Optimal Attack against Autoregressive Models by Manipulating the Environment

Yiding Chen, Xiaojin Zhu
Department of Computer Sciences, University of Wisconsin-Madison
{yiding, jerryzhu}@cs.wisc.edu

Abstract
We describe an optimal adversarial attack formulation against autoregressive time series forecast using Linear Quadratic Regulator (LQR). In this threat model, the environment evolves according to a dynamical system; an autoregressive model observes the current environment state and predicts its future values; an attacker has the ability to modify the environment state in order to manipulate future autoregressive forecasts. The attacker's goal is to force autoregressive forecasts into tracking a target trajectory while minimizing its attack expenditure. In the white-box setting where the attacker knows the environment and forecast models, we present the optimal attack using LQR for linear models, and Model Predictive Control (MPC) for nonlinear models. In the black-box setting, we combine system identification and MPC. Experiments demonstrate the effectiveness of our attacks.

Introduction
Adversarial learning studies vulnerability in machine learning, see e.g. (Vorobeychik and Kantarcioglu 2018; Joseph et al. 2018; Liu et al. 2017; Biggio and Roli 2017; Lowd and Meek 2005). Understanding optimal attacks that might be carried out by an adversary is important, as it prepares us to manage the damage and helps us develop defenses. Time series forecast, specifically autoregressive model, is widely deployed in practice (Hamilton 1994; Box et al. 2015; Fan and Yao 2008) but has not received the attention it deserves from adversarial learning researchers. Adversarial attack in this context means an adversary can subtly perturb a dynamical system at the current time, hence influencing the forecasts about a future time. Prior work (Alfeld, Zhu, and Barford 2016; 2017) did point out vulnerabilities in autoregressive models under very specific attack assumptions. However, it was not clear how to formulate general attacks against autoregressive models.

There are extensive studies on batch adversarial attacks against machine learning algorithms. But there is much less work on sequential attacks. We say an attack is sequential if the attacker take actions over time. There are batch attacks against support vector machine (Biggio, Nelson, and Laskov 2012; Biggio et al. 2014), deep neural networks (Goodfellow, Shlens, and Szegedy 2014; Nguyen, Yosinski, and Clune 2015), differentially-private learners (Ma, Zhu, and Hsu 2019), contextual bandits (Ma et al. 2018), recurrent neural networks (Papernot et al. 2016), online learning (Wang and Chaudhuri 2018) and reinforcement learning (Ma et al. 2019). Some of these victims are sequential during deployment, but they can be trained from batch offline data; hence they can be prone to batch attacks. In contrast, (Jun et al. 2018) and (Zhang and Zhu 2019) study sequential attacks against stochastic bandits and sequential prediction. Our work studies sequential attack against autoregressive model, which is closer to these two papers. Meanwhile, control theory is receiving increasing attention from the adversarial learning community (Recht 2018; Zhu 2018; Lessard, Zhang, and Zhu 2018). Our work strengthens this connection.

This paper makes three main contributions: (1) We present an attack setting where the adversary must determine the attack sequentially. This generalizes the setting of (Alfeld, Zhu, and Barford 2016; 2017), where the adversary can decide the attack after observing all environmental state values used for forecast. (2) We formulate the attacks as an optimal control problem. (3) When the attacker knows the environmental dynamics and forecast model (white-box setting), we solve the optimal attacks with Linear Quadratic Regulator (LQR) for the linear case, or Model Predictive Control (MPC) and iterative LQR (iLQR) for the nonlinear case; when the attacker does not know the environmental or forecaster (black-box setting), we additionally perform system identification.

The Attack Setting

Autoregressive Review
To fix notation, we briefly review time series forecasting using autoregressive models. There are two separate entities:

1. The environment is a fixed dynamical system with scalar-valued states \( x_t \in \mathbb{R} \) at time \( t \). The environment has a (potentially non-linear) \( q \)-th order dynamics \( f \) and is subject to zero-mean noise \( w_t \in \mathbb{R} \) with \( \mathbb{E}(w_t) = \sigma^2 \). Without manipulations from the adversary, the environmental state
We call this the state attack because it changes the underlying environmental states, see Figure 1.

\[ x_{t+1} = f(x_t, \ldots, x_{t-q+1}, u_t) \]  

for \( t = 0, 1, \ldots \). We take the convention that \( x_i = 0 \) if \( i < 0 \). We allow the dynamics \( f \) to be either linear or nonlinear.

