Competitive Randomized Nonlinear Prediction Under Additive Noise

Yasin Yilmaz and Suleyman S. Kozat

***Abstract*—We consider sequential nonlinear prediction of a bounded, real-valued and deterministic signal from its noise-corrupted past samples in a competitive algorithm framework. We introduce a randomized algorithm based on context-trees [1]. The introduced algorithm asymptotically achieves the performance of the best piecewise affine model that can both select the best partition of the past observations space (from a doubly exponential number of possible partitions) and the affine model parameters based on the desired clean signal in hindsight. Although the performance measure including the loss function is defined with respect to the noise-free clean signal, the clean signal, its past samples or prediction errors are not available for training or constructing predictions. We demonstrate the performance of the introduced algorithm when applied to certain chaotic signals.**

***Index Terms*—Nonlinear prediction, context-tree, competitive prediction, additive noise, sequential decisions.**

I. Introduction

In this letter, we consider sequential (online) nonlinear prediction of an arbitrary, deterministic and bounded signal from its noise-corrupted past samples under square error loss. In this fundamental signal processing problem [2], a bounded deterministic signal $\{x[t]\}_{t\geq 1}, |x[t]| \leq b_x, x[t] \in \mathbb{R}, 0 < b_x < \infty$ is observed through an additive noise channel, $Y[t] = x[t] + N[t]$, where $\{N[t]\}_{t\geq 1}$ is an i.i.d., bounded noise process with variance σ_n^2 such that $|N[t]| \leq b_n, N[t] \in \mathbb{R}, 0 < b_n < \infty$ with probability 1[1]. Hence, $|Y[t]| \leq b_y, b_y \triangleq b_x + b_n$ with probability 1. Then, the underlying signal $x[t]$ is predicted using the noise-corrupted past samples $\{Y[1], \ldots, Y[t-1]\}$ at each time t. We emphasize that although we desire to predict the underlying signal $\{x[t]\}_{t\geq 1}$ and the performance measure including the loss function is defined with respect to $\{x[t]\}_{t\geq 1}$, the desired clean signal $\{x[t]\}_{t\geq 1}$ is not available for prediction or training. In this sense, this framework differs from common classical adaptive signal processing approaches [3], where the desired clean signal, certain statistics or past samples are usually available for training or constructing predictions. In this letter, we refrain from making stochastic assumptions on the desired signal $\{x[t]\}_{t\geq 1}$ and require uniformly good performance for any deterministic signal $\{x[t]\}_{t\geq 1}$.

Since we make no such stochastic assumptions on the desired signal, we introduce a competitive framework in order to define a meaningful performance measure. In this competitive framework, we have, say m sequential (online) prediction algorithms as the competition class producing outputs $\{\hat{X}_k[t]\}_{t\geq 1}, k = 1, \ldots, p$, that "hypothetically" work in parallel to predict the underlying signal $\{x[t]\}_{t\geq 1}$. At each time t, each sequential algorithm suffers the loss $(x[t]-\hat{X}_k[t])^2$ (which is not available to us since $\{x[t]\}_{t\geq 1}$ is not observable). Our goal is then to find a sequential predictor that asymptotically achieves the performance of even the best algorithm in this class uniformly for any deterministic, bounded and arbitrary signal. Specifically, we seek a sequential predictor, say $\hat{X}[t]$, that has access to only noisy past samples $\{Y[1], \ldots, Y[t-1]\}$, predictions of the constituent algorithms $\{\hat{X}_k[1], \ldots, \hat{X}_k[t]\}, k = 1, \ldots, p$, never observes $\{x[t]\}_{t\geq 1}$ and satisfies

$$\lim_{n\to\infty} \frac{1}{n} E\left[\sum_{t=1}^{n} (x[t] - \hat{X}[t])^2 - \sum_{t=1}^{n} (x[t] - \hat{X}_k[t])^2\right] = 0 \tag{1}$$

for all k and n when it is used to predict any $\{x[t]\}_{t\geq 1}$. Here, the expectation is with respect to the noise process and the randomization of the introduced algorithm.

