Discovery of Precursors to Adverse Events using Time Series Data

Vijay Manikandan Janakiraman* Bryan Matthews† Nikunj Oza‡

Abstract

We develop an algorithm for automatic discovery of precursors in time series data (ADOPT). In a time series setting, a precursor may be considered as any event that precedes and increases the likelihood of an adverse event. In a multivariate time series data, there are exponential number of events which makes a brute force search intractable. ADOPT works by breaking down the problem into two steps - (1) inferring a model of the nominal time series (data without adverse event) by considering the nominal data to be generated by a hidden expert and (2) using the expert's model as a benchmark to evaluate the adverse time series to identify suboptimal events as precursors. For step (1), we use a Markov Decision Process (MDP) framework where value functions and Bellman's optimality are used to infer the expert's actions. For step (2), we define a precursor score to evaluate a given instant of a time series by comparing its utility with that of the expert. Thus, the search for precursors is transformed to a search for sub-optimal action sequences in ADOPT. As an application case study, we use ADOPT to discover precursors to go-around events in commercial flights using real aviation data.

1 Introduction

A precursor may be a pattern or a signature that occurs prior to the adverse event and is associated with an increased likelihood of the adverse event occurring in the future. Precursors are important because adverse events are costly; A flight crash, for example, may kill hundreds of people and cost millions of dollars. A natural disaster such as an earthquake or a tsunami may wipe out an entire city. Adverse events are usually sporadic and have different patterns of failure, making precursors a valuable knowledge base [1] to study and forecast adverse events. The importance of precursor analysis cannot be overemphasized in many domains such as natural calamities [2], extreme weather [3], finance [4], network security [5], etc.

The motivation behind this work is to improve aviation safety by finding precursors to aviation safety incidents. For example, an analyst investigating an incident may identify precursors such as inappropriate wind conditions, unstable descent because of high aircraft speed etc. [6] prior to the incident. Although such events may be identified from recorded data, the process is time consuming. Further, human experts cannot process high dimensional data quickly and efficiently, often finding simple and previously "known" precursors, missing precursors defined by a combination of several variables. The goal of this work is to develop a scalable data mining algorithm to assist the precursor discovery process.

The problem of precursor mining from time series data is not common in the literature and a direct algorithm does not exist. However one can find frequent patterns (or rules) in the adverse time series data [7, 8, 9] that can be identified as precursors or find events that have a causal relationship [10, 11, 12] to the adverse event. For high-dimensional data and continuous variables, such methods are often limited because of the exponential growth in the number of rules [13, 14]. Further, adversities may occur in different patterns that make frequency based mining ineffective as the precursor rule may not have sufficient support to be detected. Another approach to extract precursors in time series [5] is by discretizing the time series and ranking the events based on Granger causality criterion. The method once again may be limited to low dimensions or to time series with binary events or spikes. Motifs (time series features) [3] that discriminate the adverse and nominal time series data may be considered as precursors but suffer the same drawbacks as that of frequency based mining.

A notable limitation with the above methods is the following. A precursor may appear in the nominal data as well and usually followed by some corrective actions. The frequency based rule mining methods consider these events as false positives (belonging to the nominal data) and discard such events. The reason is that an event is evaluated based on its "hard" label that says whether the event belongs to the nominal or adverse data. Our contribution is the development of the algorithm **ADOPT - Automatic Discovery Of Precursors in Time series**, that takes a model-based approach and

*UARC/Nasa Ames Research Center, Moffett Field, CA.
†SGT Inc./Nasa Ames Research Center, Moffett Field, CA.
‡Nasa Ames Research Center, Moffett Field, CA.

uses "soft" information to identify precursors. Instead of searching for precursor events from a set (that grows exponentially with dimension and length of time series), ADOPT converts the problem into a search for suboptimal actions in the adverse time series. By doing so, we naturally find actions that increase the risk of the adverse event. We assume a Markov Decision Process (MDP) where value functions naturally capture long temporal consequences of decision making. This is a key feature of ADOPT, as a precursor may be an event that happened not just one step prior but several steps prior to the adverse event. In contrast to the rule mining methods, our approach can detect an event as a precursor although it may occur in the nominal data. This is because an event is evaluated using "soft" labels; i.e., based on rewards and values, instead of using the "hard" label that indicates if the event is present in the nominal data or the adverse data. Further, by modeling the value function in a parametric form, we hope to achieve better scalability to operate with continuous and high-dimensional data. The parametric model may also offer better generalization performance.

The remainder of the paper is organized as follows. Section 2 formally introduces the precursor discovery problem and describes the steps involved in the ADOPT algorithm. Section 3 discusses the application of ADOPT to find precursors to an aviation safety incident using real aviation data followed by concluding remarks in Section 4.