The forecaster makes predictions of future environmental states, and will be the victim of the adversary attack. In this paper we mainly focus on a fixed linear AR(1) autoregressive forecaster, regardless of whether the environment dynamics \( f \) is linear or not. Even though we allow the forecast model to be nonlinear in black-box setting, we use linear function to approximate the nonlinear autoregressive model. We also allow the possibility \( p \neq q \). At time \( t \), the forecaster observes \( x_t \) and uses the most recent observations \( x_t, \ldots, x_{t-p+1} \) to forecast the future values of the environmental state. A forecast is made at time \( t \) about a future time \( t' > t \), we use the notation \( y_{t|t} \) to denote it.

Specifically, at time \( t \) the forecaster uses a standard AR(1) model to predict. It initializes by setting \( y_{t+1-i|t} = x_{t+i} \) for \( i = 1 \ldots p \). It then predicts the state at time \( t+1 \) by

\[ y_{t+1|t} = \hat{\alpha}_0 + \sum_{i=1}^{p} \hat{\alpha}_i y_{t+1-i|t}, \]  

where \( \hat{\alpha}_0, \hat{\alpha}_1, \ldots, \hat{\alpha}_p \) are coefficients of the AR(1) model. We allow the AR(1) model to be a nonlinear function in the black-box setting. The AR(1) model may differ from the true environment dynamics \( f \) even if \( f \) is linear: for example, the forecaster may have only obtained an approximate model from a previous learning phase. Once the forecaster predicts \( y_{t+1|t} \), it can plug the predictive value in (2), shift time by one, and predict \( y_{t+2|t} \), and so on. Note all these predictions are made at time \( t \). In the next iteration when the true environmental state evolves to \( x_{t+1} \) and is observed by the forecaster, the forecaster will make predictions \( y_{t+2|t+1}, y_{t+3|t+1}, \) and so on.

### The Attacker

We next introduce a third entity – an adversary (a.k.a. attacker) – who wishes to control the forecaster’s predictions for nefarious purposes. The threat model is characterized by three aspects of the adversary:

(i) Knowledge: In the white-box setting, the attacker knows everything above; in the black-box setting, neither environmental dynamics nor forecaster model are known to the attacker.

(ii) Goal: The adversary wants to force the forecaster’s predictions \( y_{t|t} \) to be close to some given adversarial reference target \( y_{t|t} \) (the dagger is a mnemonic for attack), for selected pairs of \((t, t')\) of interest to the adversary. Furthermore, the adversary wants to achieve this with “small attacks”. These will be made precise below.

(iii) Action: At time \( t \) the adversary can add \( w_t \) to the noise \( u_t \). Together \( u_t \) and \( w_t \) enter the environment dynamics via:

\[ x_{t+1} = f(x_t, \ldots, x_{t-q+1}, u_t + w_t). \]

We call this the state attack because it changes the underlying environmental states, see Figure 1.

---

**White-Box Attack as Optimal Control**

We now present an optimal control formulation for the white-box attack. Following control convention (Lee and Markus 1967; Kwakernaak and Sivan 1972), we rewrite several quantities from the previous section in matrix form, so that we can define the adversary’s state attack problem by the tuple \((F, C, \{y_{|t}'\}, \{Q_{|t}'\}, R, T)\). These quantities are defined below.

We introduce a vector-valued environment state representation (denoted by boldface) \( \mathbf{x}_t := (x_t, \ldots, x_{t-p+1})^\top \in \mathbb{R}^{p+1} \). The first entry 1 serves as an offset for the constant term \( \hat{\alpha}_0 \) in (2). We let \( x_0 \) be the known initial state. It is straightforward to generalize to an initial distribution over \( \mathbf{x}_0 \), which adds an expectation in (9). We rewrite the environment dynamics under adversarial control (3) as \( \mathbf{x}_{t+1} = F(\mathbf{x}_t, u_t, w_t) := (1, f(x_t, \ldots, x_{t-p+1}, u_t + w_t), x_{t+1}, \ldots, x_{t+p+2})^\top. \) If \( f \) is nonlinear, so is \( F \).

