A Neural Monte Carlo Based Approach for Online Bayesian Change Point Detection

Xiaochuan Ma, Lifeng Lai, and Shuguang Cui

***Abstract*—Online quickest change-point detection (QCD) is widely used in many applications such as network monitoring, power outage detection, etc. The optimal solutions to the QCD problem under different settings have been extensively studied. Most of these solutions require a priori information about the QCD model and i.i.d. data samples. However, in many real-world applications, these requirements may not be satisfied. In these situations, the optimal QCD rules are not available. This paper proposes a Neural Monte Carlo (NMC) based data-driven change-point detection rule for the online Bayesian QCD problem. The only required assumption for the NMC-based method is that the random change point follows a Geometric distribution. In this method, the posterior probability is treated as a value function and approximated by a randomized neural network. Only the linear output layer of the neural network is trainable. Therefore, trained with the Gradient Monte Carlo algorithm, this neural network is guaranteed to converge. More importantly, this method works not only for i.i.d. data samples but also for non-i.i.d. data. Numerical results illustrate that the NMC-based method can detect the change point accurately and timely.**

***Index Terms*—Quickest change-point detection, data-driven, random neural network, Gradient Monte Carlo algorithm, non-i.i.d.**

I. Introduction

Quickest change point detection (QCD) is an important task in many applications, including intrusion detection in computer networks [1], outage detection in power systems [2], [3], health care [4], financial market surveillance [5], etc. Timely and accurate detection of the change is the goal we want to achieve in the QCD problem [6]–[11].

The QCD problem was extensively studied and many powerful methods have been proposed for different problem settings. For the Bayesian QCD problem when the pre-change and post-change distributions, and the prior probability of the change point are known, [12] gives the optimal solution. For the non-Bayesian QCD problem where the change point is assumed to be unknown but fixed, the solution Cumulative Sum (CUSUM) is given in [13]. As proved in [14], the CUSUM method is optimal for the non-Bayesian QCD problem under Lorden's criteria. In addition, QCD problems under different other assumptions of the latent stochastic QCD process are studied in [15]–[17]. However, two limitations make these methods hard to be used in many real-world applications. First, full knowledge of the sample distributions and their latent statistical structure is required to run these methods. Although there are some works that assume the distributions are unknown, they still assume the statistical structure of the observations is known. For example, in [15], the distributions of the data samples are unknown but they are assumed to belong to the multivariate exponential family. However, in many real-world applications, this knowledge about the latent stochastic QCD process is also unknown. A common situation is that the only given information is the historical ground truth data. Secondly, many of the existing methods assume the observed data samples are independent and identically distributed (i.i.d.), which is not always true in many real-world applications. As a promising approach to address these issues, the data-driven QCD method becomes more and more of interest in recent years. At the same time, machine learning provides many efficient algorithms to solve data-driven problems. Therefore, machine learning algorithms have already been applied to QCD problems. For example, [18]–[22] propose different data-driven algorithms for non-Bayesian QCD problems. On the other hand, as the Bayesian QCD process can be viewed as a partially observable Markov decision process (POMDP), reinforcement learning based methods can also be applied to solve the data-driven Bayesian QCD problems. In [23], the tabular Q-learning is applied to solve the change-point detection in the power system. A deep Q-network (DQN) based QCD rule is proposed in [24]. These two papers regard the Bayesian cost as a negative reward of a POMDP and use reinforcement learning methods to maximize the total reward. However, [23] still requires the knowledge of the pre-change distribution, and the performance of the tabular Q-learning method will be degraded when the state space is large. As for the DQN-based method in [24], although the instability in the training of DQN can be partially addressed by techniques such as experience replay and target Q-network, the convergence of the algorithm is still not guaranteed. The instability in the training process can affect the performance of the DQN-based QCD rule.

In this paper, we propose a Neural Monte Carlo (NMC) based method to solve the data-driven Bayesian QCD problem. In the optimal solution of the Bayesian QCD problem for i.i.d. data samples, an alarm will be raised once the posterior false alarm probability is lower than a threshold. On the other hand, the posterior false alarm probability can be regarded

X. Ma, L. Lai are with the Department of Electrical and Computer Engineering, University of California, Davis, CA, 95616. Email:{xcma, lflai}@ucdavis.edu. S. Cui is currently with the School of Science and Engineering (SSE) and Future Network of Intelligence Institute (FNii), the Chinese University of Hong Kong, and Shenzhen Research Institute of Big Data, Shenzhen, China, 518172 (e-mail: shuguangcui@cuhk.edu.cn). This work of X. Ma and L. Lai was supported in part by the National Science Foundation under Grants CNS-1824553 and ECCS-2000415. The work of S.Cui was supported in part by Shenzhen Outstanding Talents Training Fund 202002, and by Guangdong Research Projects No. 2017ZT07X152 and No. 2019CX01X104.

as a value function and learned from the historical data set. Concretely, at any time t we get a reward if we raise an alarm. If the change happens after t, the reward is 1; otherwise, the reward is 0. In this case, the false alarm probability at time t is equivalent to the value function of raising an alarm at time t. Inspired by these facts, we propose a reinforcement learning based method to solve the data-driven Bayesian QCD problem. First, we apply a randomized neural network to approximate the posterior false alarm probability. This neural network takes the historical data observations as input and outputs the approximation of the posterior false alarm probability. Since the posterior false alarm probability is mainly determined by the recently collected data samples rather than earlier data samples, the input of the neural network is the data samples within the most recent sliding window. With this neural network, the posterior false alarm probability can be monitored as new data samples come up. In particular, all the weights in this neural network except the linear output layer are untrainable. Therefore, training with the Gradient Monte Carlo algorithm [25], the neural network is guaranteed to converge. Afterward, following the idea of the optimal solution of the Bayesian QCD problem for i.i.d. data samples, the proposed NMC-based QCD rule raises an alarm once the approximation of the posterior false alarm probability meets a given threshold. The optimal threshold is chosen based on the performance on the validation data set. Besides, as a solution to the data-driven QCD problem, this method does not require prior knowledge about pre-change and post-change distributions. The only assumption is that the change point is a geometric random variable, which is satisfied in many real-world phenomena, such as failure times. More importantly, the proposed NMC-base QCD rule also works for non-i.i.d. data samples. The observation model of the different non-i.i.d. QCD problems could be different. Data sequences generated by a hidden Markov model (HMM) is a special case of non-i.i.d. data. In this paper, we take the HMM QCD problem as an example of non-i.i.d. QCD problems to explain how to apply the NMC-base QCD rule to non-i.i.d. observation data. Finally, numerical experiments are carried out and the results show that this NMC-based QCD rule has good performance in different Bayesian QCD problem settings.

The remainder of the paper is organized as follows. In Section II, we introduce the online Bayesian QCD problems for the i.i.d. case and HMM case. Then we introduce the optimal solution for QCD problems for i.i.d. case and HMM case in Section III. Afterwards, we propose the NMC-based QCD rule for i.i.d. case, HMM case and other non-i.i.d. cases in Section IV and V, respectively. Simulation results are provided in Section VI. Finally, we conclude this paper in Section VII.

II. Problem formulation

In this section, we introduce the formulation of the QCD problem with two different observation models, i.e., the i.i.d. case and the Hidden Markov Model (HMM) case.

A. i.i.d observation model

Consider a probability space $(\Omega, \mathcal{F}, \mathbb{P})$ that hosts a stochastic process $\{X_t\}_{1\leq t}$. Let $\lambda : \Omega \mapsto \{0, 1, \ldots\}$ be the time when the distribution of X_t changes. In the i.i.d. observation case, the observations $\{X_i, i \geq \lambda\} \overset{\text{i.i.d}}{\sim} f_1(x)$ and $\{X_i, 0 \leq i < \lambda\} \overset{\text{i.i.d}}{\sim} f_0(x)$. Given λ, $\{X_t\}_{t\geq 0}$ are independent. In addition, $\mathbb{F} = (\mathcal{F}_t)_{t\geq 0}$ is the filtration generated by the stochastic process $\{X_t\}_{t\geq 0}$; namely, $\mathcal{F}_0 = \{\emptyset, \Omega\}$ and $\mathcal{F}_t = \sigma(X_1, X_2 \ldots X_t)$.

B. HMM observation model

Let $\{Y_t, t \geq 0\}$ be a time-homogeneous Markov chain on a probability space $(\Omega_Y, \mathcal{F}_Y, \mathbb{P}_Y)$ with finite state space $\mathcal{Y} = \{1, 2, ..., I\}$, and transition matrix P in which $P(k, i) := \mathbb{P}(i|k)$ for $i, k \in \mathcal{Y}$. Suppose $\mathcal{Y}_1 = \{I_0 + 1, I_0 + 2, ..., I\}$ is a closed subset of $\mathcal{Y}$, where $I_0 < I$. The collection of remaining states $\mathcal{Y}_0 = \mathcal{Y} \backslash \mathcal{Y}_1 = \{1, 2, ..., I_0\}$ does not have any closed sets. In other words, in the transition matrix P, $P(y_t, y_{t+1}) = 0$ if $y_t \in \mathcal{Y}_1$ and $y_{t+1} \in \mathcal{Y}_0$. The change time $\lambda : \Omega \mapsto \{0, 1, \ldots\}$ is the first time the state $Y_t \in \mathcal{Y}_1$, i.e., λ is the time when the hidden states change from $\mathcal{Y}_0$ to $\mathcal{Y}_1$. In addition, the initial probability $P(Y_0 = i) = \eta_i$ where $\sum_{i\in\mathcal{Y}} \eta_i = 1$. Let $\overrightarrow{\eta} = (\eta_1, \eta_2, ..., \eta_I)$.

