Machine Learning Control Charts for Monitoring Spatio-Temporal Data Streams
(Running Title: Machine Learning Charts for Monitoring Spatial Data)

Yuhang Zhou[1], Ran Jin[2] and Peihua Qiu[1]

[1]Department of Biostatistics, University of Florida, Gainesville, FL 32610

[2]Grado Department of Industrial and Systems Engineering, Virginia Tech, Blacksburg, VA 24061

Correspondence: Peihua Qiu, Email: pqiu@ufl.edu

Funding: The authors received no specific funding for this research.

Abstract

Emerging research underscores the potential of machine learning techniques to enhance statistical process control (SPC) methodologies. However, most existing machine learning-based charts assume that in-control process observations are independent and identically distributed, a condition rarely met in practice, which limits their real-world applicability. This paper suggests a framework that integrates a sequential spatio-temporal decorrelation procedure with representative machine learning charting schemes, enabling effective monitoring of spatio-temporal data streams. Numerical studies demonstrate that the proposed control charts significantly outperform traditional methods, providing superior monitoring capabilities for spatio-temporal data with complex structures.

Keywords: Correlation; Control charts; Machine learning algorithms; Random forest; Spatio-temporal data; Support vector machines.

1 Introduction

Machine learning methods have garnered significant attention across various research fields in recent years (Aggarwal 2018; Breiman 2001; Carvalho et al. 2019; Hastie et al. 2001; Göb 2006). Within the domain of statistical process control (SPC), researchers have developed control charts that leverage machine learning methods such as k-nearest neighbor (KNN), random forest (RF), and support vector machine (SVM) approaches (cf. Megahed and Jones-Farmer 2015, Qiu 2024a). These machine learning-based control charts are typically designed for monitoring univariate or multivariate data streams. However, in applications such as environmental monitoring, disease surveillance, and others, data streams are often spatial, meaning that process observations are collected at multiple locations over time (Qiu 2024b). This paper aims to advance machine learning-based control charts to effectively monitor such spatial data streams.

Sequential monitoring of a data stream can be framed as a sequential classification problem, where the status of the underlying process must be classified as either in-control (IC) or out-of-control (OC) at each time point. Applying supervised machine learning algorithms, such as RF, to this problem requires a training dataset containing both IC and OC observations. However, in typical SPC applications (e.g., online monitoring of a production line and layerwise Additive Manufacturing), only a small IC dataset is available prior to online process monitoring. To address this limitation, Tuv and Runger (2003) proposed the artificial contrast (AC) approach, which generates an artificial OC dataset from a pre-specified distribution (e.g., a uniform distribution) and combines it with IC data to derive a classification rule using the RF algorithm. Deng et al. (2012) introduced an alternative method, the real-time contrast (RTC) approach, which uses recent observations within a time window as OC data. These OC data are then combined with the IC dataset to establish a classification rule. Jin and Liu (2013) proposed an ensemble learning-based multimodal process monitoring method, which used the piecewise linear regression tree models for baseline modeling and generate regression-adjusted monitoring statistics.

Another strategy, proposed by Sun and Tsung (2003), involves using a one-class classification (OCC) method based on the support vector data description approach. This method identifies the boundary of the IC dataset as the decision rule for process monitoring: a future observation is deemed IC if it falls within the boundary, and OC otherwise. Various improvements and generalizations of OCC-based control charts have since been developed (e.g., He et al. 2018, Sukchotrat et al. 2010, Xie and Qiu 2023a), with additional references cited therein.

Additionally, Yeganeh et al. (2022a) integrated a radial basis function (RBF) neural network with a multivariate exponentially weighted moving average (MEWMA) chart to enhance shift detection in linear profile monitoring. In a related study, Yeganeh et al. (2022b) incorporated an ensemble of artificial neural networks (ANNs) with a nonparametric EWMA chart to improve the online monitoring of nonparametric profile changes. Li et al. (2021a) applied transfer learning to enhance process monitoring by leveraging data from different sources, while Li et al. (2021b) developed a nonparametric multivariate control chart using the KNN algorithm.

Most machine learning-based control charts discussed above are designed under the assumption that in-control (IC) process observations at different time points are independent, identically distributed, and follow a normal distribution. When one or more of these assumptions are violated, these methods become unreliable, as highlighted in the SPC literature (e.g., Capizzi and Masarotto 2008, Chakraborti et al. 2015, Qiu 2018). To address these challenges, some existing methods focus on monitoring correlated data using parametric time series modeling (e.g., Capizzi and Masarotto 2008) or nonparametric moment estimation of covariances (e.g., Qiu et al., 2020). Some others focus on nonparametric or distribution-free process monitoring (Capizzi 2015, Chakraborti and Graham 2019, Qiu and Hawkins 2001). Building on these approaches and the approach of self-starting process monitoring (Hawkins 1987), Qiu and Xie (2022) proposed a general framework, known as Transparent Sequential Learning (TSL), for monitoring processes with correlated data. Subsequently, Xie and Qiu (2023b, 2024) extended the TSL framework to accommodate dynamic processes with time-varying IC distributions.

This paper focuses on the online monitoring of spatio-temporal data streams, which often exhibit complex variations, correlations, and distributions over space and time. The machine learning-based control charts discussed earlier cannot be directly applied to such data streams, as their underlying model assumptions are typically invalid in these scenarios. To address this limitation, we propose modifications to several representative machine learning-based control charts using the sequential spatio-temporal data standardization and decorrelation procedure introduced by Yang and Qiu (2020). Numerical results demonstrate that the modified charts significantly outperform their original counterparts when monitoring spatio-temporal data across various scenarios. It is worth noting that while the control charts discussed in this paper primarily focus on detecting mean shifts, they are generally sensitive to other types of shifts as well. These include changes in variance, simultaneous shifts in both mean and variance, and some more complex distributional changes, thereby enhancing their applicability in spatio-temporal process monitoring.

The remainder of the paper is organized as follows. Section 2 outlines the proposed modifications to existing machine learning-based control charts. Section 3 presents a numerical performance comparison between the modified and original charts. Section 4 applies the proposed methods to a real-world case study. Finally, Section 5 concludes the paper with key remarks and future directions.

2 Monitoring Spatio-Temporal Data Streams Using Machine Learning Approaches

In this section, we propose a data pre-processing framework for enhancing the performance of existing machine learning-based control charts when monitoring spatio-temporal data streams. A sequential spatio-temporal data standardization and decorrelation procedure is first described in Section 2.1 to pre-process the observed data. Then, several representative machine learning-based control charts are described in Section 2.2 to sequentially monitor the pre-processed data.