In control language, the forecasts are essentially measurements of the current state \( \mathbf{x}_t \). In particular, we introduce a vector of \( p \) predictions made at time \( t \) about time \( t' - p + 1, \ldots, t' \) as \( \mathbf{y}_{t|t} := (1, y_{t|t}, y_{t-1|t}, \ldots, y_{t-p+1|t})^\top \in \mathbb{R}^{p+1}. \) (For completeness, we let \( y_{t|t} = x_t \) when \( t' \leq t \).) The forecaster’s AR(1) forecast model specifies the (linear) measurements as follows. We introduce the \((p+1) \times (p+1)\) measurement (i.e. forecast) matrix \(^1\)

\[ C = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ \hat{\alpha}_0 & \hat{\alpha}_1 & \hat{\alpha}_2 & \cdots & \hat{\alpha}_{p-1} & \hat{\alpha}_p \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix}. \]  

\(^1\)The forecaster usually only has an estimate of \( f \). It is likely that the forecaster’s AR(1) model has a different order \( p \) than the environment dynamics’ order \( q \). For simplicity, we will assume \( p = q \) below, but explain how to handle \( p \neq q \) in Appendix A.
Then the measurements / forecasts are:
\[ y_{t+1} = C^T x_t, \quad \forall t > 0. \tag{5} \]

There are cases that \( y_{t+1} \) does not depend on \( x_t \) but has been decided at time \( t < 0 \). In such cases, we can simply redefine \( y_{t+1} \) to be \( y_{t+1} \) and rewrite the prediction matrix. For simplicity, we assume \( y_{t+1} \) always depends on \( x_t \) in the rest of the paper.

We also vectorize adversarial reference target: \( y_{t+1} = (1, y_{t+1}, \ldots, y_{t+1})^T \in \mathbb{R}^{p+1} \). We simply let \( y_{t+1} = 0 \) when \( t' \leq t \): this is non-essential. In fact, for \( t, t' \) pairs that are uninteresting to the adversary, the target value \( y_{t+1} \) can be undefined as they do not appear in the control cost later.

The cost of the adversary consists of two parts: (1) how closely the adversary can force the forecaster’s predictions to match the adversarial reference targets; (2) how much control \( u_t \) the adversary has to exert. In the simplest case, we let \( u_t \) define the first part, namely the cost of the attacker failing to achieve reference targets. In its simplest form, \( Q_{t+1} \) is a \((p + 1) \times (p + 1)\) matrix with all zero entries except for a scalar weight \( \beta_{t+1} \in [0, 1] \) at (2,2). In this case, \( Q_{t+1} \) simply picks out the \((t', t)\) element:
\[
(y_{t+1}^T y_{t+1}^T) Q_{t+1} (y_{t+1}^T y_{t+1}) = \beta_{t+1} (y_{t+1}^T y_{t+1})^2. \tag{6}
\]

For simplicity, we use \( \| y_{t+1} - y_{t+1}^T \|^2_{Q_{t+1}} \) to denote (6). Critically, by setting the weights \( \beta_{t+1} \) the attacker can express different patterns of attack. For example:

- If \( \beta_{t+1} = 1 \) for all \( t = 0 \ldots T - 1 \) and 0 otherwise, the adversary cares about the forecasts made at all times about the final time horizon \( T \). In this case, it is plausible that the adversarial target \( y_{t+1}^T = y_{t+1}^T \) is a constant w.r.t. \( t \).
- If \( \beta_{t+1} = 1 \) for all \( t \) and 0 otherwise, the adversary cares about all the forecasts about “tomorrow.”
- If \( \beta_{t+1} = 1, \forall t, t' \), the adversary cares about all predictions made at all times.

Obviously, the adversary can express more complex temporal attack patterns. The adversary can also choose \( \beta_{t+1} \) value in between 0 and 1 to indicate weaker importance of certain predictions.

The \( R \) term defines the second part of the adversary cost, namely how much control expenditure \( u_t \) the adversary has to exert. In the simplest case, we let \( R \) be a scalar \( \lambda > 0 \):
\[
\| u_t \|_R^2 := \lambda u_t^2. \tag{7}
\]

We use \( T \) to denote the prediction time horizon: \( T \) is the last time index (expressed by \( t' \)) to be predicted by the forecaster. We define the adversary’s expected quadratic cost \( J(u_0:(T-1)) \) for action sequence \( u_0, \ldots, u_{T-1} \) by:
\[
\mathbb{E}_{u_0:(T-1)} \left[ \sum_{t=1}^{T-1} \sum_{t'=t+1}^T \| y_{t+1} - y_{t+1}^T \|_{Q_{t+1}}^2 + \sum_{t=0}^{T-1} \| u_t \|_R^2 \right] \cdot x_0. \tag{8}
\]