In this letter, we consider the framework where the underlying competition class is the class of certain nonlinear models, i.e., piecewise affine models represented on a context-tree. As an example, suppose as shown in Fig. 1(a), we divide the space of the most recent observation space $[-b_y, b_y]$ (which $Y[t-1]$ belongs to) into four disjoint regions $\Delta_1, \ldots, \Delta_4$ such that $[-b_y, b_y] = \bigcup_{i=1}^{4} \Delta_i$ and assign each region an affine model say $(w_i Y[t-1] + c_i), w_i, c_i \in \mathbb{R}, i = 1, \ldots, 4$. These assignments define a piecewise affine predictor where the prediction at each time t is given by $(w_i Y[t-1]+c_i)$ if $Y[t-1] \in \Delta_i$. For all $w_i, c_i \in \mathbb{R}$, one can define similar piecewise affine predictors yielding a competition class that has a continuum of predictors. Although one can approximate smoothly varying nonlinear functions by increasing the number of regions and the number of past samples used in prediction in this piecewise affine model, the boundaries of the piecewise regions are fixed. To make the boundaries of the piecewise regions also a design parameter, we will use the notion of context-trees to represent a doubly exponential number different partitions of the past observations space in the next section.

To this end, we introduce a novel randomized prediction algorithm based on context-trees that uses only the past noisy samples of a desired signal and has computational complexity only linear in the depth of the context-tree per prediction. Without ever observing the desired clean signal, this algorithm will achieve the performance of the best piecewise affine predictor that can both choose its partition of the past observations space (from a class of a doubly exponential number of possible partitions) and tune the parameters of the affine models for these piecewise regions using the clean desired signal. The functional forms of the randomization weights of the introduced

Manuscript received August 01, 2009; revised December 13, 2009. First published January 08, 2010; current version published February 10, 2010. This work is supported by TUBITAK Career Award, Contract 108E195. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ricardo Merched.

The authors are with the EEE Department, Koc University, Istanbul, Turkey (e-mail: skozat,yayilmaz@ku.edu.tr).

Digital Object Identifier 10.1109/LSP.2009.2039950

[1]We reserve capital letters to random variables and bold letters to column vectors.

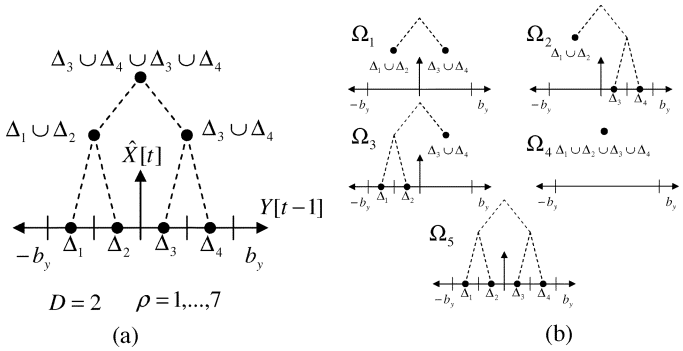

Fig. 1. (a) Binary context tree that partitions $[-b_y, b_y]$. This tree has depth $D = 2$ with four leaves and seven nodes. Each node is assigned a region as the union of the regions assigned to its children. (b) The complete subtrees along with the partitions they define. A depth-D context-tree defines $m \approx 1.5^{2^D}$ such subtrees or partitions.

A Pseudo-code of the Randomized Prediction Algorithm:

% Initialize

For $\rho = 1, \ldots, 2^{D+1} - 1$: $F_\rho[0] = \epsilon_1$, $\Gamma_\rho[0] = \epsilon_2$, where ϵ_1 and ϵ_2 are small positive constants. (line A)

% Proceed

For $t = 1, \ldots$

Find nodes such that $Y[t-1] \in \Delta_\rho$ and store them in the vector $\boldsymbol{v}$ starting from the top node to the leaf node.

% Form the prediction.

$\boldsymbol{z}(1) = 1/2$.

For $i = 2 : D+1$: $\boldsymbol{z}(i) = \frac{1}{2} F_{(\boldsymbol{v}(i),s)}[t-1]\boldsymbol{z}(i-1)$ (line B)

(where $(\boldsymbol{v}(i), s)$ is the sibling node of $\boldsymbol{v}(i)$, i.e., they are the children of the same node).

Construct a weight vector $\boldsymbol{\eta}$ of size $D+1$ as $\boldsymbol{\eta}(i) = (\boldsymbol{z}(i)\Gamma_{\boldsymbol{v}(i)}[t-1])/\left(F_{\boldsymbol{v}(1)}[t-1]\right)$.