2.1 Sequential data standardization and decorrelation

Assume that an IC spatio-temporal dataset is available in advance and the observed data in this dataset follow the nonparametric spatio-temporal model:

$$y(t_i, s_{ij}) = \mu(t_i, s_{ij}) + \epsilon(t_i, s_{ij}), \quad \text{for } i = 1, \ldots, n,\ j = 1, \ldots, m_i, \tag{1}$$

where $y(t_i, s_{ij})$ is the observation at time $t_i \in [0, T]$ and location $s_{ij} \in \Omega$, $\mu(t_i, s_{ij})$ is its mean, and $\epsilon(t_i, s_{ij})$ is a zero-mean random error. In Model (1), $[0, T]$ denotes one whole season for applications where seasonality is present. The spatio-temporal correlation is described by the covariance function

$$V(t, t'; s, s') = \text{Cov}(y(t, s), y(t', s')), \quad \text{for any } t, t' \in [0, T],\ s, s' \in \Omega.$$

As discussed in Yang and Qiu (2018), the mean function $\mu(t, s)$ in Model (1) can be estimated using the following spatio-temporal local linear kernel smoothing procedure:

$$\begin{aligned} \arg\min_{\boldsymbol{\theta} \in R^4} \sum_{i=1}^{n} \sum_{j=1}^{m_i} & [y(t_i, s_{ij}) - \theta_\mu - \theta_t(t_i - t) - \theta_u(s_{u,ij} - s_u) - \theta_v(s_{v,ij} - s_v)]^2 \\ & \times K_t\left(\frac{t_i - t}{h_t}\right) K_s\left(\frac{d_E(s_{ij}, s)}{h_s}\right), \end{aligned} \tag{2}$$

where $\boldsymbol{\theta} = (\theta_\mu, \theta_t, \theta_u, \theta_v)^T$, $s = (s_u, s_v)^T$, $d_E(\cdot, \cdot)$ is the Euclidean distance, $h_t, h_s > 0$ are two bandwidths, and $K_t(\cdot)$ and $K_s(\cdot)$ are two kernel functions. As suggested in Yang and Qiu (2018), both $K_t(\cdot)$ and $K_s(\cdot)$ are chosen to be the Epanechnikov kernel function $K_t(u) = K_s(u) = 0.75(1 - u^2)I(|u| \leq 1)$ because of its good theoretical properties (Epanechnikov 1969), and the bandwidths are chosen by a modified cross-validation procedure. Since the minimization problem (2) can be regarded as a weighted least squares estimation, the estimate of $\mu(t, s)$ has the expression:

$$\widehat{\mu}(t, s) = \zeta^\top (\boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{Y}, \tag{3}$$

where $\boldsymbol{\zeta} = (1, 0, 0, 0)^\top$, $\boldsymbol{X}$ is the related design matrix of the observations, $\boldsymbol{W}$ is a diagonal weight matrix, and $\boldsymbol{Y}$ is the vector of the observed data.

By the approach discussed in Yang and Qiu (2019), the covariance function $V(t, t'; s, s')$ in Model (1) can be estimated by the following kernel estimate: when $(t, s) \neq (t', s')$,

$$\widehat{V}(t, t'; s, s') = \frac{\sum_{i=1}^n \sum_{j=1}^{m_i} \sum_{k=1}^n \sum_{l=1}^{m_k} \widehat{\epsilon}(t_i, s_{ij}) \widehat{\epsilon}(t_k, s_{kl}) w_v(i, j, k, l; t, t', s, s')}{\sum_{i=1}^n \sum_{j=1}^{m_i} \sum_{k=1}^n \sum_{l=1}^{m_k} w_v(i, j, k, l; t, t', s, s')}, \tag{4}$$

where $\widehat{\epsilon}(t_i, s_{ij}) = y(t_i, s_{ij}) - \widehat{\mu}(t_i, s_{ij})$ are residuals, and

$$w_v(i, j, k, l; t, t', s, s') = K_t\left(\frac{t_i - t}{h_t}\right) K_s\left(\frac{d_E(s_{ij}, s)}{h_s}\right) K_t\left(\frac{t_k - t'}{h_t}\right) K_s\left(\frac{d_E(s_{kl}, s')}{h_s}\right).$$

When $(t, s) = (t', s')$, $V(t, t; s, s)$ is actually the variance of $y(t, s)$, denoted as $\sigma^2(t, s)$. Its estimate can be defined similarly to be

$$\widehat{\sigma}^2(t, s) = \frac{\sum_{i=1}^n \sum_{j=1}^{m_i} \widehat{\epsilon}^2(t_i, s_{ij}) w_\sigma(i, j; t, s)}{\sum_{i=1}^n \sum_{j=1}^{m_i} w_\sigma(i, j; t, s)}, \tag{5}$$

where

$$w_\sigma(i, j; t, s) = K_t\left(\frac{t_i - t}{h_t}\right) K_s\left(\frac{d_E(s_{ij}, s)}{h_s}\right).$$

Under some regularity conditions, Yang and Qiu (2018, 2019) confirmed that the estimates $\widehat{\mu}(t, s)$, $\widehat{V}(t, t'; s, s')$ and $\widehat{\sigma}^2(t, s)$ are all statistically consistent.

Next, we discuss online monitoring of spatio-temporal data streams. Assume that the observed spatio-temporal data at times $\{t_i^* \in (T, \infty), i \geq 1\}$ and locations $\{s_{ij}^* \in \Omega, j = 1, \ldots, m_i^*\}$

are $\{y(t_i^*, s_{ij}^*), j = 1, \ldots, m_i^*, i \geq 1\}$. To sequentially monitor them, the observed data should be properly standardized and decorrelated first, using the estimated model of (1). To this end, the estimated regular longitudinal pattern described by the estimates $\widehat{\mu}(t,s)$, $\widehat{\sigma}^2(t,s)$, and $\widehat{V}(t,t';s,s')$ should be extended periodically from $[0,T]$ to (T,∞) to represent the ongoing IC pattern. Then, the sequential data standardization and decorrelation procedure is described below.

- At the first time point t_1^*, the standardized observation is defined to be

$$\widehat{\boldsymbol{e}}(t_1^*) = \widehat{\Sigma}_1^{-1/2}\widehat{\boldsymbol{\epsilon}}(t_1^*),$$

  where $\widehat{\Sigma}_1$ is the estimate of $\Sigma_1 = \mathrm{Cov}(\mathbf{Y}(t_1^*), \mathbf{Y}(t_1^*))$ derived from the estimated covariance function $\widehat{V}(t,t';s,s')$, $\mathbf{Y}(t_1^*) = (y(t_1^*, s_{11}^*), \ldots, y(t_1^*, s_{1m_1^*}^*))^T$, $\widehat{\boldsymbol{\epsilon}}(t_1^*) = \mathbf{Y}(t_1^*) - \widehat{\boldsymbol{\mu}}(t_1^*)$, and $\widehat{\boldsymbol{\mu}}(t_1^*) = (\widehat{\mu}(t_1^*, s_{1j}^*), \ldots, \widehat{\mu}(t_1^*, s_{1m_1^*}^*))^T$.

- At time t_i^*, for $i \geq 2$, let $\mathbf{Y}(t_i^*) = (y(t_i^*, s_{i1}^*), \ldots, y(t_i^*, s_{im_i^*}^*))^T$, $\mathbf{Y}_{i-1} = (\mathbf{Y}^T(t_1^*), \ldots, \mathbf{Y}^T(t_{i-1}^*))^T$, $\widehat{\boldsymbol{\epsilon}}(t_i^*) = (\widehat{\epsilon}(t_i^*, s_{i1}^*), \ldots, \widehat{\epsilon}(t_i^*, s_{im_i^*}^*))^T$, and $\widehat{\boldsymbol{\epsilon}}_{i-1} = (\widehat{\boldsymbol{\epsilon}}^T(t_1^*), \ldots, \widehat{\boldsymbol{\epsilon}}^T(t_{i-1}^*))^T$. Then, the standardized and decorrelated data at t_i^* is defined to be

$$\widehat{\boldsymbol{e}}(t_i^*) = \widehat{\Sigma}_{ii\cdot(i-1)}^{-1/2}\left(\widehat{\boldsymbol{\epsilon}}(t_i^*) - \widehat{\Sigma}_{i-1,i}^T\widehat{\Sigma}_{i-1,i-1}^{-1}\widehat{\boldsymbol{\epsilon}}_{i-1}\right),$$

  where $\widehat{\Sigma}_{ii\cdot(i-1)} = \widehat{\Sigma}_i - \widehat{\Sigma}_{i-1,i}^T\widehat{\Sigma}_{i-1,i-1}^{-1}\widehat{\Sigma}_{i-1,i}$, $\widehat{\Sigma}_i$ is the estimate of covariance matrix of $\mathbf{Y}_i$, $\widehat{\Sigma}_{i-1,i}$ is the estimate of $\mathrm{Cov}(\mathbf{Y}_{i-1}, \mathbf{Y}(t_i^*))$, $\widehat{\Sigma}_{i-1,i-1}$ is the estimate of $\mathrm{Cov}(\mathbf{Y}_{i-1}, \mathbf{Y}_{i-1})$, and all these variance/covariance estimates can be computed from the estimated covariance function $\widehat{V}(t,t';s,s')$.