However, the sequence $\{Y_t, t \geq 0\}$ can not be directly observed. $\{X_t\}_{1\leq t}$ is the directly observable process hosted by a probability space $(\Omega, \mathcal{F}, \mathbb{P})$ and the distribution of X_t depends on the hidden state Y_t. Let $f_y(\mathcal{X}), y \in \mathcal{Y}$ be the probability measures on a measurable space $(\mathcal{X}, \mathfrak{X})$, then

$$\begin{aligned}&\mathbb{P}\left(Y_0 = y_0, Y_1 = y_1, \ldots, Y_t = y_t, X_1 = x_1, \cdots, X_t = x_t\right)\\ &= \overrightarrow{\eta}(y_0) \textstyle\prod_{n=1}^{t} P\left(y_{n-1}, y_n\right) f_{y_n}(x_n)\end{aligned}$$

for $t \geq 1$, $y_0, y_1, ..., y_t \in \mathcal{Y}$.

C. Bayesian quickest change detection problem

In Bayesian QCD problem, the change point, λ, follows a prior distribution P_λ. In this paper, we assume P_λ is

$$\mathbb{P}\{\lambda = t\} = \left\{ \begin{array}{ll} \rho, & \text{if} \quad \mathrm{t = 0} \\ (1-\rho)^t \rho, & \text{if} \quad \mathrm{t \neq 0} \end{array} \right. .$$

For the HMM case, the transition matrix satisfies $\sum_{k\in\mathcal{Y}_1} P(i, k) = \rho$ for every $i \in \mathcal{Y}_0$ and the initial probabilities satisfy $\sum_{k\in\mathcal{Y}_1} \eta_k = \rho$.

Our goal is to detect the change point λ quickly and accurately, based on the observation sequence $\{X_t, t > 0\}$. Let τ be the time we raise an alarm. Then the false alarm happens if $\tau < \lambda$ and the delay is $(\lambda - \tau)_+$. Hence we define the expected cost of the change point detection problem as

$$C(\tau) = \mathbb{E}[\mathbf{1}_{\{\tau<\lambda\}} + c(\lambda - \tau)_+] \tag{1}$$

where $\mathbf{1}_{\{\cdot\}}$ the indicator function and c is the unit cost of detection delay. Therefore, the best expected cost for the change point detection problem is

$$V_0 = \underset{\tau\in\mathcal{T}}{inf}\, \mathbb{E}[\mathbf{1}_{\{\tau<\lambda\}} + c(\lambda - \tau)_+] \tag{2}$$

where $\mathcal{T}$ is the space of stopping time $\tau : \Omega \rightarrow \{1, 2, \cdots, T\}$.

As will be discussed in Section III, the optimal solution for online Bayesian QCD problem can be found when ρ, f_0 and f_1 or $\overrightarrow{\eta}, P$ and $\mathbb{P}(x|y)$ are known. However, the knowledge is not always available in real-world problems. When the true underlying model is different from the model used to derive the optimal solution, the performance could be seriously affected. In practice, a common situation is that the only information we have is the historical data about the QCD process. In this paper, we want to solve the online Bayesian QCD problem under the data-driven problem setting. Concretely, based on the historical dataset, our goal is to find a data-driven stopping rule which can achieve or get close to V_0.

III. The Optimal solution with Prior Knowledge of the QCD process

Before discussing the proposed data-driven solution, we introduce the optimal solution of the online Bayesian QCD problem. The optimal solution only works when prior knowledge of the QCD process is known. However, the structure of the optimal solution is important for the understanding of the proposed NMC-based QCD rule, which will be introduced in Sections IV and V for the i.i.d. and HMM observation models respectively. Therefore, in this section, we provide a brief introduction of the optimal solution for Bayesian QCD problems for the i.i.d. case and the HMM case. For detailed proof of the optimal QCD rules in these two cases, please refer to [10], [17].

A. The i.i.d. case

For $t \geq 0$, let $\Pi_t = (\Pi_t^{(0)}, \Pi_t^{(1)}) \in \mathcal{Z}$ be the posterior probability process defined as $\Pi_t^{(1)} := \mathbb{P}\{\lambda \leq t|\mathcal{F}_t\}$ and $\Pi_t^{(0)} := \mathbb{P}\{\lambda > t|\mathcal{F}_t\}$ where $\mathcal{Z} \triangleq \{\Pi \in [0,1]^2 | \Pi^{(1)} + \Pi^{(0)} = 1\}$.

Following the derivation in [10], the expected cost in (1) can be expressed as

$$C(\tau) = \mathbb{E}\left[\sum\nolimits_{n=0}^{\tau-1} c\Pi_n^{(1)} + \Pi_\tau^{(0)}\right]. \tag{3}$$

Then we can define the cost-to-go function as a function of the posterior probability,

$$W(\Pi_t) = \min\left(\Pi_t^{(0)}, c\Pi_t^{(1)} + \mathbb{E}\left[W(\Pi_{t+1})|\mathcal{F}_t\right]\right). \tag{4}$$

The first item inside the minimization is the expected cost of raising an alarm immediately and the second item is the expected cost of observing more data samples. $W(\Pi_t)$ is the minimal expectation of the cost we still need to pay in the future based on the current state Π_t.

For the i.i.d case, when the pre-change distribution f_0, post-change distribution f_1, and the distribution of change ρ are known, we are able to update the posterior probability recursively following:

$$\Pi_t^{(0)} = \frac{(1-\rho)\Pi_{t-1}^{(0)} f_0(x_t)}{(1-\rho)\Pi_{t-1}^{(0)} f_0(x_t) + \left(\Pi_{t-1}^{(1)} + \Pi_{t-1}^{(0)}\rho\right) f_1(x_t)} \tag{5}$$

and $\Pi_t^{(1)} = 1 - \Pi_t^{(0)}$. The initial state, $\Pi_0 = (1-\rho, \rho)$. Based on this recursive updating rule, $\mathbb{E}\left[W(\Pi_{t+1})|\mathcal{F}_t\right]$ can be calculated as

$$\mathbb{E}\left[W(\Pi_{t+1})|\mathcal{F}_t\right] = \int W(\Pi_{t+1}(\Pi_t, x)) \sum_{i\in\{0,1\}} f_i(x)\Pi_t^{(i)} dx.$$

Then, we can use dynamic programming (DP) to solve the Bellman equation (4) and obtain the cost-to-go $W(\Pi)$ for all $\Pi \in \mathcal{Z}$.

After solving (4) using dynamic programming, we have the cost-to-go function $W_t(\Pi_t)$ for $0 \leq t \leq T$. The optimal stopping rule is $\tau^* = \inf\left\{t | W_t(\Pi_t) = \Pi_t^{(0)}\right\}$. As discussed in [10], when P_λ is a Geometric distribution, then the optimal solution can be further simplified as

$$\tau_{opt} = \inf\{t \geq 0 | \Pi_t^{(0)} \leq \pi^*\} \tag{6}$$

where $\pi^* = \sup\{\pi \in [0,1] | \pi = W((\pi, 1-\pi))\}$. This rule indicates that we should raise an alarm once the expected cost of false alarm is smaller than the expected cost of observing more data samples.

B. The HMM case

For the HMM case, we can apply a similar solution as in the i.i.d. case. However, the posterior probability we used in the i.i.d. case, Π_t, can not be recursively updated. For this reason, we define the posterior probabilities $\tilde{\Pi}_t = (\tilde{\Pi}_t^{(1)}, \tilde{\Pi}_t^{(2)}, ..., \tilde{\Pi}_t^{(I)})_{t\geq 0} \in \tilde{\mathcal{Z}}$, where $\tilde{\Pi}_t^{(i)} := \mathbb{P}\{y_t = i|\mathcal{F}_t\}$ for all $i \in \mathcal{Y}$ and $\tilde{\mathcal{Z}} = \{\tilde{\Pi} \in [0,1]^I | \sum_{i\in\mathcal{Y}} \tilde{\Pi}^{(i)} = 1\}$. With this definition, the posterior false alarm probability can be expressed as $\sum_{i\in\mathcal{Y}_0} \tilde{\Pi}^{(i)}$. Therefore, the expected cost in (3) can be expressed as

$$C(\tau) = E\left[c\sum\nolimits_{n=0}^{\tau-1}\sum\nolimits_{i\in\mathcal{Y}_1}\tilde{\Pi}_n^{(i)} + \sum\nolimits_{i\in\mathcal{Y}_0}\tilde{\Pi}_\tau^{(i)}\right]. \tag{7}$$

Then we can define the cost-to-go function for this DP problem as a function of the posterior probability,

$$\begin{aligned} W(\tilde{\Pi}_t) = \min\Bigg(&\sum\nolimits_{i\in\mathcal{Y}_0}\tilde{\Pi}_t^{(i)}, \\ &c\sum\nolimits_{i\in\mathcal{Y}_1}\tilde{\Pi}_t^{(i)} + \mathbb{E}\left[W(\tilde{\Pi}_{t+1})|\mathcal{F}_t\right]\Bigg). \end{aligned} \tag{8}$$

The first item inside the minimization is the expected cost of raising an alarm immediately and the second item is the expected cost of observing one more data sample. $W(\tilde{\Pi}_t)$ is the minimal expectation of the cost we still need to take in the future based on the current state Π_t.