2 Methodology

Let the adverse event in consideration be denoted by E_A. Consider a database $\mathcal{D} = \{X_i, Y_i\}_{i=1}^{D}$ where X_i represents a time series record and Y_i represents a label taking binary values corresponding to X_i being nominal or adverse. X_i is nominal (and $Y_i = 0$) if E_A does not occur in its episode. Let the databases containing nominal records and adverse records be denoted by $\mathcal{N}$ and $\overline{\mathcal{N}}$ respectively. Then the number of nominal and adverse records in $\mathcal{D}$ can be denoted by $|\mathcal{N}|$ and $|\overline{\mathcal{N}}|$ ($D = |\mathcal{N}| + |\overline{\mathcal{N}}|$) respectively. A time series record X is a collection of time-indexed observations of d variables and of episode length[1] L, i.e.,

$$X = \begin{bmatrix} x^1(1) & x^1(2) & . & x^1(L) \\ x^2(1) & x^2(2) & . & x^2(L) \\ . & . & . & . \\ x^d(1) & x^d(2) & . & x^d(L) \end{bmatrix}$$

For instance, X may be a flight where d sensory variables[2] such as velocity, altitude, etc. are measured at regular sampling intervals. The nominal records are episodes that do not have an adverse event, whereas adverse records are episodes that have the adverse event.

We make the following assumptions in ADOPT - (1) The labels Y_i are available, (2) the data is available in a time series format with uniform sampling, (3) there is a hidden expert who makes decisions at every instant of the nominal time series and (4) the expert's behavior is optimal in the context of avoiding the adverse event. The third assumption is true without the loss of generality to problems where there is no explicit decision making in the data, e.g. earthquake monitoring data. In such cases, the expert may be modeled as the stochastic process that generates the nominal time series data. The adverse time series on the other hand is generated by a different process favouring the adversity. We define a precursor in terms of expert decision making as follows.

DEFINITION 2.1. Precursor: A precursor $P_{E_A} = (p_k, k)$ to an adverse event E_A is an inferior decision at time k (prior to E_A) that is associated with an increased likelihood of the adverse event.

It should be noted that the precursor is an event in the time series whose label is unknown.

2.1 ADOPT Overview The ADOPT algorithm (see Figure 1 for ADOPT framework) approaches the precursor discovery problem from a decision making perspective. The time series data is considered to be sampled from an MDP where a hidden agent makes a decision at every instant in time and the quality of decisions correlate with the outcome of the time series; i.e., the nominal time series contains evidence on how an expert agent would make decisions so that an adverse event is avoided. The adverse time series on the other hand, contains evidence on how a non-expert makes suboptimal decisions that increase the likelihood of the adverse event. This requires a definition of optimality in decision making. We infer this using time series data as follows.

First we determine the underlying reward function of the expert using an inverse reinforcement learning (IRL) algorithm in Section 2.2.1 where nominal and adverse time series data are used as trajectory demonstrations of the expert and non-expert respectively. Using the reward function and considering nominal time series as Monte Carlo samples of the expert policy, we

[1] In the adverse time series, the adverse event occurs at the $(L+1)^{th}$ time step and we consider data corresponding to the past L time steps.

[2] Sensory variables are also referred to as time series variables and state variables in this paper.

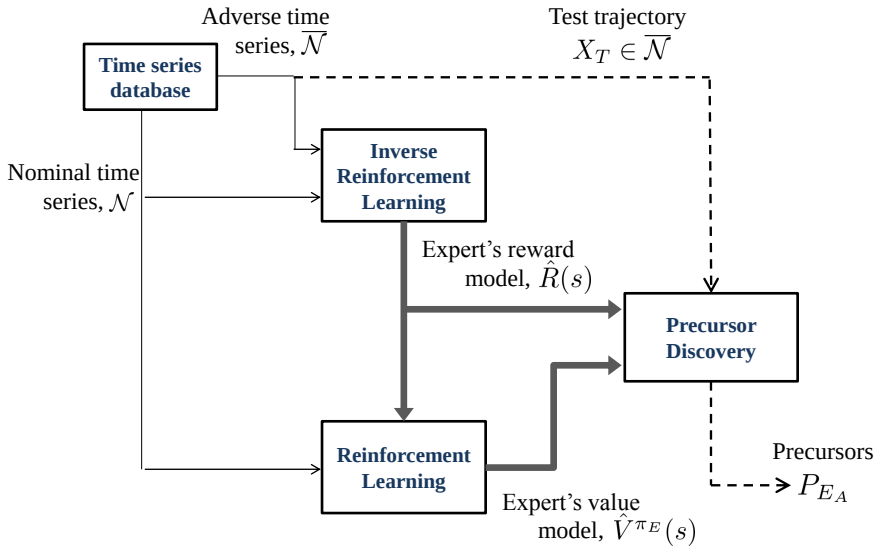

Figure 1: ADOPT framework showing the steps involved in the precursor discovery process.

model the expert's value function using reinforcement learning (RL) in Section 2.2.2. The reward function represents an instantaneous consequence of a decision while the value function indicates a consequence that is long-term. The reward model and value model of the expert is used in the precursor discovery step, where for a given time series (test data), we determine a precursor strength for every instant of the time series by comparing the decision made at that instant to that of an optimal agent. Every time instant of the adverse time series correspond to a decision made by a suboptimal agent. For each of these instants, there exist an optimal action that may reduce the likelihood of the adverse event, which can be identified using Bellman's optimality. The value function is used to compare the suboptimality of a non-expert's action against that of an expert and a precursor strength is obtained. Finally, by fixing a suitable threshold for the precursor strength, precursors are identified in Section 2.3 in an unsupervised manner.

2.2 Modeling Expert's Behavior

2.2.1 Estimating Expert's Reward function using Inverse Reinforcement Learning (IRL) The goal of inverse reinforcement learning (IRL) is to determine the underlying reward function using observed behavior of the agent making decisions in a Markov Decision Process (MDP).