After this data standardization and decorrelation procedure is applied to the observed data, we obtain the transformed observations $\{\widehat{e}(t_i^*, s_{ij}^*), j = 1, \ldots, m_i^*, i \geq 1\}$ which have the properties that i) the transformed observations at each time point are asymptotically uncorrelated with each other, ii) each of them has the asymptotic variance of 1, and iii) they are asymptotically uncorrelated with all observations at previous observation times.

2.2 Spatio-temporal process monitoring using machine learning approaches

As discussed in Section 1, most existing machine learning-based control charts are designed for monitoring processes with uncorrelated IC observations at different observation times. After data standardization and decorrelation as discussed in the previous subsection, they can be applied to the transformed observations $\{\widehat{e}(t_i^*, s_{ij}^*), j = 1, \ldots, m_i^*, i \geq 1\}$ for online process monitoring, which is described below for five representative machine learning-based control charts.

2.2.1 Control chart based on the artificial contrasts

Tuv and Runger (2003) first introduced the concept of *artificial contrast* (AC) to address the common challenge to use supervised machine learning methods for online process monitoring that only a small amount of IC data are available in some applications before online process monitoring. The key idea is to generate an artificial dataset from an off-target (e.g., Uniform) distribution, defined in the same domain as that of the IC data, and treat these artificially generated data as OC observations. After combining the IC dataset, denoted as $\mathcal{D}_{IC}$, with the AC dataset, denoted as $\mathcal{D}_{AC}$, a machine learning algorith (e.g., RF) can be used to derive a classification rule for determining the process status (IC versus OC) at any time during online process monitoring. However, such control charts have the following main limitation: the decision at each time point during process monitoring is made based solely on the current observations, without leveraging past process information. To overcome this limitation, Hu and Runger (2010) proposed a two-step modified version:

i) **Log-Likelihood Ratio Calculation:** Let $\widehat{\boldsymbol{e}}(t_i^*) = (\widehat{e}(t_i^*, s_{i1}^*), \ldots, \widehat{e}(t_i^*, s_{im_i^*}^*))^T$ be the standardized and decorrelated data at the current observation time t_i^*, for $i \geq 1$. The RF classifier can provide estimated probabilities $\widehat{p}_0(\widehat{\boldsymbol{e}}(t_i^*))$ and $\widehat{p}_1(\widehat{\boldsymbol{e}}(t_i^*))$ that the observation vector $\widehat{\boldsymbol{e}}(t_i^*)$ belongs to the IC and OC classes, respectively. Define the log-likelihood ratio at time t_i^* to be

$$\ell_i = \log\left[\widehat{p}_1(\widehat{\boldsymbol{e}}(t_i^*))\right] - \log\left[\widehat{p}_0(\widehat{\boldsymbol{e}}(t_i^*))\right].$$

ii) **EWMA-Based Charting Statistic:** An EWMA chart is then constructed by

$$E_i = \lambda(\ell_i - \mu_\ell)/\sigma_\ell + (1-\lambda)E_{i-1}, \qquad \text{for } i \geq 1,$$

where $E_i = 0$, μ_ℓ and σ_ℓ are the IC mean and standard deviation of ℓ_i that can be estimated from the IC data, and $\lambda \in (0, 1]$ is a smoothing parameter. This chart is referred to as the AC-STD chart hereafter, where AC denotes "artificial contrast" and STD implies that the chart is applied to the spatio-temporally decorrelated data.

The AC-STD chart gives a signal at time t_i^* if $E_i > h_{AC}$, where $h_{AC} > 0$ is a control limit. To determine h_{AC}, Hu and Runger (2010) recommended using the following 10-fold cross-validation (CV) procedure:

- Partition $\mathcal{D}_{IC}$ and $\mathcal{D}_{AC}$ into 10 roughly equal subsets, respectively.
- For each specific fold, train the RF classifier using 90% of $\mathcal{D}_{IC}$ and $\mathcal{D}_{AC}$, and apply the resulting AC-STD chart with a trial control limit h_{AC} to the remaining 10% of the data.
- Repeat this procedure for a total of C times (e.g., $C = 1,000$) to obtain multiple run length (RL) estimates.
- The average of these RL values approximates the true ARL_0 for the given h_{AC}. Search for the h_{AC} value using a numerical algorithm such as the bisection method until a pre-specified ARL_0 value is achieved.

In the AC-STD chart, the RF algorithm includes several key parameters that could influence its performance. They are specified in the following way. The maximum depth of each tree is allowed to grow until purity, meaning that every terminal node contains observations from a single class (i.e., either IC or OC). Additionally, the minimum node size is set to be 1. Since increasing the number of trees would reduce variance without causing overfitting, the number is typically set to a sufficiently large value, and the exact number should be chosen to minimize the overall misclassification error. Hu and Runger (2010) suggested using 500-1,000. We use 500 in all numerical examples in the paper.

2.2.2 Control chart based on the real time contrasts

The AC dataset $\mathcal{D}_{AC}$ used in the AC chart relies on an off-target distribution that may not accurately represent the actual OC behavior well. To address this limitation, Deng et al. (2012) proposed the *real-time contrast* (RTC) approach. In RTC, rather than relying on a static, artificially generated AC dataset, the contrasts are dynamically constructed from recent observations. Specifically, the IC dataset $\mathcal{D}_{IC}$ is first partitioned into two subsets: a randomly selected subset $\mathcal{D}_{IC_0}$, of size N_0, and the remaining IC dataset $\mathcal{D}_{IC_1}$. The subset $\mathcal{D}_{IC_1}$ is reserved for determining the control limit. At the current time point t_i^*, consider observations at the most recent w observation times, $\{\widehat{\boldsymbol{e}}(t_{i-w+1}^*), \widehat{\boldsymbol{e}}(t_{i-w+2}^*), \ldots, \widehat{\boldsymbol{e}}(t_i^*)\}$. These observations are temporarily regarded as OC data, forming a real-time contrast dataset $\mathcal{D}_{RTC_i}$. By combining $\mathcal{D}_{IC_0}$ with $\mathcal{D}_{RTC_i}$, one can continuously retrain the RF classifier as new data arrive. This real-time updating ensures that the classifier and therefore the charting statistic adapts to evolving process conditions. In this method, the parameters N_0 and w are typically selected based on preliminary experiments. Generally, larger values of w increase the sensitivity of the chart to smaller shifts; but, w should be substantially smaller than N_0 to ensure sufficient contrast between the IC and OC datasets. Hyperparameters for the RF algorithm are tunned as described at the end of Section 2.2.1 for the AC-STD chart.