For the HMM case, when the sample distributions $\{f_y\}_{y\in\mathcal{Y}}$, transition matrix P, and the distribution of change ρ are known, we are able to update the posterior probability $\tilde{\Pi}_t$ recursively. Concretely, $\tilde{\Pi}_t$ can be updated recursively as:

$$\tilde{\Pi}_t^{(i)} = \frac{\sum_{k\in\mathcal{Y}}\tilde{\Pi}_{t-1}^{(k)}P(k,i)f_i(x_t)}{\sum_{j\in\mathcal{Y}}\sum_{k\in\mathcal{Y}}\tilde{\Pi}_{t-1}^{(k)}P(k,j)f_j(x_t)}, \text{for } i \in \mathcal{Y}, \tag{9}$$

and $\tilde{\Pi}_0 = (\eta_1, \eta_2, ..., \eta_I)$. Based on this recursive updating rule, $\mathbb{E}\left[W(\tilde{\Pi}_{t+1})|\mathcal{F}_t\right]$ can be calculated as

$$\mathbb{E}\left[W(\tilde{\Pi}_{t+1})|\mathcal{F}_t\right] = \int W(\tilde{\Pi}_{t+1}(\tilde{\Pi}_t, x))\sum\nolimits_{i\in\mathcal{Y}} f_i(x)\tilde{\Pi}_t^{(i)} dx.$$

Then, we can use DP method to solve the Bellman equation (8) and calculate the cost to go $W(\tilde{\Pi})$ for all $\tilde{\Pi} \in \tilde{\mathcal{Z}}$. The optimal stopping rule is

$$\tau^* = \inf\left\{t | W_t(\tilde{\Pi}_t) = \sum\nolimits_{i\in\mathcal{Y}_0} \tilde{\Pi}_t^{(i)}\right\}. \tag{10}$$

This rule indicates that we should raise an alarm once the expected cost of false alarm is smaller than the expected cost of observing more data samples.

IV. A Neural Monte Carlo based QCD rule for the i.i.d case

As described in Section III, a key step in the optimal QCD rules is to update posterior probabilities (Π_t or $\tilde{\Pi}_t$) recursively. This updating step can only be implemented when the a-priori information such as P, $\overrightarrow{\eta}$, ρ, f_0, f_1 and f_y are known. However, in many applications, it is common that a historical dataset with finite data samples is the only given resource. Concretely, only the observation sequences and the true change points in the dataset are known. We even do not know if the data samples are i.i.d., following an HMM, or some other non-i.i.d. process. In this case, it is hard to accurately extract these a-priori information from the data set. Therefore, we need a data-driven method that can help us estimate the posterior probabilities from the data. In this section, we will propose the Neural Monte Carlo (NMC) based solution for the i.i.d. case. In the next section, we will explain why this method also works for the HMM case and other non-i.i.d. cases.

From (6), we know that the posterior probability $\Pi_t^{(0)}$ and the threshold π^* are key parts of the optimal QCD rule for the i.i.d. case. In the data-driven setting, if we can approximate the posterior probability with $\hat{\Pi}_t^{(0)}$ for any time t, then we can select the optimal threshold $\hat{\pi}^*$ using line search and finally have a data-driven QCD rule similar to (6). To this end, we propose a Neural Monte Carlo (NMC) based solution for the data-driven QCD problem. The steps of the NMC-based QCD rule is given in Algorithm 1. Next, we will introduce these steps of this NMC-based QCD method.

A. A Neural Monte Carlo approximation model

If the cost of false alarm $c_F = 1$, $\Pi_t^{(0)}$ can be seen as the expected cost of raising an alarm given all data samples collected by t. In other words, the value function corresponding to the observations $\{x_1, x_2, ...x_t\}$ is $\Pi_t^{(0)}$. Therefore, the problem of estimating $\Pi_t^{(0)}$ can be seen as a value function approximation problem.

In the data set, we have the data sequences and the true change points of these sequences. With this data set, we can use the Monte Carlo method to approximate $\Pi_t^{(0)}$. Because continuous observation data samples are common in the QCD problem, we have a continuous input space. Therefore, we approximate $\Pi_t^{(0)}$ using a randomized neural network in this paper. In a randomized neural network, only the weights between the hidden layer and the output layer are trained while all other weights are frozen since initialized. In particular, the last layer of the randomized neural network is a linear layer. Therefore, training the randomized neural network becomes a convex problem and has a convergence guarantee. More importantly, as proved in [26], [27], a randomized neural network can accurately approximate any continuous functions with a sufficiently wide hidden layer. Therefore, in this paper, we apply a simple shallow neural network with one hidden layer. If we need a more powerful model for specific applications, a deep extension to this neural network is immediate.

B. Data Preprocessing

Since the dimension of the input layer of the randomized neural network is fixed and the size of all observed samples $(X_1, \cdots, X_t)$ changes with different time t, we can not input $(X_1, \cdots, X_t)$ to the neural network. Therefore, we set the input of the neural network as the observations in a sliding window with width w, i.e., $\mathbf{X}_t = (X_{t-w+1}, \cdots, X_t)$. From equation (5), we can see that recently collected data samples are usually more important than earlier data samples in the calculation of $\Pi_t^{(0)}$ for the i.i.d case. In other words, with an appropriate value of w, the data samples in the sliding window $\mathbf{X}_t$ are sufficient to make a good estimation of $\Pi_t^{(0)}$. Typically, we select a large w if earlier data samples are important in the calculation of posterior probability. Otherwise, we can use a relatively narrow sliding window. For every time t and the corresponding input $\mathbf{X}_t$, the reward of raising alarm at t, R_t, is 1 if $t < \lambda$. On the other hand, if $t \geq \lambda$, $R_t = 0$.

The dataset includes N episodes of the change process. For the ith episode, the data includes a sequence of T samples, $\mathbf{S}_i = (X_{i,1}, \cdots, X_{i,T})$ and the true change point λ_i. To make the sequential data fit the sliding window, we need to preprocess the sequences in the dataset. For the first $w-1$ samples in each sequence, they do not have enough previous samples to make a w long input sequence. To handle this issue, we add a $w-1$ long prefix to every sequence in the dataset. Firstly, we collect all pre-change data samples in the training set and obtain a pool of pre-change samples. Then for each episode, $w-1$ samples are randomly picked from the pool of pre-change samples and added in front of the data sequence. As a result, for the ith episode, the data sequence becomes $\tilde{\mathbf{S}}_i = (X_{i,2-w}, \cdots, X_{i,0}, X_{i,1}, \cdots, X_{i,T})$. For each data sequence $\tilde{\mathbf{S}}_i$, we can generate T input data samples $\{\mathbf{X}_{i,1}, \mathbf{X}_{i,2}..., \mathbf{X}_{i,T}\}$ where $\mathbf{X}_{i,t} = (X_{i,t-w+1}, \cdots, X_{i,t})$. The corresponding reward samples are $\{R_{i,1}, R_{i,2}..., R_{i,T}\}$ where $R_{i,t} = 0$ if $t > \lambda_i$ and $R_{i,t} = 1$ if $t \leq \lambda_i$. By combining M episodes of data samples, $\{\tilde{\mathbf{S}}_i\}_{1\leq i\leq M}$, we have the training data set $\{\mathbf{X}_{NMC}, \overrightarrow{R}_{NMC}\}$. The rest $N-M$ episodes, $\{\tilde{\mathbf{S}}_i\}_{M+1\leq i\leq N}$, are used for building the validation set and test set. Typically, we set M as 70% of N.

Following the steps stated above, we have built a training dataset. However, in some situations, the training set could be imbalanced. A common imbalanced situation is that, when the length of the sequence T is very large, then there will be many more post-change data samples than pre-change data samples in the training set. With such an unbalanced training set, the accuracy of the posterior probability approximation model and the performance of the data-driven QCD rule can be compromised. In this case, we can use rebalance techniques, such as data selection or re-sampling, to process the training set $\{\mathbf{X}_{NMC}, \overrightarrow{R}_{NMC}\}$ if it is unbalanced. Concretely, we can rebalance the training data set by discarding the data samples after a threshold time $\tilde{T} < T$. Since most of the post-change data samples are at the later part of the sequence, dropping the later data samples can reduce the fraction of post-change data and make the training set balanced. In this paper, the threshold time $\tilde{T}$ is treated as a hyper-parameter that can be tuned using the validation data set.

C. Training Process of the Neural Monte Carlo model

The randomized neural network is trained using Monte Carlo methods, as shown in Algorithm 1. Let $\boldsymbol{\theta}_0 \in \mathbb{R}^{(w+1)\times d}$ be the weights of the hidden layer, where d is the number of nodes in the hidden layer. Then the output of the hidden layer is $\mathbf{O}_{0,k} = \sigma(\boldsymbol{\theta}_0^T \mathbf{I}_{0,k})$, where $\mathbf{I}_{0,k} = [\mathbf{X}_k, 1]$ and $\mathbf{X}_k$ is an input data sample in the training set. Here σ is the activation function of the hidden layer. Elements in $\boldsymbol{\theta}_0$ are typically initialized by the standard normal distribution and will not be changed in the training process. Let $\boldsymbol{\theta}_1 \in \mathbb{R}^{(d+1)}$ be the weights of the output layer. Then the output of the neural network is $\boldsymbol{\theta}_1^T \mathbf{I}_{1,k}$ where $\mathbf{I}_{1,k} = [\mathbf{O}_{0,k}, 1]$. $\boldsymbol{\theta}_1$ is the weights we want to optimize in the training process. Since $\boldsymbol{\theta}_0$ is fixed, we can get a hidden output data set $\mathbf{I}_{1,NMC}$ from $\mathbf{X}_{NMC}$. In this case, this training problem becomes a linear value-function approximation problem. In [25], the gradient Monte Carlo algorithm is introduced to solve this linear value-function approximation problem and the updating rule of $\boldsymbol{\theta}_{1,t}$ is

$$\boldsymbol{\theta}_{1,t+1} = \boldsymbol{\theta}_{1,t} + \alpha(R_t - \boldsymbol{\theta}_{1,t}^T \mathbf{I}_{1,t})\mathbf{I}_{1,t} \tag{11}$$

for every step t, where α is the step size. Since the approximation model we apply is linear, the convergence of this training process is guaranteed.