Since the environment dynamics can be stochastic, the adversary must seek attack policies \( \phi_t : \mathbb{R}^{p+1} \mapsto \mathbb{R} \) to map the observed state \( x_T \) to an attack action:
\[ u_t = \phi_t(x_t), \quad \forall t. \tag{9} \]

Given an adversarial state attack problem \((F, C, \{ y_{t+1} \}, \{Q_{t+1}\}, R, T)\), we formulate the optimal state attack as the following optimal control problem:
\[
\min_{\phi_0, \ldots, \phi_{T-1}} J(u_0:(T-1)) \tag{10}
\]

s.t. \( x_0 \) given
\[
u_t = \phi_t(x_t), \quad t = 0, 1, \ldots, T - 1 \tag{11}
\]
\[
x_t + 1 = F(x_t, u_t, w_t), \quad t = 0, 1, \ldots, T - 1 \tag{12}
\]
\[
y_{t+1} = C^T x_t, \quad \forall t' > t. \tag{13}
\]

We next propose solutions to this control problem for linear \( F \) and nonlinear \( F \), respectively. For illustrative purpose, we focus on solving the problem when the attack target is to change predictions for “tomorrows”. This implies \( \beta_{t+1} = 0 \) when \( t' \geq t + 2 \). Under this assumption, \( J(u_0:(T-1)) \) has the following form:
\[
\mathbb{E}_{u_0:(T-1)} \left[ \sum_{t=1}^{T-1} \| Cx_t - y_{t+1}^T \|^2_{Q_{t+1}} + \sum_{t=0}^{T-1} \| u_t \|_R^2 \right] \cdot x_0. \tag{14}
\]

More attack targets are studied in the experiment section.

### Solving Attacks Under Linear \( F \)

When the environment dynamics \( f \) is linear, the scalar environment state evolves as \( x_{t+1} = \alpha_0 + \sum_{t=1}^T \alpha_t x_{t-1} + u_t + w_t \), where the coefficients \( \alpha_0, \ldots, \alpha_p \) in general can be different from the forecaster’s AR(p) model (2). We introduce the corresponding vector operation:
\[
x_{t+1} = F(x_t, u_t, w_t) := A x_t + B(u_t + w_t), \tag{15}
\]
where \( A \) has the same structure as \( C \) in (4) except each \( \alpha \) is replaced by \( \alpha \) and \( B = (0, 1, \ldots, 0) \). The adversary’s attack problem (9) reduces to stochastic Linear Quadratic Regulator (LQR) with tracking, which is a fundamental problem in control theory (Kwakernaak and Sivan 1972). It is well known that such problems have a closed-form solution, though the specific solution for stochastic tracking is often omitted from the literature. In addition, the presence of a forecaster in our case alters the form of the solution. Therefore, for completeness we provide the solution in Algorithm 1.

The derivation is in Appendix. Once the adversarial control policies are computed, the optimal attack sequence is given by:
\[
u_t = \phi_t(x_t), \quad t = 0, 1, \ldots, T - 1 \tag{16}
\]

The astute reader will notice that, \( u_{T-1} = \phi_{T-1}(x_{T-1}) = 0 \). This is to be expected: \( u_{T-1} \) affects \( x_{T-1} \), but \( x_{T-1} \) would only affect forecasts after the prediction time horizon \( T \), which the adversary does not care. To minimize the control expenditure, the adversary’s rational behavior is to set \( u_{T-1} = 0 \).

### Solving Attacks Under Non-Linear \( F \)

When \( f \) is nonlinear the optimal control problem (9) in general does not have a closed-form solution. Instead, we introduce an algorithm that combines Model Predictive Control (MPC) (Garcia, Prett, and Morari 1989; Kouvaritakis...
with the actual noise instantiation. There are two issues that make the problem hard: the expanded model:

\[ \frac{d}{dt} x(t) = A x(t) + B u(t) + w(t), \]

\[ y(t) = C x(t) + D u(t) + v(t). \]

For planning we simply replace the random variables \( w \) with their mean, which is zero in our case. This heuristic removes the expectation, and we are left with the following deterministic system as an approximation to (15):

\[ \min_{u_t} \sum_{t=\tau}^{L(\tau)+1} \|C x_t - y_{t+1}\|_2^2 + \sum_{t=\tau}^{L(\tau)+1} \|u_t\|_R^2 \]

s.t. \( x_t \) given \( x_{t+1} = F(x_t, u_t, 0), t = \tau, \ldots, L(\tau). \) (16)

To address the second issue, we adopt ILQR (Li and Todorov 2004) in order to solve (16). The idea of ILQR is to linearize the system around a trajectory, and compute an improvement to the control sequence using LQR iteratively. We show the details in Appendix. We summarize the MPC+ILQR attack in Algorithm 2 and 3.