Among different ways to construct the charting statistic, Deng et al. (2012) proposed using the *out-of-bag* (OOB) classification rate for $\mathcal{D}_{IC_0}$ to define the charting statistic. Let $|\mathcal{D}_{IC_0}|$ denote the number of observations in $\mathcal{D}_{IC_0}$, and $\widehat{P}_{\mathrm{OOB}}(\widehat{\boldsymbol{e}}(\widetilde{t}))$ denotes the OOB correct classification probability for any $\widehat{\boldsymbol{e}}(\widetilde{t}) \in \mathcal{D}_{IC_0}$ that can be obtained by the RF algorithm. Then, the RTC charting statistic at time t_i^* is defined to be the following estimated OOB correct classification rate for observations in $\mathcal{D}_{IC_0}$:

$$P_i = \frac{\sum_{\widehat{\boldsymbol{e}}(\widetilde{t}) \in \mathcal{D}_{IC_0}} \widehat{P}_{\mathrm{OOB}}(\widehat{\boldsymbol{e}}(\widetilde{t}))}{|\mathcal{D}_{IC_0}|}, \qquad \text{for } i \geq 1.$$

Then, the chart gives a signal at time t_i^* when $P_i > h_{RTC}$, where $h_{RTC} > 0$ is a control limit. This chart is called the RTC-STD chart hereafter. Deng et al. (2012) recommended the following bootstrap procedure to determine h_{RTC}:

- Draw a bootstrap sample with replacement from $\mathcal{D}_{IC_1}$.

- Apply the RTC-STD chart with a trial control limit h_{RTC} to this sampled dataset and record the run length (RL).

- Repeat the bootstrap re-sampling process for B times (e.g., $B = 1000$) and obtain B RL values. The average RL value is then used to approximate the actual ARL_0 for the chosen h_{RTC}.

- Search for the h_{RTC} value using a numerical algorithm such as the bisection method until a pre-specified ARL_0 value is achieved.

2.2.3 Control chart based on the support vector machine

The RTC chart described above produces a discrete-valued statistic because the OOB classification probabilities are derived from an ensemble of decision trees (Breiman 2001, He et al. 2018). As a continuous alternative, He et al. (2018) proposed a control chart under the framework of *distance-based SVM* (DSVM). In this approach, the key idea is to use the distances between process observations and the SVM decision boundary as the charting statistic. Specifically, remember that $\mathcal{D}_{RTC_i}$ is the real-time contrast dataset defined in Subsection 2.2.2 at the current time t_i^*. For each observation $\widehat{\boldsymbol{e}}(\widetilde{t}) \in \mathcal{D}_{RTC_i}$, let $d(\widehat{\boldsymbol{e}}(\widetilde{t}))$ denote its distance to the SVM decision boundary. Since the distance can be positive or negative depending on which side of the boundary the given observation lies, He et al. (2018) suggested transforming it to a positive number in $(0, 1]$ using the standard logistic function:

$$g(a) = \frac{1}{1 + \exp(-a)}, \qquad \text{for } a \in (-\infty, \infty).$$

Then, the DSVM charting statistic at time t_i^* is defined to be the average of the transformed distances:

$$M_i = \frac{\sum_{\widehat{\boldsymbol{e}}(\widetilde{t}) \in \mathcal{D}_{RTC_i}} g(d(\widehat{\boldsymbol{e}}(\widetilde{t})))}{|\mathcal{D}_{RTC_i}|}.$$

The chart gives a signal at t_i^* when $M_i > h_{DSVM}$ where $h_{DSVM} > 0$ is a control limit that can be determined by a bootstrap procedure similar to the one described above for the RTC-STD chart. This chart is called DSVM-STD chart hereafter.

In the DSVM algorithm, the kernel parameter and the penalty parameter should be properly selected. He et al. (2018) suggested choosing the kernel parameter to be 2.8 and the penalty parameter to be 1 to achieve a robust charting performance, which is adopted in this paper.

2.2.4 Control chart based on the k-nearest neighbor

Sukchotrat et al. (2009) proposed a control chart using the KNN algorithm described below. For the standardized and decorrelated data $\widehat{\boldsymbol{e}}(t_i^*)$ at the current time t_i^*, the charting statistic is defined to be

$$C_i^2 = \frac{1}{k}\sum_{j=1}^{k} \|\widehat{\boldsymbol{e}}(t_i^*) - NN_j(\widehat{\boldsymbol{e}}(t_i^*))\|^2,$$

where $NN_j(\widehat{\boldsymbol{e}}(t_i^*))$ is the jth element in the set of k nearest neighbors of $\widehat{\boldsymbol{e}}(t_i^*)$ in the IC dataset $\mathcal{D}_{IC}$, and $\|\cdot\|$ is the Euclidean norm. The chart gives a signal at t_i^* if $C_i^2 > h_{KNN}$, where $h_{KNN} > 0$ is a control limit that can be determined by a bootstrap procedure from the IC dataset. This chart is called KNN-STD chart hereafter. In KNN-STD, the parameter k should be chosen properly. The optimal value of k is usually selected using preliminary experiments, balancing sensitivity to process shifts and robustness against false alarms. Generally, smaller k yields a more sensitive chart but could increase false alarms, while larger k reduces sensitivity but enhances stability. Sukchotrat et al. (2009) suggested using the cross-validation procedure for choosing k to minimize detection delays for typical shift sizes encountered in practice.

2.2.5 Control chart based on artificial neural network

Yeganeh et al. (2022b) proposed an ensemble-based artificial neural network (EANN) framework to enhance the sensitivity of a base control chart in detecting shifts in nonparametric profiles. The core idea is to leverage an ensemble of neural networks trained on partitioned regions within the IC domain of the base charting statistic, improving the accuracy of OC predictions. This method operates in the way described below.

First, the IC region of the base charting statistic is defined to be the range $[0, BL]$, where BL denotes the control limit of the chart. For monitoring the j^{th} profile, if the base charting statistic value Q_j satisfies $Q_j > BL$, then the EANN immediately signals an OC condition without further analysis. Otherwise, N ANN learners are used. For the t^{th} learner, ANN_t, the IC region $[0, BL]$ is divided into k_t subregions. The proportion of all j profiles whose charting statistic values falling into each subregion and the current statistic value Q_j serve as the inputs to ANN_t, ensuring comprehensive coverage of the IC profiles. Let T_j be the output obtained by an incorporator that aggregates the outputs of all N ANN learners. Then, T_j is compared with a predetermined threshold value ξ to determine if the current profile is OC or not. Specifically, if $T_j > \xi$, then a signal is triggered and the j^{th} profile is declared to be OC. Otherwise, the profile is IC.

To choose some important parameters of the EANN method (e.g., BL, ξ, N, and k_t), the following guidelines are recommended:

- Initially set the base control limit BL and the threshold value ξ to provisional values. Incrementally adjust these parameters by using constant steps and repeatedly evaluating the performance of the EANN method in terms of the IC average run length (ARL_0) values.

- The ensemble size N is initially chosen to be a small number (e.g., 3), and increased incrementally. The performance of the EANN method is properly evaluated at each step for detecting a target shift. This process stops once further increases would yield negligible improvements (e.g., less than 2% in ARL_1), ensuring efficient complexity.

- For choosing k_t, the formula $k_t = 2t + 1$, for $t = 1, 2, \ldots, N$, is recommended, providing a balance between complexity and prediction accuracy.

The EANN chart that is applied to the spatio-temporally standardized and decorrelated data $\{\widehat{\boldsymbol{e}}(t_i^*), i \geq 1\}$ is called EANN-STD chart hereafter.

3 Simulation Studies

In this section, we evaluate the numerical performance of the five modified machine learning-based control charts, AC-STD, RTC-STD, DSVM-STD, KNN-STD, and EANN-STD that are applied to the spatio-temporally decorrelated data $\{\widehat{\boldsymbol{e}}(t_i^*), i \geq 1\}$, and their original versions, denoted as AC, RTC, DSVM, KNN, and EANN that are applied to the original observed data $\{y(t_i^*, s_{ij}^*), j = 1, \ldots, m_i^*, i \geq 1\}$. Our objective is to study how data preprocessing by spatio-temporal data standardization and decorrelation can affect the performance of the five machine learning-based control charts under various spatio-temporal correlation structures.