Select randomly one of the entries of $\boldsymbol{v}$ by using the weights in $\boldsymbol{\eta}$, i.e., $\boldsymbol{v}(i)$ is selected with probability $\boldsymbol{\eta}(i)$.

Set $\hat{X}[t] = \hat{X}_{\boldsymbol{v}(i)}[t]$ if i is the selected entry.

% Update after observing $Y[t]$. All the other nodes remain the same.

For $i = D+1, \ldots, 1$,

$\Gamma_{\boldsymbol{v}(i)}[t] = \Gamma_{\boldsymbol{v}(i)}[t-1] \exp(-a(Y[t] - \hat{X}_{\boldsymbol{v}(i)}[t])^2)$ (line C)

If $i == D+1$, then $F_{\boldsymbol{v}(i)}[t] = \Gamma_{\boldsymbol{v}(i)}[t]$ (line D)

If $i \neq D+1$, then $F_{\boldsymbol{v}(i)}[t] = \frac{1}{2} F_{(\boldsymbol{v}(i),l)}[t] F_{(\boldsymbol{v}(i),r)}[t] + \frac{1}{2}\Gamma_{\boldsymbol{v}(i)}[t]$. (line E)

Update $\hat{X}_{\boldsymbol{v}(i)}[t+1]$ from $\hat{X}_{\boldsymbol{v}(i)}[t]$

Fig. 2. Randomized sequential prediction algorithm using context-trees. Finding the nodes that $Y[t-1]$ belongs to require $O(D+1)$ operations since one only needs to find the leaf node that $Y[t-1]$ belongs and proceeds to the top. At each time t, the algorithm combines and updates the parameters of only $D+1$ node predictors with computational complexity $O(D+1)$.

algorithm are similar to the weights used in [1] to construct weighted predictions. However, note that the algorithm of [1] trains on and uses the past observations of the clean desired signal $\{x[t]\}_{t\geq 1}$, which is unavailable here. Furthermore, we use these weights to define a randomized algorithm instead of using convex combination ideas as in [1]. Although we discuss only affine models in the introduction, as shown in the next section, one can assign arbitrary predictors (or regressors) to each region. The affine models are specifically used to yield smoothly varying arbitrary nonlinear models.

In the next section, we first summarize the notion of context trees. We then introduce a randomized predictor constructed using context-trees that competes against all piecewise affine models defined on the context-tree and that requires a computational complexity only linear in the depth of the context-tree per prediction. We conclude the paper with simulations to illustrate the performance of the introduced algorithm using chaotic signals and provide some remarks.