From the updating rule (11), we can see that

$$\mathbb{E}[\boldsymbol{\theta}_{1,t+1}|\boldsymbol{\theta}_{1,t}] = \boldsymbol{\theta}_{1,t} + \alpha(\mathbb{E}[R_t \mathbf{I}_{1,t}] - \mathbb{E}[\mathbf{I}_{1,t}\mathbf{I}_{1,t}^T]\boldsymbol{\theta}_{1,t}).$$

Therefore, this algorithm will converge to $\boldsymbol{\theta}_{1,GMC}$ at which

$$\mathbb{E}[R_t \mathbf{I}_{1,t}] - \mathbb{E}[\mathbf{I}_{1,t}\mathbf{I}_{1,t}^T]\boldsymbol{\theta}_{1,GMC} = \mathbf{0}.$$

Since the data set $\{\mathbf{I}_{1,NMC}, \overrightarrow{R}_{NMC}\}$ is given in the data-driven QCD problem, we can estimate $\mathbb{E}[R_k \mathbf{I}_{1,k}]$ and $\mathbb{E}[\mathbf{I}_{1,k}\mathbf{I}_{1,k}^T]$ with the sample mean. Here $\mathbf{I}_{1,NMC} \in \mathbb{R}^{(d+1)\times MT}$, $\overrightarrow{R}_{NMC} \in \mathbb{R}^{MT}$. If there is enough data such that $\mathbb{E}[R_k \mathbf{I}_{1,k}]$ and $\mathbb{E}[\mathbf{I}_{1,k}\mathbf{I}_{1,k}^T]$ can be well estimated by their sample mean of the data set, the weights can be directly calculated as

$$\boldsymbol{\theta}_1 = (\mathbf{I}_{1,NMC}\mathbf{I}_{1,NMC}^T)^{-1}(\mathbf{I}_{1,NMC}\overrightarrow{R}_{NMC}). \tag{12}$$

From the perspective of computational complexity, the direct calculation (12) can only be used if the number of hidden nodes is relatively small such that a $(d+1)\times(d+1)$ matrix can be inverted with a reasonable running time and memory. Therefore, as shown in Algorithm 1, if we have enough computational resource to invert a $(d+1)\times(d+1)$ matrix, we can directly calculate $\boldsymbol{\theta}_1$. Otherwise, we can update $\boldsymbol{\theta}_1$ following (11) iteratively.

Algorithm 1: NMC-based SCD rule

```
Data preprocessing following Section IV-B. Get data
  set $\{\mathbf{X}_{NMC}, \overrightarrow{R}_{NMC}\}$;
Initialize a random neural network with hyper
  parameter $d$, $w$, $\tilde{T}$ and weights following standard
  gaussian distribution;
Get data set $\mathbf{I}_{1,NMC}$;
if a $(d+1)\times(d+1)$ matrix can be inverted with
  reasonable computational resource then
    $\boldsymbol{\theta}_1 = (\mathbf{I}_{1,NMC}\mathbf{I}_{1,NMC}^T)^{-1}(\mathbf{I}_{1,NMC}\overrightarrow{R}_{NMC})$;
else
    for $\mathbf{I}_k \in \mathbf{I}_{1,NMC}$ do
        $\boldsymbol{\theta}_{1,t+1} = \boldsymbol{\theta}_{1,t} + \alpha(R_t - \boldsymbol{\theta}_{1,t}^T \mathbf{I}_{1,t})\mathbf{I}_{1,t}$
    end
end
Select the threshold $\hat{\pi}^*$ by line search method.
Fine-tune the hyper-parameters $w$, $d$ and $\tilde{T}$.
Finally, the NMC-based QCD rule is
  $\tau = \inf\{t \geq 0|\hat{\Pi}_t^{(0)} \leq \hat{\pi}^*\}$.
```

Although the neural network is designed to approximate the posterior probability, the output may not necessarily fall in $[0,1]$ since the output layer is a linear layer. Therefore, after getting the output of the neural network, we clip the output to make sure it is a value in $[0,1]$ as

$$\hat{\Pi}^{(0)}(\boldsymbol{\theta}_1, \mathbf{X}_k) = \begin{cases} 1, & \text{if} \quad \boldsymbol{\theta}_1^T \mathbf{I}_{1,k} \geq 1 \\ 0, & \text{if} \quad \boldsymbol{\theta}_1^T \mathbf{I}_{1,k} \leq 0 \\ \boldsymbol{\theta}_1^T \mathbf{I}_{1,k}, & \quad \text{Otherwise} \end{cases}.$$

D. Threshold Selection

After training the neural network, we can estimate posterior probability $\hat{\Pi}_t^{(0)}$ for any time t. Next, we need to determine the threshold $\hat{\pi}_*$ for our NMC-based QCD rule. First, we apply the well-trained neural network to approximate the posterior probability $\hat{\Pi}_t^{(0)}$ for every sequence in the validation set. Second, we run Monte Carlo experiments on sequences of in the validation set and record the Bayesian costs for all candidate threshold $\pi \in \{1/K, 2/K, ..., (K-1)/K, 1\}$, using the threshold rule $\tau = \inf\{t \geq 0|\hat{\Pi}_t^{(0)} \leq \pi\}$. Here, K is the number of candidates in the line search method. Finally, based on the Bayesian cost records, we set $\hat{\pi}^*$ as the candidate π which is corresponding to the lowest Bayesian cost. Finally, we have the NMC-based QCD rule as

$$\tau = \inf\{t \geq 0|\hat{\Pi}_t^{(0)} \leq \hat{\pi}^*\}. \tag{13}$$

V. A Neural Monte Carlo based QCD rule for the HMM case

In Section IV, by replacing the two key parts in the optimal QCD rule, $\{\Pi_t^{(0)}, \pi^*\}$, with corresponding approximations $\{\hat{\Pi}_t^{(0)}, \hat{\pi}^*\}$, we proposed a NMC-based rule (13) for the data-driven Bayesian QCD problem in the i.i.d. case. It's natural to consider if we can do the same thing to the optimal rule

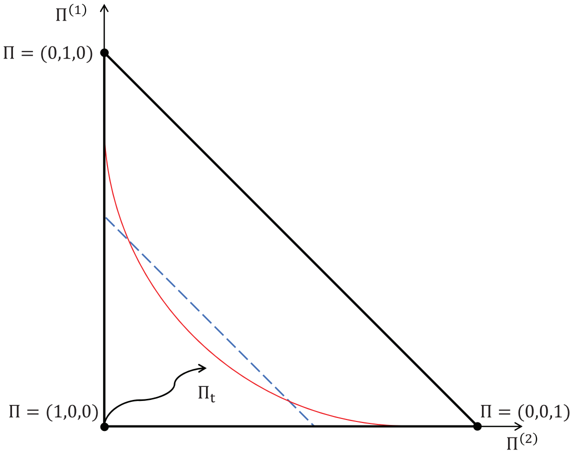


Figure 1: QCD boundaries of a simple QCD example in HMM case

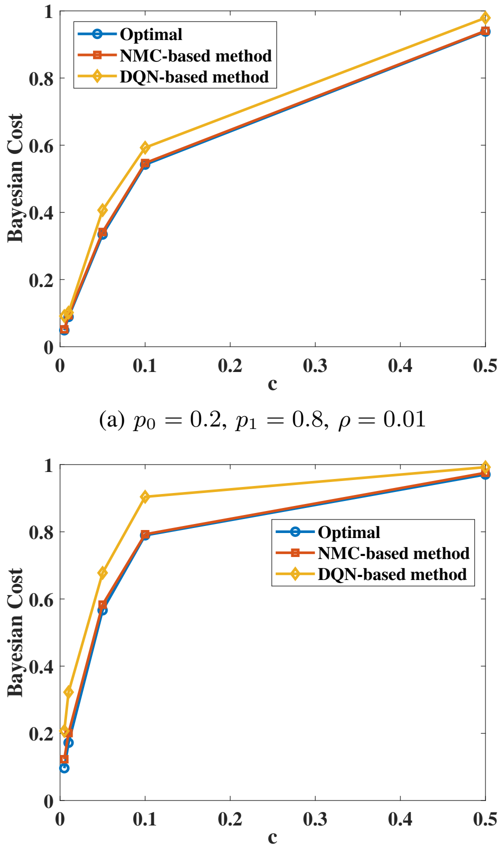
in the HMM case. Unfortunately, it is challenging to extract sufficient information for the optimal HMM QCD rule from the observations. Concretely, it is hard to estimate the hidden process from observations, e.g., the number of hidden states, which state the post-change state is and which state the pre-change state is, etc.. Without these information, even the number of elements $\tilde{\Pi}_t$ should include is unknown. Hence it is challenging to directly extend the optimal SCD rule for the HMM to a data-driven version.