3.1 Simulation setup

We assume that the true IC mean function in Model (1) is given by:

$$\mu(t, s) = \cos(2\pi t) + \exp\left\{-\left[(s_x - 0.5)^2 + (s_y - 0.5)^2\right]\right\} + 1,$$

where t is in the time interval $[0, 1]$ and the location $s = (s_x, s_y)'$ is in the spatial domain $\Omega = [0, 1] \times [0, 1]$. Figure 1 shows this mean function at $t = 0, 0.2, 0.4, 0.6, 0.8$, and 1. The observation times considered are $\{t_i = i/n, i = 1, \ldots, n\}$, and the spatial locations $\{s_j, j = 1, \ldots, m\}$ remain fixed over time and are equally spaced in Ω.

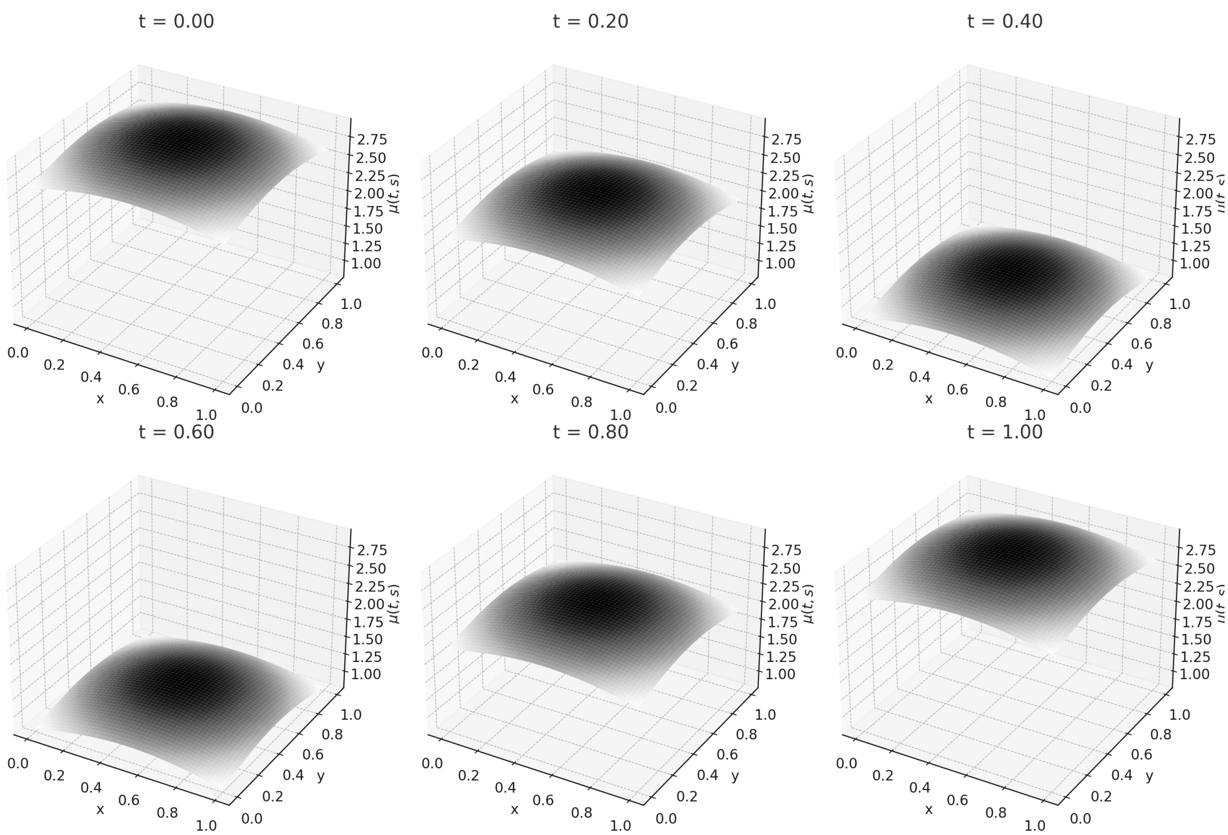

Figure 1: IC mean function $\mu(t, s)$ at selected time points.

We consider the following five cases of spatio-temporal correlation in model (1):

- **Case I:** The errors $\{\epsilon(t_i, s_{ij})\}$ are independent and identically distributed (i.i.d.) with the

$N(0,1)$ common distribution.

- **Cases II and III:** Let $\boldsymbol{\epsilon}(t_i) = (\epsilon(t_i, s_1), \ldots, \epsilon(t_i, s_m))^T$, for each i. They are generated from the following vector AR(1)model:

$$\boldsymbol{\epsilon}(t_i) = \rho_t \boldsymbol{\epsilon}(t_{i-1}) + \sqrt{1-\rho_t^2}\,\boldsymbol{\eta}(t_i),$$

where $\boldsymbol{\eta}(t_i) = (\eta(t_i, s_1), \ldots, \eta(t_i, s_m))^T$ are independent over time t_i and each is a Gaussian spatial process with covariance:

$$\mathrm{Cov}(\eta(t_i, s_j), \eta(t_i, s_l)) = \exp\left(-\frac{d_E(s_j, s_l)}{\rho_s}\right).$$

The resulting spatio-temporal covariance structure of Model (1) is:

$$V(t_i, t_k; s_j, s_l) = \rho_t^{|k-i|} \exp\left(-\frac{d_E(s_j, s_l)}{\rho_s}\right),$$

where $\rho_t > 0$ and $\rho_s > 0$ are the temporal and spatial correlation parameters with larger values implying stronger correlations. In Case II we choose $(\rho_t, \rho_s) = (0.25, 0.1)$, and in Case III we choose $(\rho_t, \rho_s) = (0.5, 0.2)$, representing weak and strong correlation scenarios.

- **Cases IV and V:** The errors $\{\epsilon(t_i, s_{ij})\}$ follow a non-separable spatio-temporal covariance model:

$$\mathrm{Cov}(\eta(t_i, s_j), \eta(t_k, s_l)) = C\big(d_E(s_j, s_l), |t_i - t_k|\big),$$

where

$$C(h, u) = \frac{1}{(a|u|+1)^{0.5}} \exp\left(-\frac{ch}{(a|u|+1)^{0.5}}\right).$$

In Case IV, the parameters (a, c) are chosen to be $(1, 1)$, representing a weak correlation. In Case V, we choose $(a, c) = (0.25, 0.25)$, representing a stronger correlation.

In all simulations, the nominal ARL_0 is fixed at 50 for all methods. Unless otherwise specified, λ in the AC chart is chosen to be 0.2, as suggested by He et al. (2010), the window parameter w in the RTC and DSVM charts is chosen to be 10 and the parameter N_0 in RTC is chosen to be 30, as suggested in Deng et al. (2012) and He et al. (2018), and the parameter k in the KNN chart is chosen to be 30, as suggested in Sukchotrat et al. (2009). The parameter N in EANN is determined as discussed in Yeganeh et al. (2022b), and the control limits BL and ξ in that procedure are chosen as discussed in Section 2.2.5 to reach $ARL_0 = 50$. For all other methods, block bootstrap procedures with boostrap sample size of $B = 1,000$ are used to determine their control limits. Any additional parameters not explicitly discussed here are set according to the recommendations provided in Sections 2.2.1-2.2.5. In Model (1), the number of spatial locations at each time is chosen to be $m = 25$ or 49, and the number of time points is fixed at $n = 200$.