II. Algorithm Description and Results

In this section, as an illustrative example, we present a binary context-tree to partition the space of only the most recent past observation, i.e., $[-b_y, b_y]$ where $Y[t-1]$ belongs to. As shown in Fig. 1(a), a depth-D binary context-tree ($D = 2$ in this figure) has 2^D leaves and $2^{D+1} - 1$ nodes. Each node on the context-tree, if it is not a leaf node, has two children: the left hand side child and the right hand side child. We use this context-tree to define different partitions of $[-b_y, b_y]$ as follows. We first assign each leaf of the context-tree a different region of $[-b_y, b_y]$ as seen in Fig. 1(a). Each node on the context-tree is then assigned the region which is constructed as the union of the regions assigned to its children. On this context-tree, one can define a doubly exponential number, $m \approx 1.5^{2^D}$ [1], of different prunings or "complete" subtrees. As an example, for a depth-2 context-tree, we provide five different subtrees in Fig. 1(b). We call these subtrees "complete" since the union of the regions assigned to the leaves of a subtree (which are the nodes or the leaves of the original context-tree) yields $[-b_y, b_y]$. Hence, a subtree along with the regions assigned to its leaves defines a partition of $[-b_y, b_y]$. Given a depth-D binary context-tree, we get a doubly exponential number $m \approx 1.5^{2^D}$ of such partitions, say $\Omega_k, k = 1, \ldots, m$. For each partition Ω_k, we represent the constituent regions as $\Omega_k \triangleq \{\Lambda_{k,1}, \ldots, \Lambda_{k,K_k}\}$ such that $[-b_y, b_y] = \bigcup_i \Lambda_{k,i}$, K_k is the number of the leaves in the partition and $\Lambda_{k,i}$ are the regions assigned to the leaves of the partition, e.g., for Ω_1, we have $\Lambda_{1,1} = \Delta_1 \cup \Delta_2, \Lambda_{1,2} = \Delta_3 \cup \Delta_4$ and $K_1 = 2$. Suppose, given this context-tree, we assign each node $\rho, \rho = 1, \ldots, 2^{D+1} - 1$, a sequential predictor $\hat{X}_\rho[t]$ and define sequential predictors $\hat{X}_{\Omega_k}[t], k = 1, \ldots, m$, corresponding to each partition Ω_k as: $\hat{X}_{\Omega_k}[t] = \hat{X}_\rho[t]$ if $Y[t-1] \in \Lambda_{k,j}$ (for some j) and $\Delta_\rho = \Lambda_{k,j}$. We initially define these m sequential predictors as our competition class. Later, by selecting $\hat{X}_\rho[t]$'s as certain sequential affine predictors, we will demonstrate that our algorithm asymptotically achieves the performance of the best piecewise affine model which can tune both its partitioning of the real line as well as the affine models in each region to $\{x[t]\}_{t\geq 1}$. To this end, we introduce the randomized algorithm in Fig. 2, i.e., $\hat{X}[t]$, that is constructed using context tree weighting method [4]. This randomized algorithm hypothetically constructs all $\hat{X}_{\Omega_k}[t], k = 1, \ldots, m$, and run these in parallel. At each time t, the final estimation $\hat{X}[t]$ selects one of the outputs $\hat{X}_{\Omega_k}(t)$ to repeat it as its prediction, where the selection weights are calculated proportional to the performance of each $\hat{X}_{\Omega_k}[t]$ on the past data. However, note that, although there are m different piecewise competing algorithms, at each time t, each $\hat{X}_{\Omega_k}[t]$ is equal to one of the $D+1$ node predictions that $Y[t-1]$ belongs to. Hence, as shown in [1], at each time t, for the nodes that $Y[t-1]$ belongs to (these nodes are stored in vector $\boldsymbol{v}$ in Fig. 2), all the weights assigned to $\hat{X}_{\Omega_k}[t], k = 1, \ldots, m$ can be merged using certain functions of node performance. These functions are represented as $F_\rho[t]$ and $\Gamma_\rho[t]$ in Fig. 2, and updated recursively in (line C), (line D) and (line E) in Fig. 2 with computational complexity only linear in depth of the context tree. At each time t, these functions that reflect the combined prediction performance of that node on the past data are used to construct the probabilities $\boldsymbol{\eta}$ that are used for randomization. The randomized algorithm introduced in Fig. 2 satisfies:

Theorem 1: Let $Y[t] \triangleq x[t] + N[t]$ represents the observation sequence such that $\{x[t]\}_{t\geq 1}$ is the desired deterministic signal with $|x[t]| \leq b_x$ and $N[t]$ is i.i.d. with variance $\sigma_n^2, |N[t]| \leq b_n$, i.e., $|Y[t]| \leq b_y = b_x + b_n$ with probability 1. The sequential randomized prediction algorithm presented in Fig. 2, which uses only the past noisy observations $\{Y[1], \ldots, Y[t-1]\}, \{\hat{X}_\rho[1], \ldots, \hat{X}_\rho[t]\}, \rho =$

$1,\dots,2^{D+1}-1$ and never observes $\{x[t]\}_{t\geq 1}$ or prediction errors $(x[t]-\hat{X}[t]),(x[t]-\hat{X}_\rho[t])$, achieves

$$\frac{1}{n}E\left[\sum_{t=1}^{n}(x[t]-\hat{X}[t])^2 - \sum_{t=1}^{n}(x[t]-\hat{X}_{\Omega_k}[t])^2\right]\leq O\left(\frac{2^{\frac{D+1}{2}}}{\sqrt{n}}\right) \quad (2)$$

for all n and any partition Ω_k, when it is applied to predict any $\{x[t]\}_{t\geq 1}$. Here, the expectation is with respect to the noise process and randomization.