However, although we cannot estimate $\tilde{\Pi}_t$, we can still estimate the posterior probability $\Pi_t^{(0)} = \sum_{i \in \mathcal{Y}_0} \tilde{\Pi}_t^{(i)}$, similar to in the i.i.d. case. As the sufficient statistics $\tilde{\Pi}_t$ is unavailable, the posterior probability $\Pi_t^{(0)}$ becomes a reasonable alternative indicator that can help to detect the change. Concretely, we can apply the threshold rule as

$$\tau = \inf \left\{ t \geq 0 | \Pi_t^{(0)} = \sum\nolimits_{i \in \mathcal{Y}_0} \tilde{\Pi}_t^{(i)} \leq \pi \right\}. \tag{14}$$

To illustrate this threshold rule, a simple example is given in Fig. 1. Assume we have a HMM QCD problem in which $\mathcal{Y} = \{1, 2, 3\}$, $\mathcal{Y}_0 = \{1\}$ and $\mathcal{Y}_1 = \{2, 3\}$. In the posterior probability space $\tilde{\mathcal{Z}} = \{\tilde{\Pi} \in [0, 1]^3 | \sum_{i \in \mathcal{Y}} \tilde{\Pi}^{(i)} = 1\}$, the optimal decision boundary given by (10) is the red curve. The straight blue line represents the decision boundary of the QCD rule (14). In general HMM QCD problems, the decision boundary of the QCD rule (14) is a plane in the space $\tilde{\mathcal{Z}}$ while the decision boundary of the optimal rule (10) is a surface in $\tilde{\mathcal{Z}}$. Since the information about the HMM is incomplete, we use a plane as the alternative of the surface. In the data-driven QCD problem, we apply the NMC-based QCD rule introduced in Section IV to approximate the plane of the QCD rule (14). Finally, we obtain the same NMC-based QCD rule as (13).

Due to the difference between the HMM case and the i.i.d. case, we need to make some adjustments when we apply the NMC-based SCD rule to HMM QCD problems. In HMM case, the data samples in one sequence are not independent. Therefore, the elements in the prefix of the data sequence should not be independently selected as in the i.i.d. case. In order to apply the NMC-based QCD rule to the HMM case, one change is needed for the data preprocessing step. In HMM case, we collect all pre-change subsequences from the data sequences $\{\mathbf{S}_i\}_{1 \leq i \leq N}$ as the pool of pre-change subsequences. Note that, we only collect pre-change subsequence longer than $w - 1$ samples. After that, for every sequence $\{\mathbf{S}_i\}$, $w - 1$ continuous data samples are randomly selected from the pool of pre-change subsequences and added to $\mathbf{S}_i$ as the prefix. Afterwards, following the same steps as discussed in Section IV, we have the NMC-based QCD rule for the HMM case, which has the same expression as (13).



(a) $p_0 = 0.2$, $p_1 = 0.8$, $\rho = 0.01$

(b) $p_0 = 0.3$, $p_1 = 0.7$, $\rho = 0.01$

Figure 2: The Bayesian costs of the three QCD methods in the i.i.d. Bernoulli experiment

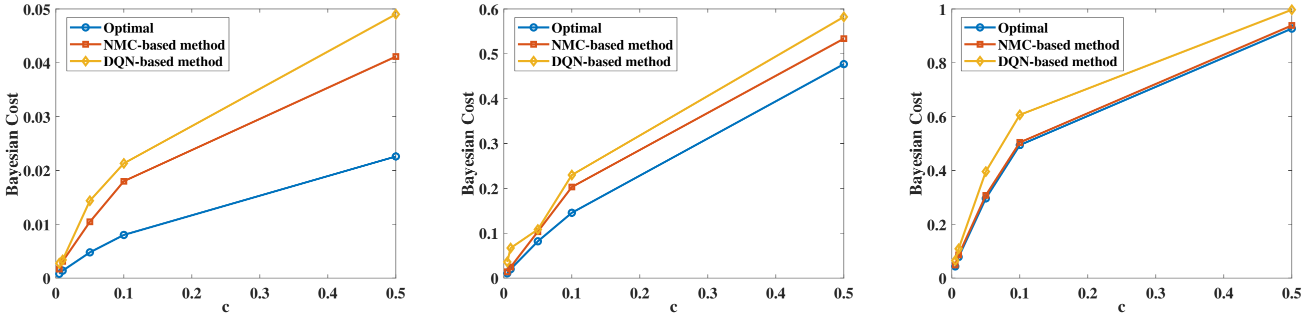
In the general non-i.i.d case, the posterior analysis and the detection boundary could be even more complicated than in the HMM case. That means getting the optimal solution for a QCD problem in general non-i.i.d. case becomes even harder. However, the posterior false alarm probability is still a reasonable indicator for the non-i.i.d. QCD problem and can be learned by following similar steps as those in the i.i.d. case and the HMM case. Therefore, the NMC-based rule could still be used for the QCD problem in different non-i.i.d. settings. This will be validated in the following section by simulation.

VI. Numerical results

To evaluate the performance of the proposed Neural Monte Carlo QCD method, different numerical examples are provided. In the following examples, we evaluate the performance of the NMC-based QCD method in i.i.d. and non-i.i.d. cases.

Table I: Performances of the three QCD rules in the i.i.d. Bernoulli experiment

		$p_0 = 0.2,\, p_1 = 0.8,\, \rho = 0.01$			$p_0 = 0.3,\, p_1 = 0.7,\, \rho = 0.01$		
		Delay	False alarm probability	Bayesian cost	Delay	False alarm probability	Bayesian cost
Optimal	c=0.5	0.1912	0.8427	0.9383	0.0401	0.9505	0.9706
	c=0.1	3.6891	0.1733	0.5422	2.7852	0.511	0.7895
	c=0.05	5.2904	0.0695	0.334	6.8788	0.2224	0.5663
	c=0.01	7.6478	0.0123	0.0888	14.1973	0.03	0.172
	c=0.005	8.9288	0.0044	0.049	16.1054	0.0157	0.0962
NMC-based method	c=0.5	0.2178	0.8316	0.9405	0.0137	0.9688	0.9757
	c=0.1	3.5632	0.191	0.5473	2.3779	0.5549	0.79269
	c=0.05	5.4253	0.0717	0.343	6.6513	0.2508	0.5832
	c=0.01	7.9296	0.0177	0.097	13.8156	0.0639	0.2021
	c=0.005	8.4346	0.0113	0.05347	17.3196	0.0382	0.1248
DQN-based method	c=0.5	0.017	0.9707	0.9792	0.0792	0.9528	0.9924
	c=0.1	3.8384	0.2089	0.5928	3.73	0.5373	0.9042
	c=0.05	3.2874	0.2423	0.4066	5.4489	0.4519	0.7243
	c=0.01	7.4965	0.0268	0.1017	13.0369	0.1922	0.3225
	c=0.005	7.4978	0.0539	0.0914	17.4648	0.1199	0.2072



(a) $\vec{\boldsymbol{\mu}}_0 = (0, 0)$, $\vec{\boldsymbol{\mu}}_1 = (4, 4)$ (b) $\vec{\boldsymbol{\mu}}_0 = (0, 0)$, $\vec{\boldsymbol{\mu}}_1 = (2, 2)$ (c) $\vec{\boldsymbol{\mu}}_0 = (0, 0)$, $\vec{\boldsymbol{\mu}}_1 = (1, 1)$

Figure 3: The Bayesian costs of the Gaussian QCD experiment with change in mean vector

We also test the robustness of the NMC-based QCD method using data generated by distributions that are different from the training data. Moreover, we compare the performances of the optimal solution, the DQN-based solution [24], and the NMC-based solution in these numerical examples.

In all the following experiments, we assume the pre-change and post-change distributions, and the prior distribution of the change point is known for the optimal QCD method. On the other hand, for training the NMC-based and DQN-based QCD rules, we only use a limited historical dataset, including data sequences and the corresponding change points. Concretely, we build the NMC-based and DQN-based QCD rules with data set including 20000 observation sequences and corresponding change points. Each sequence includes 600 observations. For the NMC-based method and DQN method, 14000 sequences are used to train the neural network, 3000 sequences are used for the validation set (tuning hyperparameters such as the size of the neural network, the width of the sliding window, the data rebalance parameter $\tilde{T}$, and the best threshold $\hat{\pi}^*$, etc.) and the rest 3000 sequences are used to test the performance of these methods. The hidden layer of the neural network in the NMC-based method has 1000 nodes. The DQN model includes two hidden layers with 200 and 100 nodes respectively. ReLU is used as the activation function of all the hidden layers in these two methods. For the training of the NMC-based method, the rebalance parameter is set as $\tilde{T} = 100$. In addition, the width of sliding windows for the NMC-based method and the DQN-based method are both 10. In most of the following experiments, the DQN-based and NMC-based models are implemented following these instructions. Further information will be provided if we need to make changes to these parameters in specific experiments.