Following a setup similar to [15], an MDP is a tuple $(\mathcal{S},\ \mathcal{A},\ P_{s,a},\ \gamma,\ R)$ where $\mathcal{S}$ is a continuous state space with d state variables. Every state $s \in \mathcal{S}$ can be represented by a vector in $\mathbb{R}^d$ as $[s^1, s^2, .., s^d]^T$. Similarly, $\mathcal{A}$ is a continuous action space with l action variables. $\{P_{s,a}\}$ (or $P_{ss'}$ if actions are undefined) are the state transition probabilities corresponding to an action a at state s. $\gamma \in [0, 1)$ represents the discount factor and $R : \mathcal{S} \to \mathbb{R}$ is the underlying reward function. A policy π can be defined as any map $\pi : \mathcal{S} \mapsto \mathcal{A}$ specifying an action a at every state s. The value function of a policy π evaluated at a state s_1 can be given by

$$V^{\pi}(s_0) = E_{\pi}[R(s_0) + \gamma R(s_1) + ... + \gamma^L R(s_L)|\pi] \tag{2.1}$$

where the expectation is over the distribution of state sequences $(s_0, s_1, s_2, ..., s_L); s_i \in \mathbb{R}^d$ generated by following the policy π starting from s_0 [15, 16]. Assuming availability of sampled trajectories (time series collections in $\mathcal{N}$ and $\overline{\mathcal{N}}$), the IRL problem can be posed as in [15]. The sampled trajectories can be considered as demonstrations of both the expert and non-expert acting in the MDP. For infinite dimension problems (i.e., with continuous state variables as in our case), the unknown reward function can be parameterized [15] as

$$R(s; \boldsymbol{\alpha}) = \alpha_1\phi_1(s) + \alpha_2\phi_2(s) + .. + \alpha_m\phi_m(s), \tag{2.2}$$

where $\boldsymbol{\alpha} = [\alpha_1 \quad \alpha_2 \quad ... \quad \alpha_m]^T$ is the unknown parameter vector, $\phi_i(s); i = 1, 2, .., m$ represent the basis functions of the reward model. The expert's value function determined using the parametric reward, following pol-

icy π_E at state s_0 can be given by

$$\begin{aligned} V^{\pi_E}(s_0;\boldsymbol{\alpha}) &= E[R(s_0)+\gamma R(s_1)+...+\gamma^L R(s_L)|\pi_E] \\ &= E[\sum_{i=1}^{m}\alpha_i\phi_i(s_0)+... \\ &\quad +\gamma^L\sum_{i=1}^{m}\alpha_i\phi_i(s_L)|\pi_E] \\ &= E[\sum_{i=1}^{m}\{\alpha_i\sum_{j=0}^{L}\gamma^j\phi_i(s_j)\}|\pi_E] \\ &= \sum_{i=1}^{m}\alpha_i E[\sum_{j=0}^{L}\gamma^j\phi_i(s_j)|\pi_E] \\ &= \sum_{i=1}^{m}\alpha_i\lambda_i \end{aligned} \tag{2.3}$$

where $\lambda_i = E[\sum_{j=0}^{L}\gamma^j\phi_i(s_j)|\pi_E]$ represent the feature expectations; i.e., the value function if the reward function is composed of $\phi_i(s)$ only. Similarly, by knowing the sequence of states (trajectories) for non-expert/non-optimal policies, the $V^{\pi_{adv}}(s_0;\boldsymbol{\alpha})$ can be calculated in terms of $\boldsymbol{\alpha}$. Our objective is to determine the coefficients α_i so that $E_{s_0}[V^{\pi_E}(s_0;\boldsymbol{\alpha})] \geq E_{s_0}[V^{\pi_{adv}}(s_0;\boldsymbol{\alpha})]$; i.e., to have the expert's state values to be higher than that of a non-expert. By doing so, the estimated model assigns a higher value for the actions taken in the nominal time series and lowers the value for suboptimal actions taken in the adverse time series. A linear programming problem [15] can be solved for $\boldsymbol{\alpha}$ as follows

$$\min_{\alpha}\{E_{s_0}[V^{\pi_{adv}}(s_0;\boldsymbol{\alpha})] - E_{s_0}[V^{\pi_E}(s_0;\boldsymbol{\alpha})]\} \tag{2.4}$$

$$\text{s.t. } |\alpha_i| \leq 1, i = 1,1,..,m \tag{2.5}$$

The estimated reward model is given as follows

$$\hat{R}(s) = f_R(s;\boldsymbol{\alpha}^*) = \sum_{i=1}^{m}\alpha_i^*\phi_i(s), \tag{2.6}$$

where $\boldsymbol{\alpha}^*$ is the optimal solution of the above linear programming problem. By having a parameterized model for the expert's reward function, we expect the model to generalize well and scale better to high dimensional problems. Using the flight labels Y that correspond to a flight being nominal or adverse, cross validation can be performed by having a hold-out set and the model hyper-parameters can be optimized. Using a generalizing reward model, the expert's value function can be estimated as follows.