3.2 IC performance

For each chart, its actual ARL_0 values is computed as follows. First, an IC dataset is generated from the IC model (1) and the model is estimated. Second, the chart is applied to an IC spatio-temporal data stream and the run length (RL) value is recorded. This simulation of the online spatio-temporal process monitoring procedure is repeated for 1,000 times to obtain 1,000 IC RL values, which are then averaged to obtain an estimate of the conditional ARL_0 value given the IC dataset. Third, the first two steps are repeated for 100 times to obtain 100 conditional ARL_0 values. The average of these 100 conditional ARL_0 values is then used as the final estimate of the actual ARL_0 value of the chart. In various cases when $m = 25$ or 49, the computed actual ARL_0 values of different charts and their standard errors are presented in Table 1.

From Table 1, the following conclusions can be made. First, in Case I when there is no spatio-temporal correlation, the original machine learning-based charts AC, RTC, DSVM, KNN, and EANN all have reliable IC performance since their actual ARL_0 values are reasonably close to the nominal value of 50. Their IC performance in Case II when there is a weak spatio-temporal correlation is still reasonably good. But, in Cases III-V when the spatio-temporal correlation is quite substantial, their IC performance is not reliable since their ARL_0 values are quite far away from the nominal value of 50, especially in Cases III and V. Second, the modified versions AC-STD, RTC-STD, DSVM-STD, KNN-STD, and EANN-STD have reliable IC performance in all cases considered. Therefore, the proposed data pre-processing can indeed improve the IC performance of the related machine learning-based charts. Third, by comparing the results in Cases II and III, it can be seen that the IC performance of all the charts is better in Case II when the correlation is weaker. A similar conclusion can be made for the results in Cases IV and V.

Table 1: Actual ARL_0 values and their standard errors (in parentheses) in different cases when $m = 25$ or 49, and the nominal ARL_0 value of chart is fixed at 50.

m	Methods	Case I	Case II	Case III	Case IV	Case V
25	AC	47.21(3.49)	44.33(4.25)	29.01(4.30)	42.17(3.53)	30.15(3.93)
	AC-STD	47.98(2.27)	47.62(3.41)	47.51(3.61)	49.27(2.72)	48.17(2.99)
	RTC	51.48(2.74)	43.17(4.13)	35.31(4.62)	43.21(2.52)	31.72(2.92)
	RTC-STD	49.21(2.42)	48.91(3.01)	47.67(3.68)	51.13(2.23)	47.83(2.58)
	DSVM	47.09(2.98)	45.19(4.42)	31.69(4.45)	45.42(3.21)	29.83(3.44)
	DSVM-STD	49.28(2.02)	48.62(2.65)	52.09(3.14)	49.20(2.96)	48.19(3.01)
	KNN	52.32(3.88)	43.12(4.22)	34.57(4.41)	44.54(3.26)	32.24(3.76)
	KNN-STD	48.75(1.73)	48.38(2.14)	51.92(2.65)	51.05(2.58)	52.24(2.67)
	EANN	48.19(2.91)	43.74(3.74)	37.13(4.41)	42.91(3.52)	38.31(4.37)
	EANN-STD	48.62(1.52)	48.01(2.59)	47.84(3.84)	48.11(3.08)	47.65(3.45)
49	AC	47.41(3.72)	45.32(4.45)	26.64(4.92)	44.12(3.83)	32.15(4.23)
	AC-STD	49.89(2.75)	48.53(3.66)	48.15(3.91)	48.55(2.88)	47.98(3.04)
	RTC	51.23(3.74)	46.92(4.43)	31.53(4.64)	42.25(3.82)	35.41(4.52)
	RTC-STD	49.76(2.52)	49.64(3.23)	48.25(3.71)	49.65(2.44)	47.68(2.63)
	DSVM	52.63(2.99)	44.01(4.57)	32.64(4.71)	45.26(4.54)	37.32(4.64)
	DSVM-STD	49.51(2.23)	48.42(2.98)	48.15(3.31)	48.98(3.07)	52.27(3.13)
	KNN	54.24(3.61)	45.53(4.22)	29.58(4.52)	47.12(3.56)	29.57(4.56)
	KNN-STD	49.31(2.75)	51.86(2.80)	47.72(2.92)	48.99(2.78)	47.69(3.01)
	EANN	48.79(2.09)	43.94(2.86)	38.18(3.95)	43.11(3.67)	38.98(3.14)
	EANN-STD	48.94(1.89)	48.47(2.81)	48.06(3.89)	48.49(3.14)	47.88(3.67)

3.3 OC performance

In this part, we examine the OC performance of the related charts in various cases considered before. In each case, a shift is introduced at the start of the online process monitoring and its size keeps the same across both time and space. We consider OC scenarios when the shift size changes among 0.2, 0.4, 0.6, 0.8, and 1.0. Unless otherwise stated, all other simulation settings are the same as before. To ensure a fair comparison, the control limits of all charts have been properly tuned so that their actual ARL_0 values equal the nominal value of 50.

Figures 2 and 3 present the computed actual ARL_1 values when $m = 25$ and 49, respectively, under different scenarios considered. From the figures, we can make the following conclusions. First, in Case I when there is no spatio-temporal data correlation, the five original machine learning-based charts AC, RTC, DSVM, KNN, and EANN perform slightly better than their modified counterparts AC-STD, RTC-STD, DSVM-STD, KNN-STD, and EANN-STD. One explanation of this phenomenon was given in You and Qiu (2019) that the related data decorrelation procedure used in the modified charts would partially mask the shift in the observed data. Second, in Cases II-V when spatio-temporal correlation is present, the modified charts are unanimous better than their original counterparts, implying that the data standardization and decorrelation procedure used in the former charts can effectively enhance their sensitivity to the shift under various data correlation conditions.