To get the upper bound in (2), we need to set $a=\sqrt{8\,2^{D+1}\ln 2/n}$ in Fig. 2. Note that although a is optimized over n, this need for *a priori* knowledge of n can be readily surpassed by applying the algorithm over exponentially increasing segments of $\{Y[t]\}_{t\geq 1}$. To achieve the performance of the best affine model with the best partition, we assign each node a special affine predictor studied in [2], which uses only the past samples $\{Y[1],\dots,Y[t-1]\}$ that belong to that node as $\hat{X}_\rho[t]\triangleq \tilde{w}_\rho[t-1]Y[t-1]+\tilde{c}_\rho[t-1]$, where

$$\begin{aligned}&[\tilde{w}_\rho[t]\quad \tilde{c}_\rho[t]]^T\\ &\quad=\boldsymbol{R}^{-1}[t-1]\boldsymbol{p}[t-1],\boldsymbol{R}[t-1]\\ &\quad\triangleq\left(\sum_{l=1}^{t}\boldsymbol{Y}[l-1]\boldsymbol{Y}[l-1]^T s_\rho[l]+\delta I\right)\end{aligned} \quad (3)$$

and $\boldsymbol{p}[t-1]\triangleq\sum_{l=1}^{t-1}Y[l]\boldsymbol{Y}[l-1]s_\rho[l]$, where $\boldsymbol{Y}[l]=[Y[l]\quad 1]^T, s_\rho[l]$ is the indicator variable for node ρ, i.e., $s_\rho[l]=1$ if $Y[l-1]\in\Delta_\rho$ otherwise $s_\rho[l]=0$, and $(\cdot)^T$ is the transpose. The affine predictor in (3) is a least squares predictor that trains only on the observed data $\{Y[t]\}_{t\geq 1}$ that belongs to that node, i.e., that falls into the region Δ_ρ. Note that the update in (3) can be implemented with $O(D)$ computations using the matrix inversion lemma [3]. For the randomized predictor in Fig. 2 using these least squares predictors in each node, we have the following result:

Corollary 1: The sequential randomized prediction algorithm of Fig. 2, with the affine predictors (3) at each node ρ that only depends on the past noisy observations $\{Y[1],\dots,Y[t-1]\}$ and never observes $\{x[t]\}_{t\geq 1}$, achieves

$$\begin{aligned}&\frac{1}{n}E\left[\sum_{t=1}^{n}(x[t]-\hat{X}[t])^2 - \sum_{t=1}^{n}(x[t]-w_{k,d_k[t]}Y[t-1]+c_{k,d_k[t]})^2\right]\\ &\leq O\left(\sqrt{\frac{\ln n}{n}}\right)+O\left(\frac{2^{\frac{D+1}{2}}}{\sqrt{n}}\right)\end{aligned} \quad (4)$$

for all $n, w_{k,j}\in\mathbb{R}, c_{k,j}\in\mathbb{R}$, $j=1,\dots,K_k$, for all partitions Ω_k, when it is applied to predict any $\{x[t]\}_{t\geq 1}$. Here, $d_k[t]$ is the selection variable for partition Ω_k such that if $Y[t-1]\in\Lambda_{k,j}, d_k[t]=j$.

Note that one can use the binary context tree to partition the space of $\{Y[t-1],Y[t-2],\dots,Y[t-r]\}$ for some r or use dth order affine predictors in each node that use $\{Y[t-1],\dots,Y[t-d]\}$ as input regressor for some d. To use dth order regressors for affine prediction, one need to only update (3). To partition $[-b_y,b_y]^r$, one needs to change the line in Fig. 2 that explains how to find the leaf node that $\{Y[t-1],\dots,Y[t-r]\}$ belongs to. As explained in the caption of Fig. 2, the randomized algorithm has $O(D+1)$ computational complexity at each time t since finding the nodes used for prediction as well as updating the node predictors require only $O(D+1)$ additions and multiplications. The algorithm also has $O(2^{D+1})$ storage complexity to store weights and predictors corresponding to all nodes.

Remark 1: Note that the corollary holds for any $w_{k,j}\in\mathbb{R}, c_{k,j}\in\mathbb{R}, i=1,\dots,K_k$, even with the ones that are tuned by observing the whole $\{x[t]\}_{t\geq 1}$ and $\{Y[t]\}_{t\geq 1}$, in hindsight, for all n, before we even start predicting $\{x[t]\}_{t\geq 1}$. Hence, the algorithm of corollary 1 asymptotically achieves the performance of the best piecewise affine predictor that can choose both its partitions as well as the prediction coefficients for that partition based on $\{x[t]\}_{t\geq 1}$ and $\{Y[t]\}_{t\geq 1}$ in hindsight.