2.2.2 Estimating Expert's Value Function using Reinforcement Learning (RL) In ADOPT, a model of the expert's value function[3] is estimated using the estimated reward model and the nominal time series data. We consider the collection of nominal time series as Monte Carlo samples from the expert's policy (π_E) and determine the value function using the Monte Carlo (MC) method [16]. This fits well in our setup as the MC method requires only the policy demonstrations (time series data in our case) without any prior knowledge of the environment's dynamics [16] such as a full transition probability matrix. A parametric model of the value function can be developed as follows. For every sample of the expert's policy π_E, i.e., for every episode in $\mathcal{N}$, the first visit Monte Carlo return $Ret(s)$ for each state s is determined using the estimated reward model $\hat{R}(s)$. For every labeled pair $(s_i, Ret(s_i))$, a regression model $\hat{V}^{\pi_E}(s;\theta)$ parameterized by θ can be built by solving the following optimization problem

$$\min_{\theta}\frac{1}{N_s}\sum_{i=1}^{N_s}\|Ret(s_i) - \hat{V}^{\pi_E}(s_i;\theta)\|^2 + \frac{\mu}{2}\|\theta\|^2, \tag{2.7}$$

$$\hat{V}^{\pi_E}(s) = f_V(s;\theta^*) \tag{2.8}$$

where N_s and μ represent the number of labeled data samples and regularization coefficient respectively. Standard parametric models such as linear regression, artificial neural networks, etc. can be used to approximate the value function. The suitability of function approximation models and its convergence properties are well analyzed in literature [16, 17]. In addition to making value estimation feasible for continuous and high-dimensional state spaces, function approximation also improves the generalization of the value function model to unseen state spaces.

2.2.3 Precursors as Suboptimal Decision Events As mentioned earlier, the adverse time series is considered a demonstration by a non-expert whereas the nominal time series is considered a demonstration by an expert who makes optimal decisions to avoid the adverse event. For precursor discovery, we look for prior events in the adverse time series where there is a deviation from the nominal behavior. The analysis

[3] Since the paper considers time series in a general sense where action variables are not always easy to identify, the term 'value function' refers to the state value function in this paper. For the case where action variables are well defined, the algorithm can be modified where state value function can be replaced by state-action value function [16].

begins by asking the following question

(2.9)

"For every instant s_k of the adverse time series $\bar{X}$, what is the expert's optimal action and how suboptimal is the action taken by the non-expert in $\bar{X}$?"

The first part of the question involves finding the optimal action made by the expert, which can be determined using the expert's value function and Bellman's optimality principle [16], while the second part of the question involves finding a means to compare actions. With access to the optimal value function ($V^{\pi_E}(s)$), starting from state s_k, one can search for all possible actions (or state transitions) from s_k and identify the one that corresponds to the maximum $V^{\pi_E}(s_{k+1})$ [16]. By doing the above search, the best action to avoid the adverse event can be identified for every instant of the adverse time series. This answers the first part of the question in (2.9).

In the second part of the question, we are interested in evaluating the actions executed by the non-expert in the adverse time series. For this, the difference between the values corresponding to the optimal action and the suboptimal action is used. When the difference is large, the action taken by the non-expert favors the adverse event more. Thus, at every instant of the adverse series, (1) the value corresponding to the executed suboptimal action can be determined, (2) an optimal action and the corresponding value function using Bellman's optimality can be determined, and (3) the drop in value can be determined. The drop in value is considered for subsequent calculation of the the precursor strength. Some implementation details of ADOPT are discussed in the following section.

2.3 Discovering Precursors using ADOPT In this subsection, specifics on evaluating a test time series for precursors are discussed. In order to handle the situation where an action set $\mathcal{A}$ is unknown (not observed in data) or undefined (no clear decision maker such as in earthquake monitoring data, where only the magnitude of the quake is monitored without any notion of actions), ADOPT defines an abstracted action as follows[4].

Definition 2.2. Abstracted Actions: An abstracted action at time k is defined as an action a_k^{abst} that transitions a state of an MDP from s_k to s_{k+1} following the same transition probability of the original MDP. Let the set of possible abstracted actions at state s_k be denoted by a restricted action space $\mathcal{A}_R^{s_k}$.

Definition 2.3. Reachable States: Reachable states from a state s_k include states obtained by taking all possible abstracted actions $a_k^{abst} \in \mathcal{A}^{s_k}$ at state s_k. The set of reachable states from s_k can be denoted by $S_R^{s_k}$.

With knowledge of the state transition probabilities in time series data, $P_{ss'}$, it is relatively easier to define $S_R^{s_k}$ than $\mathcal{A}_R^{s_k}$. The concepts of abstracted actions and reachable states help in converting the behavior of time series data into actionable sequences where each action can be considered as a candidate precursor. A loss in utility and precursor index can be defined to quantitatively evaluate the candidates.

Definition 2.4. Loss in Expected Utility: The Loss in Expected Utility (LiEU) at an instant k in a time series episode can be defined as the loss in utility by taking an action a_k as against an optimal action a_k^*, i.e., $LiEU_k = E\{U(s_k, a_k^*)\} - E\{U(s_k, a_k)\}$, where $U(s_k, a_k)$ represents the utility of taking action a_k at state s_k and E{.} is an expectation operator.

In terms of value function, $LiEU$ can be expressed as

$$\begin{aligned} LiEU_k &= E\{U(s_k, a_k^*)\} - E\{U(s_k, a_k)\} \\ &= R(s_k) + V^{\pi_E}(s_{k+1}^*) - R(s_k) - V^{\pi_E}(s_{k+1}) \\ &= V^{\pi_E}(s_{k+1}^*) - V^{\pi_E}(s_{k+1}), \end{aligned} \tag{2.10}$$

where s_{k+1}^* represents the state obtained by taking an optimal action a_k^*. The LiEU can be thought of as a score to measure the level of sub-optimality of an action a_k with respect to an optimal action a_k^* at a given state s_k.