Algorithm 1: \( LQR(F, C, \{y_{t\mid \tau}\}, \{Q_{t\mid \tau}\}, R, T) \)

Input : \( F, C, \{y_{t\mid \tau}\}, \{Q_{t\mid \tau}\}, R, T \)

1  \( P_\tau = 0; \)
2  \( Q_\tau = 0; \)
3  for \( t = T-1, T-2, \ldots, 1 \) do
4      \( P_\tau = C A^\top Q_{t+1}\mid t + A^\top (I + 1/\lambda P_{t+1})^{-1} P_{t+1} A; \)
5      \( q_\tau = -2 C^\top Q_{t+1}\mid t y_{t+1\mid t}^\top + A^\top q_{t+1} - \frac{1}{\lambda + B^\top P_{t+1} B} A^\top P_{t+1} B^\top q_{t+1}; \)
6  end
7  for \( t = 0, 1, \ldots, T-1 \) do
8      \( \phi_t(z) = -\frac{B^\top q_{t+1} + 2 P_{t+1} A z}{2(\lambda + B^\top P_{t+1} B)} \)
9  end

Output : \( \phi_0(\cdot), \ldots, \phi_{T-1}(\cdot) \)

Algorithm 2: MPC

Input : \( F, C, \{y_{t\mid \tau}\}, \{Q_{t\mid \tau}\}, R, T, l, \maxiter, \text{tol} \)

1  for \( t = 0, 1, \ldots, T-2 \) do
2  \( u_t \leftarrow \min_{(t+1, T-2)} \) \( \text{ILQR}(x_t, F, C, \{y_{t\mid \tau}\}, \{Q_{t\mid \tau}\}, R, min(t+l+1, T), \maxiter, \text{tol}); \)
3  \( \text{Output} : u_t \)

A Greedy Control Policy as the Baseline State Attack Strategy

The optimal state attack objective (9) can be rewritten as a running sum of instantaneous costs. At time \( t = 0, 1, \ldots, T \) the instantaneous cost involves the adversary’s control expenditure \( u_t^2 \), the attack’s immediate effect on the environment state \( x_{t+1} \) (see Figure 1), and consequently on all the forecaster’s predictions made at time \( t+1 \) about time \( t+2, \ldots, T \). Specifically, the expected instantaneous cost \( g_t(x_t, u_t) \) at time \( t \) is defined as:

\[ \mathbb{E}_{w_t} [\|C F(x_t, u_t, w_t) - y_{t+1}\|_2^2 + \|u_t\|_R^2]. \]

This allows us to define a greedy control policy \( \phi^G \), which is easy to compute and will serve as a baseline for state attacks. In particular, the greedy control policy at time \( t \) minimizes the instantaneous cost:

\[ \phi^G_t(x_t) = \argmin_{u_t} g_t(x_t, u_t). \]

When \( F \) is linear, \( \phi^G_t(\cdot) \) can be obtained in closed-form. We show the solution in Appendix D.

When \( F \) is nonlinear, we let noise \( u_t = 0 \) and solve the following nonlinear problem using numerical solvers:

\[ \min_{u_t} \|C F(x_t, u_t, 0) - y_{t+2\mid t+1}\|^2_{Q_{t+2\mid t+1}} + \|u_t\|_R^2. \]

Black-Box Attack via System Identification

We now consider black-box attack setting where the environment dynamics \( f \) and forecaster’s model \( C \) are no longer
The buffer produces \( t \) environmental dynamics and forecast model. At time \( t \), we use \( \hat{p}_0, \hat{c}_1, \ldots, \hat{c}_p \) to denote the estimation of forecast model and let
\[
C(\hat{c}_{0:p}) = \begin{bmatrix}
1 & 0 & 0 & \cdots & 0 & 0 \\
\hat{c}_0 & \hat{c}_1 & \hat{c}_2 & \cdots & \hat{c}_{p-1} & \hat{c}_p \\
0 & 1 & 0 & \cdots & 0 & 0 \\
0 & 0 & 1 & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & 1 & 0
\end{bmatrix},
\]