Remark 2: We emphasize that the best piecewise model with the optimal weights tuned using all $\{x[t]\}_{t\geq 1}$ corresponds to the finest partition, i.e., the fifth partition shown in Fig. 1(b). Hence, at first sight, one is tempted to use the sequential predictor corresponding to the finest partition, i.e., $\hat{X}_{\Omega_5}[t]$ in Fig. 1(b), with the sequential algorithms from (3). However, note that this sequential predictor needs to learn the corresponding optimal weights in each region sequentially, hence, it may not be the best sequential algorithm as shown in the simulations section. Furthermore the bound in (4) holds for all partitions and the regret of our algorithm with respect to the best piecewise affine model corresponding to the finest partition is the largest, however, still $o(n)$.

Outline of the Proofs of the Theorem 1 and the Corollary 1: For each sequential predictor corresponding to a partition, we define a function of its prediction loss on $\{Y[t]\}_{t\geq 1}$ as $F_{\Omega_k}[n]\triangleq\exp(-a\sum_{t=1}^{n}(Y[t]-\hat{X}_{\Omega_k}[t])^2), a\in\mathrm{R}^+$, at each time n, that only depends on noisy observations. We further define a weighted sum of these functions $F[n]=\sum_{k=1}^{m}2^{-C(\Omega_k)}F_{\Omega_k}[n]$, where $C(\Omega_k)$ are certain weights introduced in [4] satisfying $\sum_{k=1}^{m}2^{-C(\Omega_k)}=1$ and $C(\Omega_k)\leq 2K_k-1\leq 2\,2^D-1$. The weights $C(\Omega_k)$ are introduced for proof purposes and are not explicitly used in the final algorithm. Clearly, $F[n]$ is as large as any $2^{-C(\Omega_k)}F_{\Omega_k}[n]$. Our initial goal is to show that for some randomized predictor, say $\hat{X}[t], F_{\hat{X}}[n]$ is as large as $F[n]$, i.e., the performance of $\hat{X}[t]$ for predicting $\{Y[t]\}_{t\geq 1}$ is as good as any $\hat{X}_{\Omega_k}[t]$. We will then use this predictor for prediction of $\{x[t]\}_{t\geq 1}$. To accomplish the first step, we observe that $F[n]=\prod_{t=1}^{n}(F[t]/F[t-1])$ by telescoping. However for each term in this product, we have

$$\begin{aligned}\frac{F[t]}{F[t-1]}&=\sum_{k=1}^{m}\frac{2^{-C(\Omega_k)}F_{\Omega_k}[t-1]}{F[t-1]}\\ &\quad\times\exp(-a(Y[t]-\hat{X}_{\Omega_k}[t])^2)\\ &\leq\exp\left\{-aE\left[(Y[t]-\hat{X}_{\Omega_k}[t])^2\right]+\frac{a^2}{8}\right\}\end{aligned} \quad (5)$$

where the expectation in the last inequality is with respect to the probabilities $(2^{-C(\Omega_k)}F_{\Omega_k}[t-1])/(F[t-1])$ and the inequality follows from Hoeffding's inequality [5]. Hence, if we construct a randomized predictor, say $\hat{X}[t]$, that outputs $\hat{X}_{\Omega_k}[t]$ as its prediction with probability $(2^{-C(\Omega_k)}F_{\Omega_k}[t-1])/(F[t-1])$ at each time t, then by (5), the accumulated loss of this algorithm will satisfy $\ln F_{\Omega_k}[n]-C(\Omega_k)\ln 2=-a\sum_{t=1}^{n}(Y[t]-\hat{X}_{\Omega_k}[t])^2-C(\Omega_k)\ln 2\leq\ln F[n]\leq -a\sum_{t=1}^{n}E[(Y[t]-\hat{X}[t])^2]+(na^2)/(8)$ for all k. Since $C(\Omega_k)<2^{D+1}$, setting $a=\sqrt{8\,2^{D+1}\ln 2/n}$, yields an upper bound $\sqrt{n2^{D+1}\ln 2/8}$ on the accumulated loss of $\hat{X}[t]$. This upper bound yields the upper bound in the theorem after normalization with n. Hence the randomized predictor $\hat{X}[t]$ is the desired predictor if the goal was to predict $\{Y[t]\}_{t\geq 1}$. However, even in this case, this