Definition 2.5. Precursor Index: A precursor index (PI) at an instant k in a time series episode can be defined as the weighted average of the LiEU score and the relative instantaneous utility (the state reward) R_k.

The precursor index is given by

$$PI_k = w_1 LiEU_k + w_2(1 - R_k), \tag{2.11}$$

where $w_1 + w_2 = 1$, w_1 and w_2 are the weights trading off the "long-term" expected utility (state value) and the "immediate" utility (state reward). The precursors to an adverse event could have an influence on the adverse-event both over "long-term" as well as "short-term" in the trajectory. The importance of "long-term" versus "short-term" could be adjusted using the weight parameters. It should be noted that both $LiEU$ and R are normalized to lie between 0 and 1 for a given test flight. The ADOPT algorithm can be summarized below.

[4] A simple abstraction could be $a \in$ {increase, maintain, decrease} for a given state variable. In cases where actions are well defined and observed, our method proceeds with the known actions of the system.

2.3.1 ADOPT Algorithm The pseudocode of ADOPT is shown in Algorithm 1. The algorithm begins by taking time series data corresponding to nominal and adverse events. The expert's reward model $\hat{R}(s)$ is estimated using IRL while the expert's value model $\hat{V}^{\pi_E}(s)$ is estimated using RL. These models are given as inputs along with a test trajectory $X_T \in \overline{\mathcal{N}}$ to the precursor analysis function in Step 3. For each state transition s_k to s_{k+1} in the test trajectory[5] X_T, the utility V_k^T of the suboptimal action (or state transition) at k is determined using the expert's value model $\hat{V}^{\pi_E}$ evaluated at (s_{k+1}). The reason for evaluating at s_{k+1} is to determine the utility of the transition from s_k to s_{k+1}. Using the data history (or $P_{ss'}$ if known), the set of reachable states $S_R^{s_k}$ is determined. This represents the set of all possible states the agent can achieve including the optimal state s_{k+1}^* as well as the trajectory's next state s_{k+1}. The values corresponding to all such transitions $V_k^{S_R}$ using the expert's value model is determined. Using Bellman's optimality, the state transition (or action) corresponding to the maximum value V_k^* is identified. The loss in utility $LiEU_k$ of the state transition s_k to s_{k+1} can be compared to an optimal transition s_k to s_{k+1}^* and is calculated as the difference between the values of the two transitions. The $LiEU_k$ is weighted averaged with the instantaneous reward R_k^T and a precursor index PI_k is calculated for every state transition. A threshold[6] δ_P that sets the strength of identified precursors is defined, using which the set of precursors $\mathcal{P}_{E_A}$ for the episode X_T is obtained.

The main computational component of ADOPT algorithm is the calculation of the value trajectories corresponding to the nominal and adverse time series (as in equation (2.3)) and the feature expectations $\lambda_i; i = 1, 2, .., m$. Roughly, the runtime scales in the order of the number of basis functions m and the length of time series L, but independent of the dimension of the time series. Although the number of basis functions typically increase with dimension, if the intrinsic dimension of the data is low, then ADOPT can efficiently handle it. As the value trajectory for each time series record can be determined independently, it is straightforward to use distributed processing when dealing with a large number of time series records. Another means to scale the algorithm to large data sets is to update the parametric models sequentially by processing time series records one by one or in chunks.

[5] T as a subscript or superscript indicates the test trajectory.

[6] If the threshold is set to be low, then we expect the algorithm to find many weak precursors and possibly with a high false positive rate. On the other hand, if δ_P is set to be very high, we expect that some precursors may be missed.