A. *QCD experiments in i.i.d. case*

In the first example, we study the performance of the optimal QCD solution, the NMC-based solution, and the DQN-based QCD solution when the observations are i.i.d. discrete random variables. Concretely, $f_0 = Bern(p_0)$ and $f_1 = Bern(p_1)$, where p_0 and p_1 are the parameters of the pre-change and post-change Bernoulli distributions. The prior distribution of the change-points, P_λ, is a geometric distribution with parameter $\rho = 0.01$. Based on this information, we can calculate the posterior probability and further obtain the optimal solution by dynamic programming. Using the training data set, we obtain the DQN-based and NMC-based QCD rules. After that, we compare the Bayesian costs, $C(\tau)$, of

Table II: Performances of the three QCD rules in the i.i.d. 2D Gaussian experiment: change in mean vector

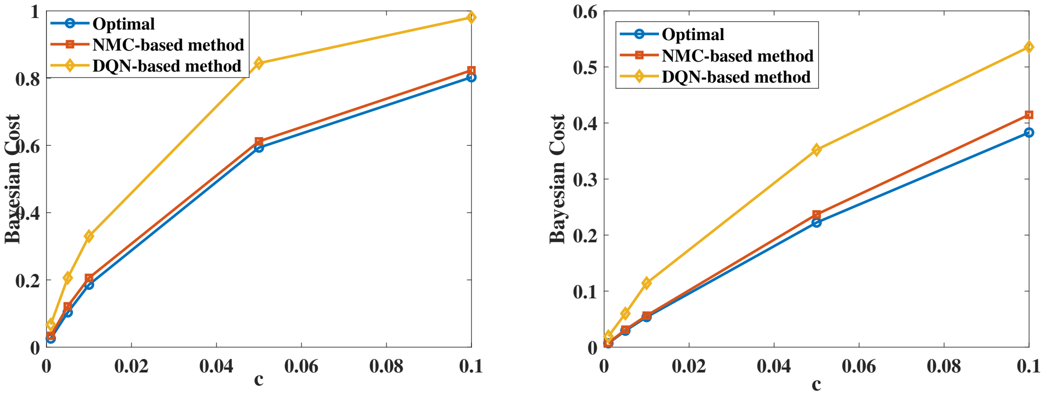
		$\vec{\boldsymbol{\mu}}_0 = (0,0)$, $\vec{\boldsymbol{\mu}}_1 = (1,1)$			$\vec{\boldsymbol{\mu}}_0 = (0,0)$, $\vec{\boldsymbol{\mu}}_1 = (2,2)$			$\vec{\boldsymbol{\mu}}_0 = (0,0)$, $\vec{\boldsymbol{\mu}}_1 = (4,4)$		
		Delay	False alarm probability	Bayesian cost	Delay	False alarm probability	Bayesian cost	Delay	False alarm probability	Bayesian cost
Optimal	c=0.5	0.2412	0.8059	0.9265	0.6128	0.1707	0.4771	0.0269	0.0091	0.02255
	c=0.1	3.4536	0.1447	0.49	1.2526	0.0236	0.1489	0.054	0.0024	0.0078
	c=0.05	4.7335	0.0574	0.294	1.4577	0.011	0.0839	0.0015	0.0659	0.0048
	c=0.01	6.6182	0.01	0.0783	1.7059	0.0044	0.0215	0.1071	0.0006	0.0017
	c=0.005	7.1565	0.0064	0.0422	2.0608	0.0012	0.0115	0.1449	0.0004	0.0011
NMC-based method	c=0.5	0.1731	0.8521	0.9387	0.227	0.6139	0.5338	0.0495	0.01644	0.04115
	c=0.1	3.374	0.1668	0.5041	1.9943	0.00361	0.203	0.109	0.00711	0.018
	c=0.05	4.5207	0.0818	0.3077	2.0039	0.0341	0.1036	0.1269	0.00411	0.01045
	c=0.01	7.0666	0.01532	0.08567	2.0391	0.0042	0.02459	0.2018	0.0011	0.0031
	c=0.005	7.5891	0.0106	0.04855	2.3923	0.002	0.01396	0.2329	0.0005	0.00166
DQN-based method	c=0.5	0.2645	0.8648	0.9971	0.5649	0.2999	0.5823	0.0589	0.0196	0.049
	c=0.1	3.7571	0.2308	0.6066	1.3214	0.0977	0.2298	0.0418	0.0171	0.0213
	c=0.05	3.7968	0.2059	0.3958	1.4892	0.0339	0.1084	0.0629	0.0112	0.0144
	c=0.01	9.2189	0.0166	0.1088	2.8778	0.0383	0.0671	0.0968	0.0024	0.0034
	c=0.005	10.2965	0.0154	0.0668	3.6808	0.0176	0.036	0.493	0.0003	0.0028

Table III: Performances of the three QCD rules in the i.i.d. 2D Gaussian experiment: change in variance

		$\Sigma_0 = \boldsymbol{I}_2$, $\Sigma_1 = 2\boldsymbol{I}_2$			$\Sigma_0 = \boldsymbol{I}_2$, $\Sigma_1 = 4\boldsymbol{I}_2$		
		Delay	False alarm probability	Bayesian cost	Delay	False alarm probability	Bayesian cost
Optimal	c=0.1	2.6591	0.5368	0.8027	2.9756	0.0875	0.38296
	c=0.05	6.762	0.2553	0.5934	3.6813	0.0383	0.2224
	c=0.01	15.3527	0.03228	0.1858	4.7954	0.0058	0.05375
	c=0.005	18.2008	0.01242	0.1034	5.3881	0.0025	0.02944
	c=0.001	23.5036	0.0021	0.0256	5.9819	0.001	0.00698
NMC-based method	c=0.1	2.5955	0.5637	0.82325	3.1248	0.104	0.41458
	c=0.05	6.8413	0.2699	0.6119	3.9871	0.0396	0.2369
	c=0.01	15.5789	0.05033	0.206	5.1454	0.0048	0.05625
	c=0.005	18.6246	0.02835	0.1214	5.8751	0.0018	0.03117
	c=0.001	26.7755	0.007	0.03378	6.801	0.0006	0.0074
DQN-based method	c=0.1	2.4158	0.7391	0.9807	2.9831	0.2373	0.5357
	c=0.05	8.0369	0.4427	0.8445	5.6804	0.0681	0.3521
	c=0.01	16.9336	0.1606	0.3299	9.5317	0.019	0.1143
	c=0.005	23.093	0.0906	0.2061	11.5702	0.0021	0.06
	c=0.001	41.7379	0.0254	0.0671	18.4294	0.0009	0.0193

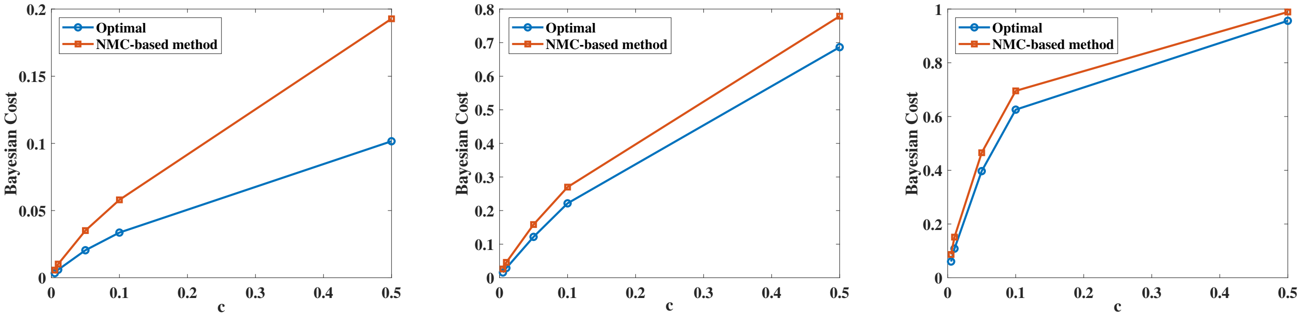
the optimal QCD solution, DQN-based method, and the NMC-based method under different Bernoulli settings on the test set. The results are shown in Fig. 2. From Fig. 2a, compared with the DQN-based method, the Bayesian costs of the NMC-based QCD method are generally closer to that of the optimal QCD method. As the unit delay cost decreases, the performance gap between the two solutions also decreases. Besides, by comparing Fig.2a and Fig.2b, we can see that when the KL-divergence between pre-change and post-change distributions gets smaller, i.e., the QCD task becomes harder, the costs of the three solutions increase. Although the Bayesian cost of the NMC-based solution increases as the QCD problem becomes harder, the performance gap between the NMC-based solution and the optimal solution is still small. In addition, Table I presents the more detailed performance of the three methods, including the delay and false alarm probability. In Table I, the DQN-based often achieves a lower delay or false alarm probability than the NMC-based method. But it still can not beat the NMC-based method on Bayesian cost. This result indicates that the NMC-based method achieves a better balance between the false alarm cost and delay cost than the DQN-based method. In addition, as the key of the NMC-based SCD rule, we also evaluate the estimation of the posterior probability $\Pi_t^{(0)}$. On the test set, the mean absolute errors of the posterior probability of the two experiment cases, i.e. the mean value of $|\hat{\Pi}_t^{(0)} - \Pi_t^{(0)}|$, are 0.0530 and 0.0935, respectively.