randomized predictor $\hat{X}[t]$, at each time t, needs to calculate and update a doubly exponential number, i.e., m, of predictions, which is clearly an impossible feat even for modest m. However, note that, in $\hat{X}[t]$, at each time t, only $D+1$ node predictions $\hat{X}_\rho[t]$ that $Y[t-1]$ belongs to are used such that all the weights with same node predictions can be merged. It can be shown as in [1] that if one defines certain functions of performance for each node as $F_\rho[t], \Gamma_\rho[t]$ which are initialized in (line A) and updated in (line C), (line D) and (line E) of Fig. 2, then the corresponding $F[t]$ can be written as $F[t] = \sum_{i=1}^{D+1} \boldsymbol{z}(i) \exp(-a \sum_{j=1}^{t} [(Y[j] - \hat{X}_{\boldsymbol{v}(i)}[j])^2 s_{\boldsymbol{v}(i)}[j]])$, where $\boldsymbol{v}$ is the vector of nodes that $Y[t-1]$ belongs to, the recursion for $\boldsymbol{z}$ is given in (line B) of Fig. 2 and $\boldsymbol{v}(i)$ is the ith entry of the vector $\boldsymbol{v}$. Hence $\hat{X}[t]$ is defined as the randomized predictor using probabilities $\boldsymbol{\eta}(i) \triangleq (\sum_{i=1}^{D+1} \boldsymbol{z}(i) \exp(-c \sum_{j=1}^{t-1} [(Y[j] - \hat{X}_{\boldsymbol{v}(i)}[j])^2 s_{\boldsymbol{v}(i)}[j]]))/(F[t-1])$. Note that all these performance bounds are with respect to the prediction of $\{Y[t]\}_{t\geq 1}$ not with respect to the prediction of the desired signal $\{x[t]\}_{t\geq 1}$. However, we observe that $E[(Y[t] - \hat{X}_{\Omega_k}[t])^2] = E[(x[t] + N[t] - \hat{X}_{\Omega_k}[t])^2] = E[(x[t] - \hat{X}_{\Omega_k}[t])^2] + \sigma_n^2$ and $E[(Y[t] - \hat{X}[t])^2] = E[(x[t] + N[t] - \hat{X}[t])^2] = E[(x[t] - \hat{X}[t])^2] + \sigma_n^2$, since $N[t]$ is i.i.d. and independent from both $\hat{X}_{\Omega_k}[t]$ and $\hat{X}[t]$. Note that $\hat{X}_{\Omega_k}[t]$ and $\hat{X}[t]$ are sequential and do not use $Y[t]$. Hence, using these equations yields the result in (2). This concludes the proof of the Theorem 1. □

To get the corollary 1, it has been shown in [2] that the affine predictor $\hat{X}_\rho[t]$ in (3) achieves $E[\sum_{t=1}^{n} (x[t] - \hat{X}_\rho[t])^2 s_\rho[t] - \sum_{t=1}^{n} (x[t] - wY[t-1] - c)^2 s_\rho[t]] \leq 2\ln(n_\rho) + O(1)$ for all $w, c \in \mathbb{R}$ and n, where n_ρ is the number of times node ρ is used in prediction, i.e., $Y[t-1] \in \Delta_\rho$. Applying this result to any $\hat{X}_{\Omega_k}[t]$ that uses these affine predictors in each node yields, after maximization over n_ρ's, $E[\sum_{t=1}^{n} (x[t] - \hat{X}_{\Omega_k}[t])^2 - \sum_{t=1}^{n} (x[t] - wY[t-1] - c)^2] \leq 2\ln(n) + O(1)$. Combining this bound with (2) and selecting an appropriate value for a yields the result in the corollary 1. This completes the outline of the proof of the corollary 1. □