to denote the corresponding forecast matrix. Let \( \Lambda(t_1, t_2) \) denote the set of prediction indices which are visible to the attacker: \( \Lambda(t_1, t_2) = \{ (t, t') | t_1 \leq t \leq t_2, y_{t|t'} \text{ is visible to the attacker} \} \). At time \( t \), the forecast model is estimated over the visible forecasts: \( y_{t'|t}, (t, t') \in \Lambda(t-b, t) \):
\[
\min_{\hat{c}_{0:p}} \sum_{(t,t') \in \Lambda(t-b,t)} (B^T C^{t'-t} x_t - y_{t|t'})^2,
\]
we use \( \hat{c}(t, b) \), a \( p+1 \)-dimensional column vector to denote the minimum point of (21). If the attacker only observes sufficient predictions for “tomorrows”, i.e. \( \Lambda(t_1, t_2) = \{ (t, t') | t_1 \leq t \leq t_2, t' = t+1 \} \), then \( \hat{c}_0, \hat{c}_1, \ldots, \hat{c}_p \) is the OLS solution to (21). However, for more complex prediction pattern, (21) might involve polynomials of \( c_0, c_1, \ldots, c_p \).

We can summarize the proposed black-box attack method in Algorithm 4.

Algorithm 3: ILQR
\begin{algorithm}
\caption{ILQR}
\begin{algorithmic}[1]
\Input \: \(x_0, F, C, \{y_{t|t}', \{Q_{t|t}\}, R, T, \maxiter, \tol\)
\State Initialize \( u_{0:T-2} \);
\For {\( i = 0, \ldots, \maxiter \)}
\For {\( t = 0 : T - 2 \)}
\State \( x_{t+1} = F(x_t, u_t, 0) \);
\State \( D_x F_t = D_x F(x_t, u_t, 0) \);
\State \( D_x F_t = D_x F(x_t, u_t, 0) \);
\EndFor
\State \( P_{T-1} = C^T Q_{T|T-1} C \);
\State \( q_{T-1} = 2C^T Q_{T|T-1} (C x_{T-1} - y_{T|T-1}) \);
\For {\( s = T - 2, \ldots, 1 \)}
\State \( P_s = C^T Q_{s+1} s + D_x F^T_s (I + \frac{1}{2} P_{s+1} D_x F D_x F^T s)^{-1} P_{s+1} D_x F^T s \);
\State \( q_s = 2C^T Q_{s+1} s (C x_s - y_{s+1|s}) + D_x F^T_s q_{s+1} + (D_x F^T s + D_u F^T s) (D_x F^T s + D_u F^T s + 2\lambda u_s) \)
\EndFor
\If {\( \|\delta u_{0:T-2}\|^2/(T-1) < \tol \)}
\Break;
\EndIf
\State \( u_{0:T-2} \leftarrow u_{0:T-2} + \delta u_{0:T-2} \);
\EndFor
\EndFor
\end{algorithmic}
\end{algorithm}

The attacks picks an estimation model order \( p \) for both the environment and forecaster. It also picks a buffer length \( b \). In the first \( b + p - 1 \) iterations, the attacker does no attack but collects observations on the free-evolving environment and the forecasts. Then, in each subsequent iteration the attacker estimates a linear environmental model \( \hat{a} \) and linear forecast model \( \hat{c} \) using a rolling buffer of previous \( b + p - 1 \) iterations. The buffer produces \( \hat{b} \) data points for the attacker to use MLE to solve \( p + 1 \) unknowns in environment model. The attacker then uses MPC and LQR to design an attack action.

The attacker use linear models to estimate both the environmental dynamics and forecast model. At time \( t \), we use \( \hat{a}(t, b) \), a \( p+1 \)-dimensional column vector to denote the minimum point of (19). Due to the nature of attacking autoregressive models, \( B = (0, 1, 0, \ldots, 0)^T \in \mathbb{R}^{(p+1) \times 1} \) is always known to the attacker.