Algorithm 1: ADOPT Pseudocode

Data: Time series data from nominal database $\mathcal{N}$ and adverse database $\overline{\mathcal{N}}$.
Result: Precursors $\mathcal{P}_{E_A}$ corresponding to $X_a \in \overline{\mathcal{N}}$.
Step 1. Estimation of Expert's Reward function
Input : Time series data from nominal database $\mathcal{N}$ and adverse database $\overline{\mathcal{N}}$.
for $j \leftarrow 1$ **to** $|\mathcal{N}|$ **do**
  $V_j^{nom}(s_0; \boldsymbol{\alpha}) \leftarrow \sum_{i=1}^{d} \alpha_i \lambda_i^j$;
end
for $j \leftarrow 1$ **to** $|\overline{\mathcal{N}}|$ **do**
  $V_j^{adv}(s_0; \boldsymbol{\alpha}) \leftarrow \sum_{i=1}^{d} \alpha_i \lambda_i^j$;
end
$V^{\pi_E}(\boldsymbol{\alpha}) \leftarrow mean(V_j^{nom}(s_0; \boldsymbol{\alpha})), j = 1, 2, ..., |\mathcal{N}|$;
$V^{\pi_{adv}}(\boldsymbol{\alpha}) \leftarrow mean(V_j^{adv}(s_0; \boldsymbol{\alpha})), j = 1, 2, ..., |\overline{\mathcal{N}}|$;
$\boldsymbol{\alpha^*} \leftarrow \arg\min_{\alpha}\{V^{\pi_{adv}}(\boldsymbol{\alpha}) - V^{\pi_E}(\boldsymbol{\alpha})\}$ subject to $|\alpha_i| \leq 1, i = 1, 1, .., m$;
Output: Expert's Reward model $\hat{R}(s) = f_R(\boldsymbol{\alpha^*})$.
Step 2. Estimation of Expert's Value function
Input : Expert's demonstration X_i, $i = 1, 2, ..., |\mathcal{N}|$, Expert's Reward model $\hat{R}(s)$.
for $i \leftarrow 1$ **to** $|\mathcal{N}|$ **do**
  for $k \leftarrow 1$ **to** L_i **do**
    $Ret(s_k) \leftarrow E\{\sum_{p=k}^{L_i} \gamma^{p-1} \hat{R}(s_p)\}$;
  end
end
$\theta^* \leftarrow \arg\min_{\theta} \frac{1}{N_s} \sum_{i=1}^{N_s} \|Ret(s_i) - \hat{V}^{\pi_E}(s_i; \theta)\|^2 + \frac{\mu}{2}\|\theta\|^2$;
Output: Expert's value model $\hat{V}^{\pi_E}(s) = f_V(\theta^*)$.
Step 3. Discovery of Precursors
Input : Expert's reward model $\hat{R}(s)$, Expert's value model $\hat{V}^{\pi_E}(s)$, test trajectory $X_T \in \overline{\mathcal{N}}$.
for $k \leftarrow 1$ **to** L_T **do**
  $V_k^T \leftarrow \hat{V}^{\pi_E}(s_{k+1})$;
  Determine $S_R^{s_k}$ using state transition matrix $P_{ss'}$;
  $V_k^{S_R}(s_i) \leftarrow \hat{V}^{\pi_E}(s_i), \forall s_i \in S_R^{s_k}$;
  Perform Bellman's optimality principle (greedy search), $V_k^* = \max_{s_i}\{V^{S_R^{s_k}}\}$;
  $LiEU_k \leftarrow V_k^* - V_k^T$;
  $R_k^T \leftarrow \hat{R}(s_k)$;
  $PI_k \leftarrow w_1 LiEU_k + w_2(1 - R_k^T)$;
end
$\mathcal{P}_{E_A} \leftarrow \{PI_k | PI_k > \delta_P\}$;
Output: PI_k and $\mathcal{P}_{E_A}$.

3 Aviation Safety Application

In this section we discuss application of ADOPT algorithm to discover precursors to go-arounds[7] using aviation data. Each data record is in the form of a time series consisting of aircraft trajectory data (latitude, longitude, altitude, ground speed), landing airport data (runway configuration, counts and rates of departure and arrival, total air and taxi delays, meteorological data). Additionally, features derived from radar track and weather data such as the headwind components, altitude in feet above ground level (AGL), horizontal/vertical distance between aircraft, and the corresponding closing rates to the nearest aircraft were included. The number of variables in each time series record is 41 with an average length in the order of about 250 time steps. With about 1000 flight records, our data matrix is sized 41x250000.

About 500 go-around flights and 500 nominal (no go-around) flights were selected from the database of flights landed at Dallas Ft. Worth International Airport. From this, about 400 flights from each set were used for training ADOPT's reward function using IRL (in Section 2.2.1). The remaining 100 flights in each set were used for tuning the model hyper-parameters based on cross-validation. We used gaussian functions as the basis functions for the reward model which require tuning of hyper-parameters such as the number of basis functions (m) and the spread of the gaussian functions[8]. The hyper-parameters were tuned based on cross validation and evaluated using the hold-out set. In our experiments we normalized the data to lie between 0 and 1. We used a grid search and identified that a total of 5000 gaussian functions with a spread of 0.05 worked best. For the value estimation step (in Section 2.2.2), we used a linear regression model to estimate the expert's value function. Using the estimated models, we proceed to the precursor discovery step (in Section 2.3) and the procedure outlined in Section 2.3.1 is followed. We parallelized the algorithm and used several cluster nodes to speed up the cross-validation and training in both the IRL step and value estimation step.

Our cross-validation in the IRL step ensures that we have a model that generalizes well. As the precursor discovery is unsupervised (a priori labels for precursors unknown), we validate our base model (model of the expert) using labels of the adverse events (Y_i). We note that the training and generalization (on unseen data) errors are -55.42 and -10.32 respectively. The errors correspond to equation (2.4) and a lower value indicates a better model. More importantly, the errors being negative means that the model assigns a higher value to the expert demonstration compared to the non-expert which is desirable. With this model, we proceed to the unsupervised precursor discovery step. From a separate set of unseen flight data, we selected two go-around flights[9] and two nominal flights for analysis by ADOPT. A former commercial airline captain was utilized as a subject matter expert (SME) and asked to analyze these flights. The analysis is compared qualitatively with the precursors identified by ADOPT.