III. Simulations and Conclusion

In this section, we illustrate the performance of the introduced algorithm when it is used to predict noise-corrupted chaotic signals generated using the Duffing map described by $x[t] = c_1 x[t-2] + c_2 x[t-1] - (x[t-1])^3$. The Duffing map demonstrates chaotic behavior when $c_1 = -0.2$ and $c_2 = 2.75$. For these values of c_1 and c_2, we plot in Fig. 3(b), a sample function generated using the Duffing map. Note that although the sample function is completely predictable from the governing dynamical equations using only the last two samples, it exhibits rather erratic behavior, and is in fact known to exhibit chaotic behavior for this set of coefficients. We also plot the corresponding attractor for the Duffing map in Fig. 3(a) showing its highly nonlinear nature. The desired signal $x[t]$ is then corrupted by an additive noise $N[t]$ with standard deviation 0.05. In Fig. 3(c), we plot the accumulated and normalized MSE of different algorithms averaged over 100 random iterations of $\{x[t]\}_{t\geq 1}$ and $\{N[t]\}_{t\geq 1}$. In this figure, the context-tree based algorithms use a context-tree of depth-6, $a = 1$ and first order sequential linear predictors in each node. In Fig. 3(c), we have the context-tree algorithm from Fig. 2 "alg"; the sequential algorithm corresponding to the finest partition (discussed in Remark 2) on this tree "finest"; the context tree algorithm that trains on the clean signal $\{x[t]\}_{t\geq 1}$, however, still uses $Y[t-1]$ as the regressor "clean alg". Since at each time the introduced algorithm requires $O(7)$ computations, we also simulate a 7th order linear least squares algorithm using $\{Y[t-1], \ldots, Y[t-7]\}$ as its input regressor [3] "RLS". Note that this RLS algorithm provides significantly worse performance since it tries to approximate the nonlinear terms in the Duffing map by linear combinations. In order to model the nonlinear terms in the Duffing Map, we also implement a 4th order linear least squares algorithm using $\{Y[t-1], (Y[t-1])^2, \ldots, (Y[t-1])^4\}$ as the input regressor "NRLS." We observe in these simulations that the "alg" algorithm is able to outperform the "finest" algorithm in the initial samples since the finest partition needs to learn all the affine models used in the leaves. The context-tree algorithm is able to exploit the smaller subtrees (or coarser models) which have less parameters to train, hence it provides better performance in the start of the simulations against all algorithms. As the data length grows, when the "finest" algorithm has enough data to train on, both algorithms provide similar performance. The performance of the context-tree algorithm trained on noisy samples is nearly the same as the performance of the "clean alg" algorithm, i.e., the curves are nearly the same. This result was expected as shown in the proof of the introduced algorithm. For these chaotic signals, the introduced algorithm outperforms all other algorithms that also use only the noise-corrupted samples.

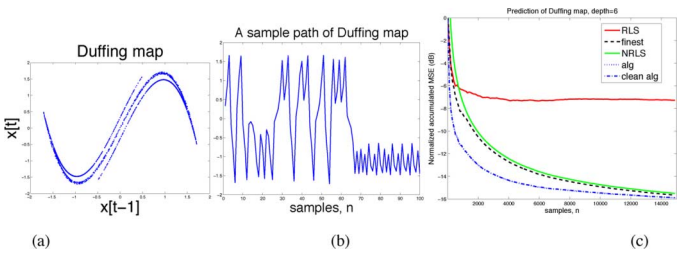

Fig. 3. Duffing map. (a) Attractor of the Duffing map. (b) Sample function. (c) Normalized MSE for the algorithms described in the text.

In this letter, we introduced a novel randomized sequential prediction algorithm that only uses noise-corrupted past samples of a deterministic desired signal to predict the clean desired signal. This algorithm is shown to achieve asymptotically the performance of the best piecewise affine model that can both select the best partition of the space of past regressor (from a doubly exponential number of possible partitions) and the affine model parameters based on the clean desired signal. We demonstrated the performance of the introduced algorithm when it is used to predict chaotic signals generated using the Duffing map.

References

[1] S. S. Kozat, A. C. Singer, and G. Zeitler, "Universal piecewise linear prediction via context trees," *IEEE Trans. Signal Process.*, vol. 55, pp. 3730–3745, 2007.

[2] S. S. Kozat and A. C. Singer, "Competitive prediction under additive noise," *IEEE Trans. Signal Process.*, vol. 57, Sept. 2009.

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

[4] F. Willems, Y. Shtarkov, and T. Tjalkens, "The context-tree weighting method: Basic properties," *IEEE Trans. Inform. Theory*, vol. IT-41, pp. 653–664, May 1995.

[5] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *J. Amer. Statist. Assoc*, vol. 58, pp. 13–30, 1963.