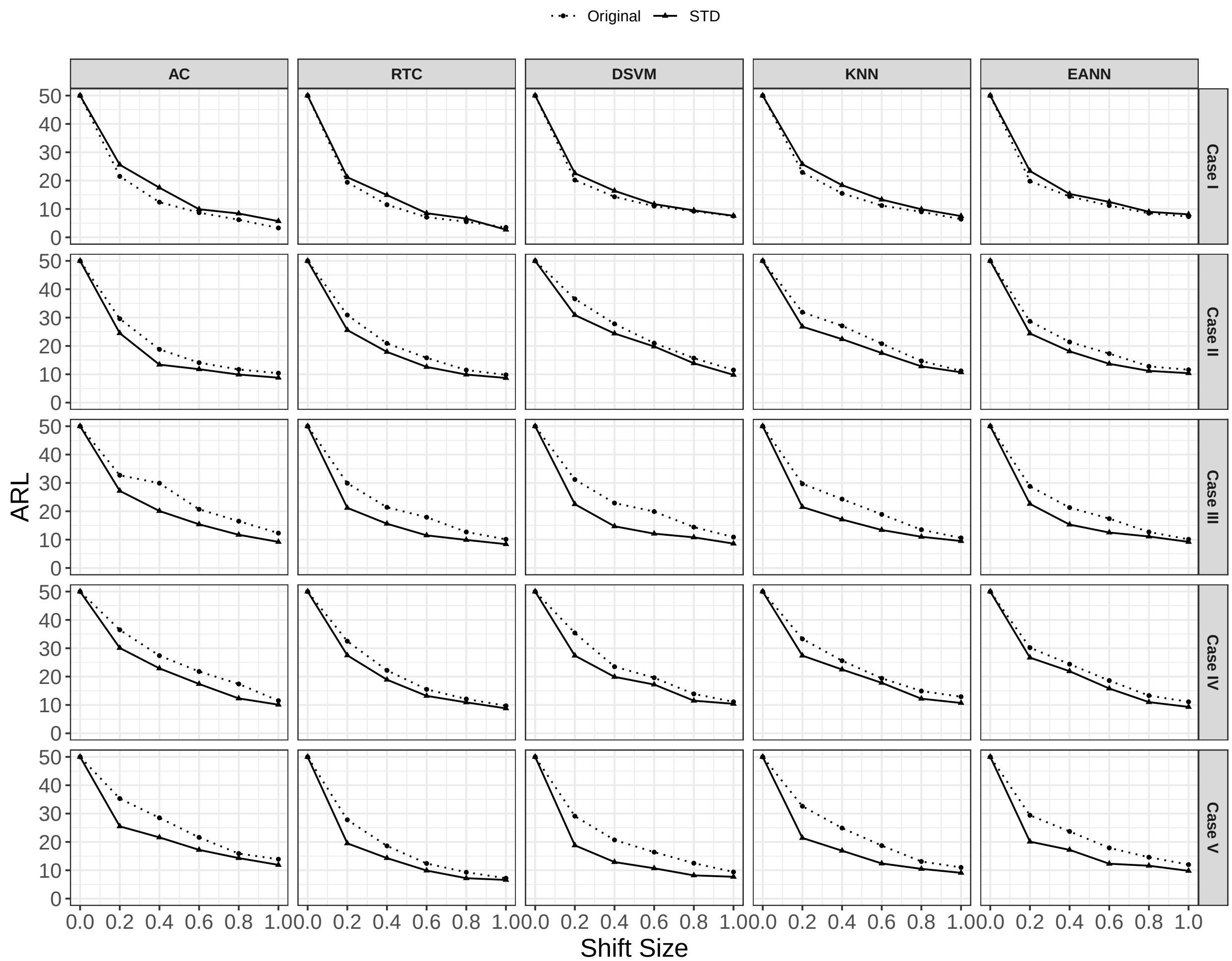

Figure 2: Actual ARL_1 values of the machine learning-based charts AC, RTC, DSVM, KNN, and EANN (dotted lines) and their modified counterparts (solid lines) in different cases when $m = 25$, the nominal ARL_0 is 50, and the shift size changes among 0.2, 0.4, 0.6, 0.8, and 1.0.

4 An Application

The COVID-19 pandemic highlighted the critical importance of effective surveillance for infectious diseases. Monitoring the influenza-like illness (ILI) is not only valuable for early detection of

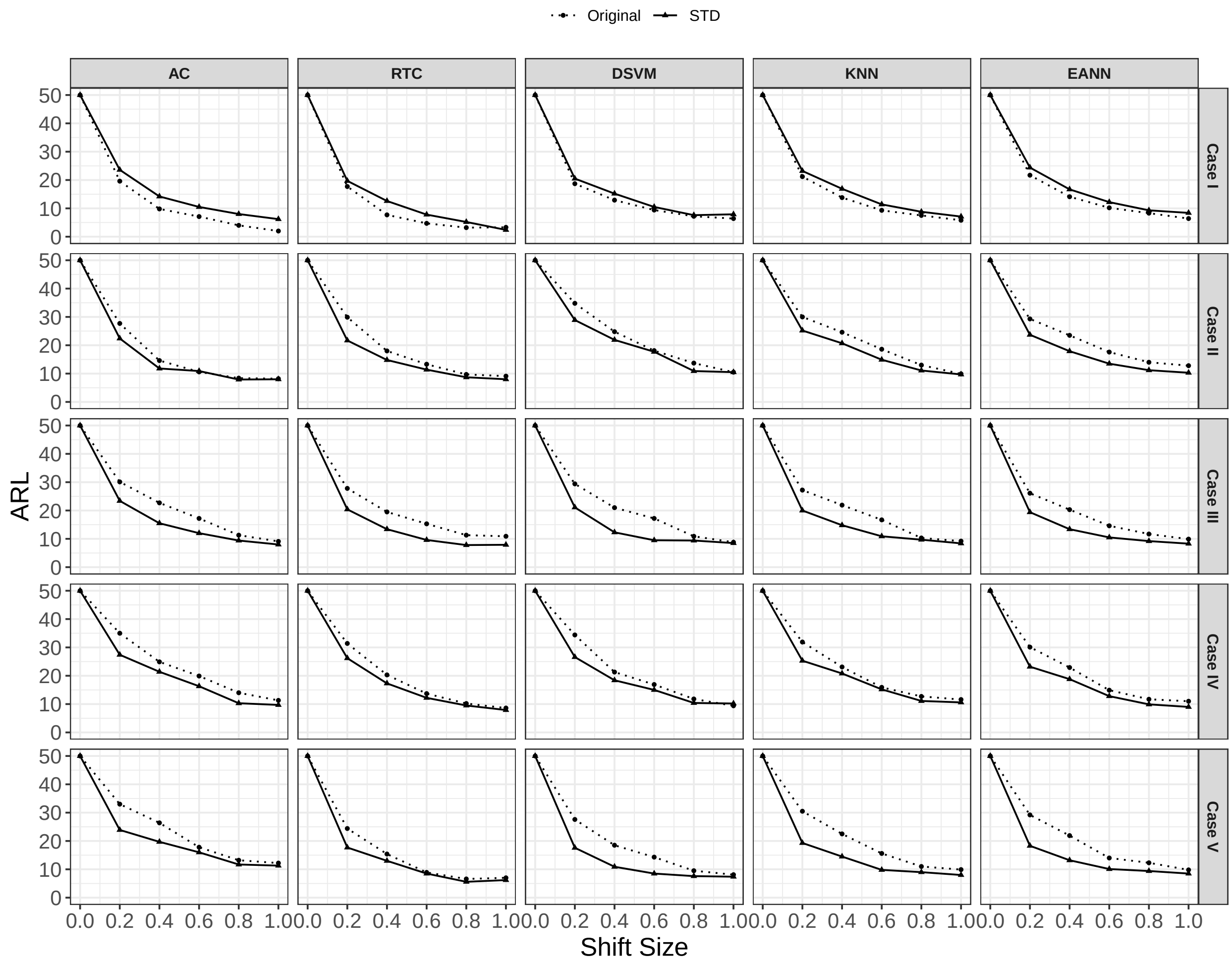

Figure 3: Actual ARL_1 values of the machine learning-based charts AC, RTC, DSVM, KNN, and EANN (dotted lines) and their modified counterparts (solid lines) in different cases when $m = 49$, the nominal ARL_0 is 50, and the shift size changes among 0.2, 0.4, 0.6, 0.8, and 1.0.

outbreaks, but also for evaluating vaccine effectiveness and understanding disease transmission dynamics. Insights from analysis of the ILI data can inform policy decisions, strengthen preparedness, and guide interventions to mitigate the impact of such diseases.

In this section, we apply our modified machine learning-based control charts to an ILI data that are from the Outpatient Influenza-Like Illness Surveillance Network (ILINet) maintained by the CDC (https://www.cdc.gov/fluview/surveillance/usmap.html). ILINet aggregates weekly reports from healthcare providers across the U.S. about the total patient visits and those meeting the ILI definition. To contextualize the data, we note that the CDC officially reported the emergence of COVID-19 in the U.S. in January 2020 (CDC 2020). Figure 4 shows the observed ILI incidence rates in Week 52 of 2018 (left panel) and Week 52 of 2019 (right panel), approximately one week before the CDC's COVID-19 announcement. A notable increase in ILI incidence rates can be observed from 2018 to 2019, suggesting that a significant shift in disease patterns had already begun before the end of 2019.