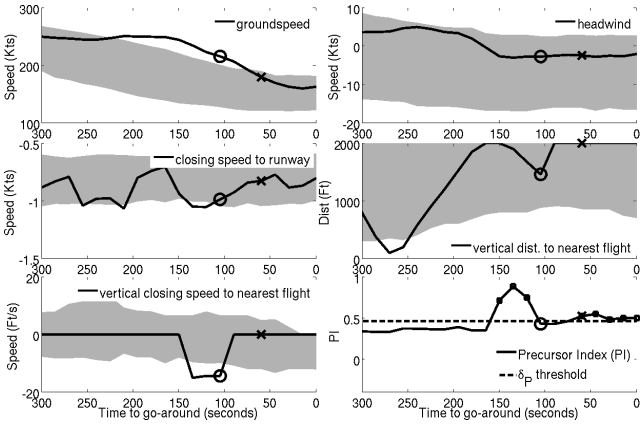

Figure 2: Energy Mis-Management Scenario case: Flight variables of interest along with ADOPT's precursor index (PI) are plotted against time before the go-around event. Negative values for closing rates correspond with converging aircrafts. The shaded regions corresponds to a nominal data distribution (10 to 90 percentile). The 'O' indicates when the aircraft crossed over the outer marker which is approximately 5 mi before the runway threshold. The 'X' represents when the aircraft was closest to 1,000 ft above ground at which flights are required to be stable on final approach.

3.1 Analysis of Go-Around Flights

3.1.1 Energy Mis-Management Scenario The flight in this example (see Figure 2) executed a go-around because of high speed prior to landing. It can be seen that between 160 and 130 seconds and around 50 seconds prior to the go-around, the PI value is increasing significantly and at the same time there is a significantly high ground speed above the shaded band (nominal data distribution) indicating a possible precursor to the go-around. The SME confirmed that the flight's high speed around the two markers (outer marker and 1000 ft al-

[7] A go-around is defined as a flight path executed by an aircraft after an aborted landing attempt to avoid losing safety.

[8] All gaussian functions have the same spread.

[9] It takes a couple of hours for a human expert to fully review a given flight for precursors

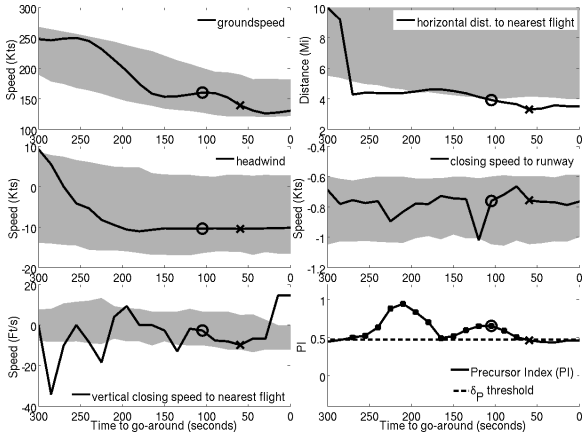

Figure 3: Potential Overtake Scenario case 1: Flight variables of interest along with ADOPT's PI values are plotted against time before the go-around. Please follow caption from Figure 2 for easy interpretation.

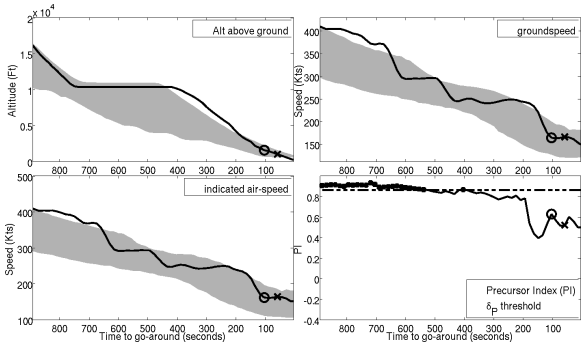

Figure 4: Nominal Flight 1 with a precursor but risk mitigated which avoided a go-around. Please follow caption from Figure 2 for easy interpretation.

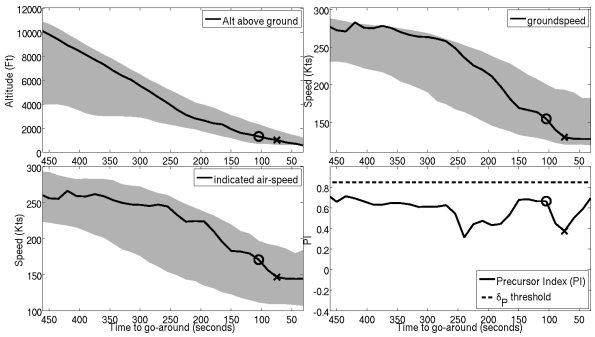

Figure 5: Nominal Flight 2 with no precursor. Please follow caption from Figure 2 for easy interpretation.

titude marker) could be a main reason behind the go-around. ADOPT was able to identify these points and even time instants prior to these as precursors.

3.1.2 Potential Overtake Scenario The flight in this example (see Figure 3) executed a go-around because of the presence of another flight in the front causing a potential overtake scenario (possible conflict in landing sequence). It can be seen that between 270 and 210 seconds prior to go-around, and around the outer marker, there is an increasingly high PI value. This corresponds to the region where the horizontal distance to the nearest aircraft is lower than nominal and the vertical closing rate to the nearest aircraft is negative while also dipping below the nominal values indicating that there may be an unsafe overtake situation. To avoid this, the flight performed a go-around. The SME confirmed the precursors and the time instances.

It should be noted that the SME analysis was done completely independent of ADOPT's results.