In the second numerical example, we study the performance of the three QCD methods when the observations are continuous random variables. The observations in this experiment are 2-D Gaussian random variables with $f_0 = \mathcal{N}(\vec{\boldsymbol{\mu}}_0, \boldsymbol{\Sigma}_0)$ and $f_1 = \mathcal{N}(\vec{\boldsymbol{\mu}}_1, \boldsymbol{\Sigma}_1)$. To illustrate the performance of the NMC-based QCD method facing different kinds of changes, we study 2 different cases : (1). The change happens to the mean vector of the 2-D Gaussian distribution; (2) The change happens to the covariance matrix of the 2-D Gaussian distribution. For



(a) $\Sigma_0 = \boldsymbol{I}_2$, $\Sigma_1 = 2\boldsymbol{I}_2$ (b) $\Sigma_0 = \boldsymbol{I}_2$, $\Sigma_1 = 4\boldsymbol{I}_2$

Figure 4: The Bayesian costs of the Gaussian QCD experiment with change in covariance



(a) $\vec{\boldsymbol{\mu}}_{0,train} = (0,0)$, $\vec{\boldsymbol{\mu}}_{1,train} = (4,4)$ $\vec{\boldsymbol{\mu}}_{0,test} = (0,0)$, $\vec{\boldsymbol{\mu}}_{1,test} = (3.2,3.2)$ (b) $\vec{\boldsymbol{\mu}}_{0,train} = (0,0)$, $\vec{\boldsymbol{\mu}}_{1,train} = (2,2)$ $\vec{\boldsymbol{\mu}}_{0,test} = (0,0)$, $\vec{\boldsymbol{\mu}}_{1,test} = (1.6,1.6)$ (c) $\vec{\boldsymbol{\mu}}_{0,train} = (0,0)$, $\vec{\boldsymbol{\mu}}_{1,train} = (1,1)$ $\vec{\boldsymbol{\mu}}_{0,test} = (0,0)$, $\vec{\boldsymbol{\mu}}_{1,test} = (0.8,0.8)$

Figure 5: Robustness test: testing on 2D Gaussian data which has mean vector different from the training data

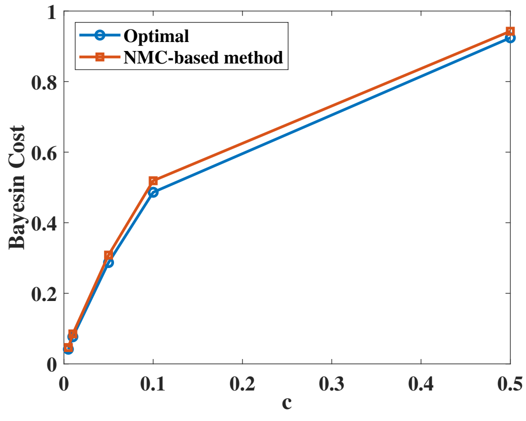


Figure 6: Robustness test: Change from Gaussian distribution to logistic distribution

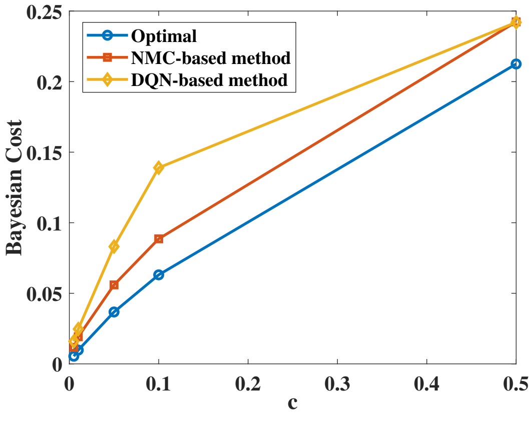


Figure 7: Bayesian costs of 10D Gaussian data

the first case, we set $\boldsymbol{\Sigma}_0 = \boldsymbol{\Sigma}_1 = \boldsymbol{I}_2$ and $\boldsymbol{\mu}_0 = (0,0)$. Then we carry out three experiments with $\boldsymbol{\mu}_1 = (1,1)$, $\boldsymbol{\mu}_1 = (2,2)$, and $\boldsymbol{\mu}_1 = (4,4)$, respectively. In addition, P_λ, is a geometric distribution with parameter $\rho = 0.01$. These three NMC-based models are denoted as model A, B and C. In Fig. 3 and Table II, we compare the performance of the three QCD methods in this mean vector QCD problem. On the test set, the mean absolute errors of the posterior probability of the three models are 0.1321, 0.0319 and 0.0315, respectively. For the second case, we set $\boldsymbol{\mu}_0 = \boldsymbol{\mu}_1 = (0,0)$ and $\boldsymbol{\Sigma}_0 = \boldsymbol{I}_2$. Then we implement two experiments with $\boldsymbol{\Sigma}_1 = 2\boldsymbol{I}_2$, $\boldsymbol{\Sigma}_1 = 4\boldsymbol{I}_2$. In Fig. 4 and Table III, we compare the performance of the three QCD methods in this variance QCD problem. On the test set, the mean absolute errors of the posterior probability of the two cases are 0.1320 and 0.1382, respectively. In Fig. 3 and

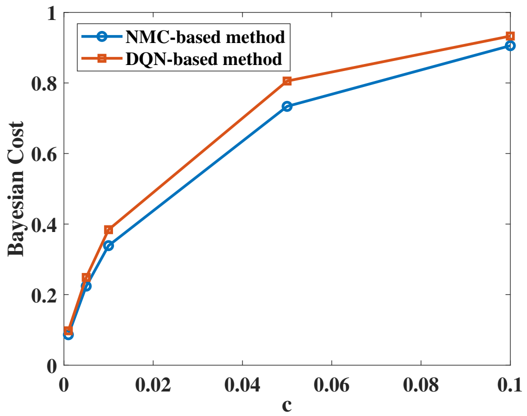
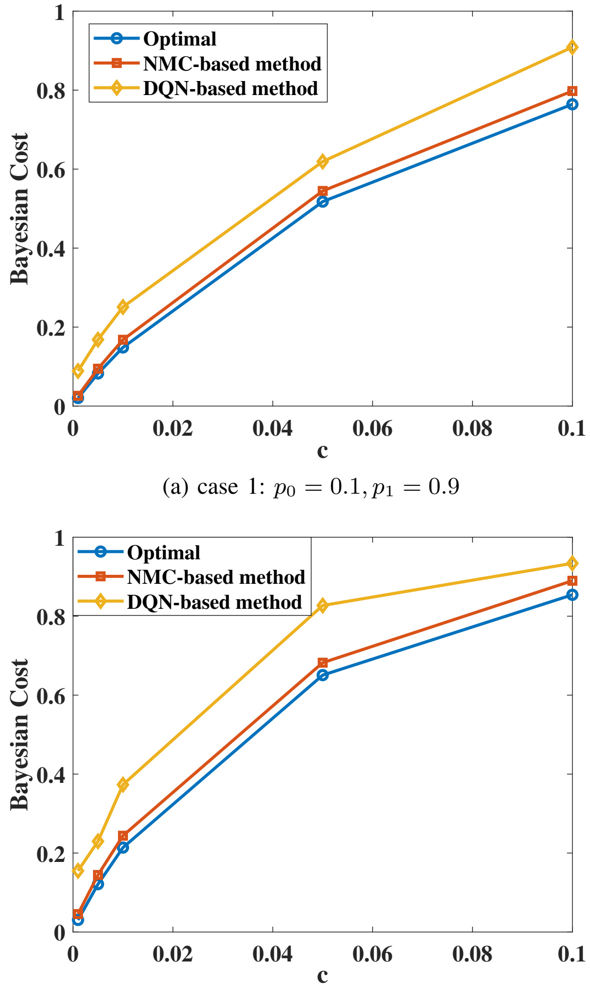


Figure 8: Bayesian costs of the NMC-based method and the DQN-based method in Non-i.i.d. case

4, the Bayesian costs achieved by the NMC-based method are close to the costs of the optimal QCD rule and are lower than the costs of the QCD-based methods. These results validate our conclusion that the NMC-based QCD method has a good performance for the 2D continuous QCD problem. From Table II and III, we can also see that the NMC-based method performs better in balancing the false alarm probability and delay costs than the DQN-based method. Next, we conduct another two experiments to test the robustness of the NMC-based method in the 2D Gaussian experiment. Firstly, we test the performance of models A, B, and C on data whose mean vector is different from the training data. Concretely, the mean vectors of the testing data for Model A, B and C are $(3.2, 3.2)$, $(1.6, 1.6)$ and $(0.8, 0.8)$, receptively. Other parameters of the testing data distributions are the same as the training data. The Bayesian costs of the optimal QCD rules and the NMC-based model A, B, and C obtained on the testing data are shown in Fig. 5. Secondly, we investigate the performance of the NMC-based model C on data whose post-change distribution is non-Gaussian. For the testing data distribution, we define a 2-D distribution, $F_L(1, 1)$ as the post-change distribution of the testing data. With $F_L(1, 1)$, the two elements in each data sample are independent and follow the Logistics distributions, $L(1, \sqrt{3/2\pi})$. The other parameters of the testing data are the same as the training data of NMC-based model C. The Bayesian costs of the optimal QCD rules and the NMC-based model C are shown in Fig. 6. In Fig. 5 and 6, the optimal solutions are obtained using the distribution of the testing data. From the results shown in Fig. 5 and 6, we can see that the performance of the NMC-based method is close to the optimal QCD rules for data generated with distributions different from the training data. These results indicate that the NMC-based QCD rule is robust to the instability of data.

In addition, we also implement an experiment for high dimensional data with a more complicated covariance matrix. Here, the observation data follows a 10D Gaussian distribution with mean vector $\boldsymbol{\mu}_1 = (1, ..., 1)$, $\boldsymbol{\mu}_1 = (0, ..., 0)$. The covariance matrix is a randomly generated positive-definite matrix. The width of the sliding window in this experiment is set as 1. The Bayesian costs of the optimal rule and the NMC-based method are shown in Fig. 7. From this figure, we can see that the NMC-based QCD method works well for high-dimensional data. On the test set, the mean absolute error of the posterior probability estimated by the NMC approximation model is 0.0887.



(a) case 1: $p_0 = 0.1, p_1 = 0.9$

(b) case 2: $p_0 = 0.2, p_1 = 0.8$

Figure 9: The Bayesian costs of the NMC-based and optimal QCD solutions in the HMM case

B. QCD experiments in non-i.i.d. case

To evaluate the performance of the NMC-based method with non-i.i.d. data, two numerical examples are conducted.

