mathbf{x}_t^\top\left(\mathbf{D}_{t-1}^{-1}+c^{-1}\mathbf{I}\right)\mathbf{x}_t} \qquad (25)\end{aligned}$$

Finally, from (24) we get,

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

We develop the last equality and use (24) and (25) in the second equality below,

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

□

B.4 Derivations for Thm. 4

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

where the last equality follows (8). We proceed to develop the last equality,

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

B.5 Details for the bound (22)

To show the bound (22), note that, $V \geq T\frac{Y^2dM}{\mu^2} \Leftrightarrow \mu \geq \sqrt{\frac{TY^2dM}{V}} = c$. We thus have that the right term of (19) is

upper bounded as follows,

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

Using this bound and plugging $c = \sqrt{Y^2dMT/V}$ we bound (19), $\sqrt{\frac{Y^2dMT}{V}}V + \frac{1}{\sqrt{\frac{Y^2dMT}{V}}}TdY^2M = 2\sqrt{Y^2dMTV}$.

B.6 Proof of Lem. 6

Proof. For the first property of the lemma we have that $f(\lambda) = \lambda\beta/(\lambda+\beta) + x^2 \leq \beta \times 1 + x^2$. 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. □