3.2 Analysis of Nominal Flights Looking for precursors in a nominal flight (flight without a go-around) is meaningless as we know that the adverse event did not occur (from available label $Y_i = 0$). However, the nominal flight may have a precursor whose risk is mitigated quickly prior to flight landing. Such examples give insights into the expert's corrective actions. For instance, ADOPT is used to analyze a flight that had a high speed and/or altitude from about 800 to 300 seconds prior to landing (see Figure 4). However, the speed and altitude are reduced satisfactorily prior to the 1000 ft altitude marker (by when the aircraft must achieve nominal speed and altitude to prevent a go-around) and thus the flight lands without a go-around. The PI score of ADOPT explains this scenario well. Around the same time when the speed and/or altitude is high, the precursor score is also high indicating precursors but subsequently the PI score reduced indicating a corrective action by the pilot. In another example of a nominal flight (see Figure 5), the aircraft speed and altitude always lies within the nominal region and correspondingly, the PI score is always lower than the threshold indicating an absence of precursor alarms.

Although ADOPT's results are promising, the above evaluations do not give a complete picture on the performance of ADOPT. Both false alarms and missed detections are to be expected from ADOPT as precursors are detected based on the precursor index crossing a threshold. Similar to unsupervised learning methods such as clustering or anomaly detection, the threshold can only be set to discover precursors for a given strength. Setting the threshold too low may have several false positives while setting it too high may have a high missed detection. Unless one has ground truth labels for precursor events, finding an optimal threshold

is not possible. The aviation data analyzed here has no ground truth and constructing the ground truth labels for precursors is not a trivial task. Thus we report qualitative results. The quality of the underlying decision model (reward model) has been evaluated quantitatively using a hold out set and the error rates are as reported in the beginning of Section 3. It should be noted that no benchmark precursor data (data that has labels for precursors to an adverse event) exist in the literature and so a quantitative evaluation or a baseline comparison cannot be made at this time. We are working on creating a benchmark data set for precursor discovery and further evaluations will be done in the future.

4 Conclusions

In summary, we developed ADOPT, a scalable algorithm to discover precursors in multivariate time series data. ADOPT takes a model-based approach and captures the underlying behavior of the time series using value functions. We demonstrated ADOPT's working using real aviation data where ADOPT's results are promising. Our future work includes constructing an artificial benchmark data set that has ground truth labels for precursors and performing a quantitative evaluation of ADOPT.

References

[1] M. Cheok, "Accident sequence precursor program," 2004. [Online]. Available: http://pbadupws.nrc.gov/docs/ML0425/ML042540242.pdf

[2] J. J. McGuire, P. F. Ihmle, and T. H. Jordan, "Time-domain observations of a slow precursor to the 1994 romanche transform earthquake," *Science*, vol. 274, no. 5284, pp. 82–85, 1996.

[3] A. McGovern, D. Rosendahl, R. Brown, and K. Droegemeier, "Identifying predictive multi-dimensional time series motifs: an application to severe weather prediction," *Data Mining and Knowledge Discovery*, vol. 22, no. 1-2, pp. 232–258, 2011.

[4] N. Vandewalle, M. Ausloos, P. Boveroux, and A. Minguet, "Visualizing the log-periodic pattern before crashes," *The European Physical Journal B - Condensed Matter and Complex Systems*, vol. 9, no. 2, pp. 355–359, 1999.

[5] J. B. D. Cabrera and R. K. Mehra, "Extracting Precursor Rules from Time Series: A Classical Statistical Viewpoint," in *SIAM International Conference on Data Mining*, 2002.

[6] "Aviation safety network investigation report: Serious runway excursion incident following unstabilized approach," 2011. [Online]. Available: http://news.aviation-safety.net/2011/01/10/report-serious-runway-excursion-incident-following-unstabilized-approach

[7] S. Laxman and P. Sastry, "A survey of temporal data mining," *Sadhana*, vol. 31, no. 2, pp. 173–198, 2006.

[8] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Proceedings of the Eleventh International Conference on Data Engineering*, ser. ICDE '95. Washington, DC, USA: IEEE Computer Society, 1995, pp. 3–14.

[9] G. Das, K. ip Lin, H. Mannila, G. Renganathan, and P. Smyth, "Rule discovery from time series," in *KDD-98*. AAAI Press, 1998, pp. 16–22.

[10] K. Karimi and H. Hamilton, "Discovering temporal/causal rules: A comparison of methods," in *Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science, Y. Xiang and B. Chaib-draa, Eds. Springer Berlin Heidelberg, 2003, vol. 2671, pp. 175–189.

[11] S. Kleinberg, "Causal inference with rare events in large-scale time-series data," in *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, ser. IJCAI '13. AAAI Press, 2013, pp. 1444–1450.

[12] A. Arnold, Y. Liu, and N. Abe, "Temporal causal modeling with graphical granger methods," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '07. New York, NY, USA: ACM, 2007, pp. 66–75.

[13] G. Das, K. ip Lin, H. Mannila, G. Renganathan, and P. Smyth, "Rule discovery from time series." AAAI Press, 1998, pp. 16–22.

[14] B. Chiu, E. Keogh, and S. Lonardi, "Probabilistic discovery of time series motifs," in *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '03. New York, NY, USA: ACM, 2003, pp. 493–498.

[15] A. Y. Ng and S. Russell, "Algorithms for inverse reinforcement learning," in *in Proc. 17th International Conf. on Machine Learning*. Morgan Kaufmann, 2000, pp. 663–670.

[16] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, ser. A Bradford book. MIT Press, 1998.

[17] G. Taylor and R. Parr, "Kernelized value function approximation for reinforcement learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: ACM, 2009, pp. 1017–1024.