Experiments

We now demonstrate the effectiveness of control-based white-box attacks on time series forecast problems. We compare the optimal attacks computed by LQR (for linear \( f \), MPC+ILQR (for nonlinear \( f \)), black-box attack, greedy attacks, and the no-attack baseline. While the attack actions were optimized under an expectation over random noise \( w \) (c.f. (9)), in the
experiments we report the actual realized cost based on the noise instantiation that the algorithm experienced:
\[
\sum_{t=1}^{T-1} \sum_{t'=t+1}^{T} \|C^{t'-t}x_t - y_{t'|t}^f\|_Q^2 + \sum_{t=0}^{T-1} \|u_t\|_R^2
\]
(22)
where the noise sequence \(\{u_t\}_{t=0}^{T-2}\) is incorporated implicitly in \(x_t\), together with the actual attack sequence \(\{u_t\}_{t=0}^{T-1}\). To make the balance between attack effect (the quadratic terms involving \(Q\)) and control expenditure (the term involving \(R\)) more interpretable, we let \(R := \lambda = \lambda \sum_{t=1}^{T-1} \sum_{t'=t+1}^{T} \beta^{|t'|/T}\)

**The Effect of \(Q\) on Attack**

In our first synthetic example we demonstrate the adversary’s ability to target different parts of the forecasts via \(Q\), the quadratic coefficient in cost function. Figure 2 illustrates three choices of attack targets \(Q\): attack “tomorrow”, “last day” and “all predictions”.

![Figure 2: LQR solution on three attack patterns \(Q\). y-axis shows the value of \(x\) and \(u\). Each blue forecast curve beginning at time \(t\) shows the sequence \(\{y_{t'|t-1}\}_{t=1}^{T}\). The pattern of attack defined by the corresponding \(Q\) is highlighted with * on the forecast curves.](image)

For simplicity, we let all adversarial reference targets \(y_{t'|t}^f\) be the constant 1. We let the environment evolve according to an \(AR(1)\) model: \(x_{t+1} = 1 + 0.5x_t + u_t\). We let the noise \(u_t \sim N(0, 0.1^2)\) and the initial state \(x_0 = 0\). We simulate the case where the forecaster has only an approximation of the environment dynamics, and let the forecaster’s model be \(x_{t+1} = 0.9 + 0.6x_t\) which is close to, but different from, the environment dynamics. For illustrative purpose, we set the prediction time horizon \(T = 10\). Recall that the attacker can change the environment state by adding perturbation \(u_t\):

\[x_{t+1} = 1 + 0.5x_t + u_t + u_t\]

We set \(\lambda = 0.1\).

We run LQR and compute the optimal attack sequences \(u\) under each \(Q\) scenario. They are visualized in Figure 2. Each attack is effective: the blue *’s are closer to the green target line on average, compared to where they would be in the upper-left no-attack panel. Different target selection \(Q\) will affect the optimal attack sequence.

**Comparing LQR vs. Greedy Attack Policies**

We now show the LQR attack policy is better than the greedy attack policy. We let the environment evolves by an \(AR(3)\) model: \(x_{t+1} = f(x_t, x_{t-1}, x_{t-2}, u_t) = 0.4x_t - 0.3x_{t-1} - 0.7x_{t-2} + w_t\). \(w_t \sim N(0, 0.1^2)\). The initial values are \(x_0 = 10\), \(x_{-1} = x_{-2} = 0\), prediction horizon \(T = 15\). This environment dynamic is oscillating around 0.

We let the forecaster’s model be: \(x_{t+1} = 0.41x_t - 0.29x_{t-1} - 0.68x_{t-2}\). \(Q\) is “tomorrows”. The attacker wants the forecaster to predict a sequence oscillating with smaller amplitude. \(y_{t+1}^f\) is set as following: we simulate \(x_{t+1} = f(x_t, x_{t-1}, x_{t-2}, 0)\), then, let the attack reference target be \(y_{t+1}^f = 0.5x_t\), \(t = 2, \ldots, T\). We set \(\lambda = 0.1\).

![Figure 3: LQR vs. Greedy attacks. The black horizontal lines in the right plot mark the vertical axis range of the middle plot.](image)

We run LQR and Greedy, respectively, to solve this attacking problem. We generate 50 trials with different noise sequences, see Figure 3. Interestingly, LQR drives the predictions close to the attack target, while Greedy diverges. The mean actual realized cost of no attack, LQR attack and Greedy attack are 133.9, 11.5, 1492, respectively. The standard errors are 1.20, 0.02, 12.99. We perform a paired \(t\)-test on LQR vs. Greedy. The null hypothesis of equal mean is rejected with \(p = 4 \times 10^{-61}\). This clearly demonstrate the myopic failure of the greedy policy.