In the first numerical example, the data samples are generated by the Markov Gaussian sequence

$$x_t = \begin{cases} 0.5x_{t-1} + \epsilon_t, & \text{if} \quad \text{t} < \lambda \\ -0.5x_{t-1} + \epsilon_t, & \text{if} \quad \text{t} \geq \lambda \end{cases}$$

where $\epsilon_t \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, 1)$ for $t > 0$. Since the distribution of the current data sample only depends on the last data sample, we set $w = 2$ for both the NMC-based method and the DQN-based method. The Bayesian cost of the NMC-based rule and the QCD-based rule are shown in Fig. 8. From the results, we

Table IV: Performances of the three QCD rules in the HMM experiment

		case 1: $p_0 = 0.1, p_1 = 0.9$			case 2: $p_0 = 0.2, p_1 = 0.8$		
		Delay	False alarm probability	Bayesian cost	Delay	False alarm probability	Bayesian cost
Optimal	c=0.1	3.1978	0.4444	0.7642	2.0889	0.6452	0.8541
	c=0.05	6.9195	0.1717	0.5177	6.6656	0.3174	0.6507
	c=0.01	12.0809	0.0276	0.1484	17.3519	0.0404	0.2139
	c=0.005	14.4003	0.0105	0.0825	19.9785	0.0215	0.1214
	c=0.001	18.0703	0.0023	0.0204	28.3053	0.0024	0.0307
NMC-based method	c=0.1	3.4926	0.4485	0.7978	2.3752	0.6522	0.8897
	c=0.05	6.7486	0.2073	0.54473	6.3758	0.3634	0.68219
	c=0.01	14.6158	0.0217	0.1679	17.5915	0.0681	0.2440
	c=0.005	15.4936	0.0171	0.09457	21.1274	0.0389	0.1445
	c=0.001	20.4434	0.0055	0.02594	34.2069	0.0114	0.0456
DQN-based method	c=0.1	3.2805	0.5808	0.9089	1.7592	0.7579	0.9338
	c=0.05	7.5862	0.2399	0.6192	6.1065	0.5217	0.827
	c=0.01	14.1083	0.1097	0.2508	14.8038	0.2252	0.3733
	c=0.005	21.4553	0.06093	0.1682	20.7133	0.1262	0.2298
	c=0.001	80.0278	0.00877	0.0888	15.701	0.1394	0.155

can see that the NMC-based QCD method outperforms the DQN-based methods in this non-i.i.d. QCD problem.

In the second example, we study the performance of the NMC-based method when the data follows an HMM. There are two hidden states in the HMM. For state 1: the data will generate data following Bernoulli distribution with parameter 0.1. For state 2: the data will generate data following a Bernoulli distribution with parameter 0.9. The change happens to the transition probability of the HMM. Before the change, the transition probability between states 1 and 2 is p_0. After change, the transition probability becomes to p_1. In addition, P_λ, is a geometric distribution with parameter $\rho = 0.01$. In the HMM experiment, we set the width of the sliding window $w = 15$ for both the NMC-based method and the DQN-based method. In Fig. 9 and Table IV, we compare the performances of the NMC-based QCD method and the optimal solution with different values of p_0 and p_1. Similar to the results of the i.i.d experiments, the performance of the NMC-based QCD method is still generally closer to the optimal QCD rule than the QCD method. In addition, the mean absolute errors of the posterior probability of the two experiment cases are 0.0957 and 0.1244, respectively.

VII. Conclusion

In this paper, we have studied the online data-driven Bayesian QCD problem with geometrically distributed change points. Inspired by the key role that the posterior false alarm probability plays in the i.i.d. QCD problem, we have proposed an NMC-based QCD rule for the data-driven Bayesian QCD problem. Trained by the Gradient Monte Carlo algorithm, a randomized neural network is applied to approximate the posterior false alarm probability. By comparing the posterior false alarm probability with a well-chosen threshold, we obtain an NMC-based QCD rule. This NMC-based method works not only for the i.i.d. QCD problem, but also for the HMM QCD problem or the more general non-i.i.d. QCD problems. Moreover, this NMC-based method is guaranteed to converge. Numerical results have been carried out to evaluate the performance of the NMC-based method. The results have validated that the performance of the DQN-based QCD solution is generally better than the DQN-based method and close to the performance of the optimal solution.

References

[1] A. G. Tartakovsky, B. L. Rozovskii, R. B. Blazek, and H. Kim, "A novel approach to detection of intrusions in computer networks via adaptive sequential and batch-sequential change-point detection methods," *IEEE Transactions on Signal Processing*, vol. 54, no. 9, pp. 3372–3382, Sept. 2006.

[2] Y. C. Chen, T. Banerjee, A. D. Domínguez-García, and V. V. Veeravalli, "Quickest line outage detection and identification," *IEEE Transactions on Power Systems*, vol. 31, no. 1, pp. 749–758, Feb. 2015.

[3] G. Rovatsos, X. Jiang, A. D. Domínguez-García, and V. V. Veeravalli, "Statistical power system line outage detection under transient dynamics," *IEEE Transactions on Signal Processing*, vol. 65, no. 11, pp. 2787–2797, Feb. 2017.

[4] P. Yang, G. Dumont, and J. M. Ansermino, "Adaptive change detection in heart rate trend monitoring in anesthetized children," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 11, pp. 2211–2219, 2006.

[5] A. P. and A. S. P., "Real-time financial surveillance via quickest change-point detection methods," *arXiv: Applications*, 2015.

[6] T. Lau, W. Tay, and V. V. Veeravalli, "Quickest change detection with unknown post-change distribution," in *Proc. on IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* New Orleans, LA, Mar. 2017, pp. 3924–3928.

[7] J. Geng and L. Lai, "Quickest change-point detection over multiple data streams via sequential observations," in *Proc. on IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* Calgary, Canada, Sept. 2018, pp. 4404–4408.

[8] T. Banerjee and V. V. Veeravalli, "Data-efficient minimax quickest change detection," in *Proc. on IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* Kyoto, Japan, Mar. 2012, pp. 3937–3940.

[9] Y. Xie and D. Siegmund, "Sequential multi-sensor change-point detection," in *Proc. on IEEE Information Theory and Applications Workshop*. San Diego, CA, May 2013, pp. 1–20.

[10] H. V. Poor and O. Hadjiliadis, *Quickest detection*. Cambridge University Press, 2008.

[11] Y. Wang and Y. Mei, "Large-scale multi-stream quickest change detection via shrinkage post-change estimation," *IEEE Transactions on Information Theory*, vol. 61, no. 12, pp. 6926–6938, Dec. 2015.

[12] A. N. Shiryaev, *Optimal stopping rules*. Springer Science & Business Media, 2007, vol. 8.

[13] E. S. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, no. 1/2, pp. 100–115, 1954.

[14] G. V. Moustakides, “Optimal stopping times for detecting changes in distributions,” *the Annals of Statistics*, vol. 14, no. 4, pp. 1379–1387, 1986.

[15] T. L. Lai and H. Xing, “Sequential change-point detection when the pre- and post-change parameters are unknown,” *Sequential Analysis*, vol. 29, no. 2, pp. 162–175, Apr. 2010.

[16] Y. Mei, “Suboptimal properties of Page’s CUSUM and Shiryayev-Roberts procedures in change-point problems with dependent observations,” *Statistica Sinica*, pp. 883–897, Jul. 2006.

[17] S. Dayanik and C. Goulding, “Sequential detection and identification of a change in the distribution of a markov-modulated random sequence,” *IEEE Transactions on Information Theory*, vol. 55, no. 7, pp. 3323–3345, 2009.

[18] H. Li, “Data driven quickest change detection: An algorithmic complexity approach,” in *Proc. IEEE International Symposium on Information Theory*. Barcelona, Spain, 2016, pp. 21–25.

[19] M. N. Kurt, Y. Yılmaz, and X. Wang, “Real-time nonparametric anomaly detection in high-dimensional settings,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 7, pp. 2463–2479, 2020.

[20] T. Flynn, O. Hadjiliadis, I. Stamos, and F. J. Vázquez-Abad, “Data driven stochastic approximation for change detection,” in *Proc. on IEEE Winter Simulation Conference (WSC)*. Las Vegas, NV, Dec. 2017, pp. 2279–2290.

[21] Y. Cao, L. Xie, Y. Xie, and H. Xu, “Sequential change-point detection via online convex optimization,” *Entropy*, vol. 20, no. 2, p. 108, 2018.

[22] H. Chen, “Sequential change-point detection based on nearest neighbors,” *The Annals of Statistics*, vol. 47, no. 3, pp. 1381–1407, 2019.

[23] M. N. Kurt, O. Ogundijo, C. Li, and X. Wang, “Online cyber-attack detection in smart grid: A reinforcement learning approach,” *IEEE Transactions on Smart Grid*, vol. 10, no. 5, pp. 5174–5185, Sept. 2018.

[24] X. Ma, L. Lai, and S. Cui, “A deep Q-network based approach for online bayesian change point detection,” in *Proc. on IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*. Golden Coast, Australia, Oct. 2021, pp. 1–6.

[25] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, 2018.

[26] G.-B. Huang, L. Chen, C. K. Siew *et al.*, “Universal approximation using incremental constructive feedforward networks with random hidden nodes,” *IEEE Trans. Neural Networks*, vol. 17, no. 4, pp. 879–892, 2006.

[27] G.-B. Huang and L. Chen, “Convex incremental extreme learning machine,” *Neurocomputing*, vol. 70, no. 16-18, pp. 3056–3062, 2007.