**MPC+iLQR attack on US Real GNP**

This real world data (Tiao and Tsay 1994) models the growth rate of quarterly US real GNP from the first quarter of 1947 to the first quarter of 1991. We use the GNP data to evaluate MPC+iLQR and Greedy, which attack the “last day.” The environment’s nonlinear threshold model dynamics is:

\[x_{t+1} = \]
We now show where (according to (Tiao and Tsay 1994)). The attacker can change the null hypothesis of equal mean is rejected with the standard errors are. The forecaster’s model is \( x_t = 0 \) (6 respectively. Even closer to \( y_t \), we use the same dynamic in (Fan and Yao 2008) but we change the noise to be \( w_t \sim N(0, 0.1^2) \). The dynamic is \( x_{t+1} = 2x_t/(1 + 0.8x_t^2) + w_t \). We let \( x_0 = 3 \), \( T = 50 \). We simulate this dynamical system to \( t = 50 \) and get a sample sequence from this dynamical system. The forecaster’s AR(1) model \( C \) is estimated from this sequence. We introduce an attacker who can add perturbation \( u_t \) to change the state value: \( x_{t+1} = 2x_t/(1 + 0.8x_t^2) + u_t + w_t \). \( Q \) is “tomorrow”. The attack target is set to be \( y^I_{t+1} = 2 \). We let \( \lambda = 0.01 \).

Both the attacker and the oracle do nothing but observe the \( x_t \) and \( y_{t+1} \) at time \( t = 0, \ldots, b + p - 2 \), and attack the forecast at time \( t = b + p - 1, \ldots, T - 2 \). For system identification, we let \( b = 15 \), \( l = 5 \), \( p = 3 \). For the oracle, the time step of MPC is set to be \( l = 10 \). Inside the MPC loop, the stopping condition of iLQR is \( tol = 10^{-4} \). The maximum iteration of iLQR is set to be 1000.

We run 100 trials, the last one is shown in Figure 5. The mean actual realized cost of system identification attack and oracle MPC+iLQR attack are 4.20, 1.43 respectively. Even though the cost of System identification attack is larger than that of the oracle, it is evident from Figure 5 that the attacker can quickly force the forecasts (blue) to the attack target (green) after \( t = 25 \); the chaotic period between \( t = 20 \) and \( t = 25 \) is the price to pay for system identification.

Using System Identification in Black-box Attack

We now show system identification can perform black-box attack effectively. We compare the system identification attack to an oracle, who has full information of both the environmental dynamics and forecast model. The oracle uses MPC+iLQR to attack the forecast.

We again run 50 trials, the last one is shown in Figure 4. The mean actual realized cost of no attack, MPC+iLQR attack and Greedy attack are \( (6.87, 3.03, 3.23) \times 10^{-3} \) respectively. The standard errors are \( (1.40, 0.18, 0.19) \times 10^{-3} \) respectively. The null hypothesis of equal mean is rejected with \( p = 6 \times 10^{-65} \) by a paired \( t \)-test. As an interesting observation, in the beginning MPC+iLQR adopts a larger attack than Greedy; at time \( t = 4 \), MPC+iLQR adopts a smaller attack than Greedy, but drives \( y_{T|5} \) closer to 0.1. This shows the advantage of looking into future. Since Greedy only focus on current time, it ignores how the attack will affect the future.

We run 50 trials, the last one is shown in Figure 4. The mean actual realized cost of system identification attack and oracle MPC+iLQR attack are 4.20, 1.43 respectively. Even though the cost of System identification attack is larger than that of the oracle, it is evident from Figure 5 that the attacker can quickly force the forecasts (blue) to the attack target (green) after \( t = 25 \); the chaotic period between \( t = 20 \) and \( t = 25 \) is the price to pay for system identification.

Figure 4: MPC+iLQR and Greedy attack on GNP data. MPC+iLQR: \( u_4 = 0.0590, y_{T|5} = 0.0084 \); Greedy: \( u_4 = 0.0719, y_{T|5} = 0.0079 \). \( x^I_{T|5} = 0.01 \).

Figure 5: System identification and MPC+iLQR oracle attack

Conclusion

In this paper we formulated adversarial attacks on autoregressive model as optimal control. This sequential attack problem differs significantly from most of the batch attack work in adversarial machine learning. In the white-box setting, we obtained closed-form LQR solutions when the environment is linear, and good MPC approximations when the environment is nonlinear. In the black-box setting, we propose a method via system identification then perform MPC. We demonstrated their effectiveness on synthetic and real data.

Acknowledgment

We thank Laurent Lessard, Yuzhe Ma, and Xuezhou Zhang for helpful discussions. This work is supported in part by NSF 1836978, 1545481, 1704117, 1623605, 1561512, the MADLab AF Center of Excellence FA9550-18-1-0166, and the University of Wisconsin.
References