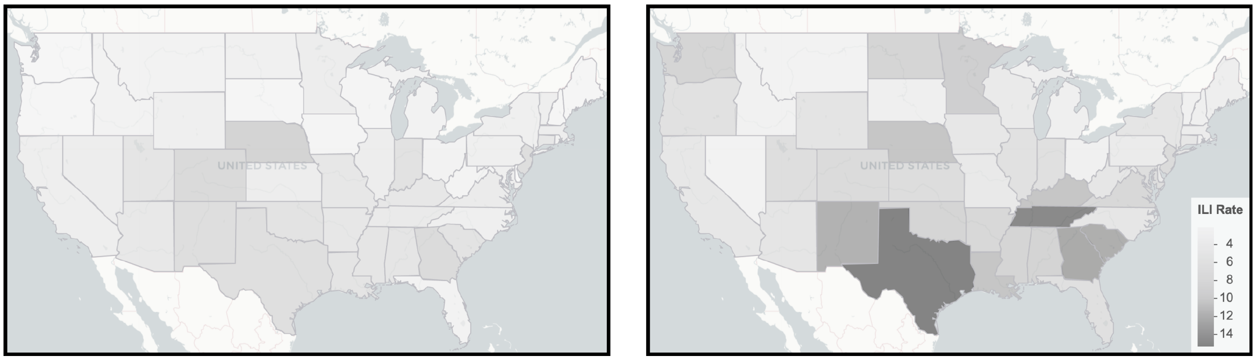

Figure 4: Observed ILI incidence rates in the U.S. in Week 52 of 2018 (left) and Week 52 of 2019 (right). Darker colors indicate larger values.

In our analysis, each flu season spans from Week 41 of one calendar year to Week 40 of the following year. We focus on incidence rates of 48 contiguous U.S. states, excluding Alaska and Hawaii. The observed data from Week 41 of 2018 to Week 40 of 2019 are used as the IC data, and the ones from Week 41 of 2019 to Week 40 of 2020 are used for online monitoring. The distance between two states is defined to be the Euclidean distance between their geometric centroids.

For the five charts AC-STD, RTC-STD, DSVM-STD, KNN-STD, and EANN-STD, their control limits are determined as described in Section 3 and their nominal ARL_0 values are chosen to be 50. The resulting charts are shown in Figure 5. From the figure, it can be seen that the charts AC-STD, RTC-STD, DSVM-STD, KNN-STD, and EANN-STD give their first signals at early January 2020, mid-December 2019, early May 2020, late February 2020, and mid-December 2019, respectively. These detected shifts generally align with the period during which the CDC reported the emergence and spread of COVID-19 within the U.S. Although the exact signal timing differs by the charting method, reflecting variations in sensitivity and the underlying assumptions of different methods, the signals from AC-STD, RTC-STD and EANN-STD, in particular, closely precede or coincide with the early stage of the COVID-19 outbreak.

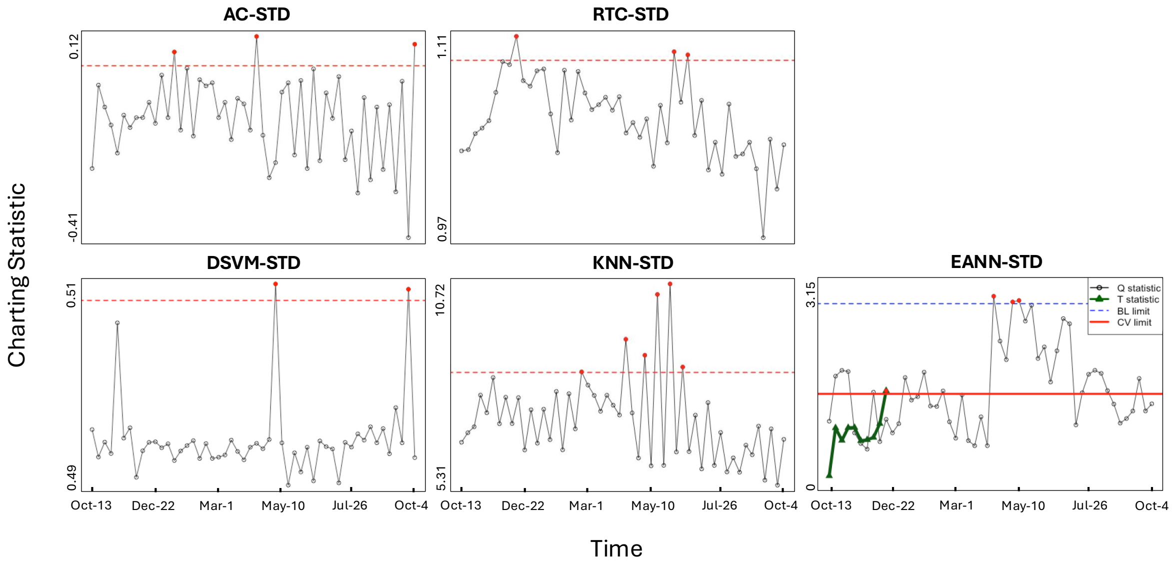

Figure 5: Control charts AC-STD, RTC-STD, DSVM-STD, KNN-STD, and EANN-STD when their nominal ARL_0 values are chosen to be 50. The dashed horizontal lines in all plots indicate the control limits, and the solid horizontal line in the last plot denotes the threshold value of the incorporator in the EANN procedure.

5 Concluding Remarks

Machine learning-based control charts are designed primarily for cases with independent and identically distributed IC process observations. When these methods are applied to spatio-temporal data with complex data structure, our results show that such charts could become unreliable and ineffective. In this paper, we suggest pre-processing the observed spatio-temporal data properly, before the machine learning-based control charts are used for monitoring spatial data streams. Numerical studies presented in the previous sections show that such a modification can substantially improve the performance of the related control charts.

Nonetheless, certain limitations remain with the modified machine learning-based control charts. For instance, the computational burden becomes substantial when the number of spatial locations is large (e.g., $m > 100$), due to the intensive computation when estimating the covariance function and implementing the spatio-temporal decorrelation procedure. To partially address this issue, Qiu and Xie (2022) assumed that the serial correlation in the observed data was shortranged in the sense that the serial correlation could be ignored when two observation times were at least $b_{\max}$ apart, where $b_{\max}$ denoted the range of serial correlation. In practice, this assumption is often reasonable, and the computation involved in the data standardization and decorrelation procedure described in Section 2.1 could be substantially reduced under the short-range serial correlation assumption. Another important issue is related to post-signal diagnoses. Once a signal is triggered by a chart when monitoring a spatio-temporal process, post-signal diagnostic tools are needed to identify anomalous regions and the onset time of the detected shift. Additionally, an autonomous machine learning control system could greatly improve the adaptiveness of SPC by using an appropriate decorrelation procedure and a machine learning control chart, catering to different spatio-temporal distributions and operating characteristic requirements. Future research is needed to address all these issues.

Acknowledgments:

The authors thank a guess editor and two referees for many constructive comments and suggestions, which improved the quality of the paper greatly.

Reference

Aggarwal, C.C. (2018), *Neural Networks and Deep Learning*, New York: Springer.

Apley, D.W., and Tsung, F. (2002), "The autoregressive T^2 chart for monitoring univariate autocorrelated processes," *Journal of Quality Technology*, **34**, 80–96.

Breiman, L. (2001), "Random forests," *Machine Learning*, **45**, 5–32.

Capizzi, G. (2015) "Recent advances in process monitoring: nonparametric and variable-selection methods for phase I and phase II," *Quality Engineering*, **27**, 44–67.

Capizzi, G., and Masarotto, G. (2008), "Practical design of generalized likelihood ratio control charts for autocorrelated data," *Technometrics*, **50**, 357–370.

Carvalho, T.P., Soares, F., Vita, R., Francisco, R., Basto, J.P., and Alcalá, S.G.S. (2019), "A systematic literature review of machine learning methods applied to predictive maintenance," *Computers & Industrial Engineering*, **137**, 106024.

Chakraborti, S., and Graham, M.A. (2019), "Nonparametric (distribution-free) control charts: An updated overview and some results," *Quality Engineering*, **31**, 523–544.

Chakraborti, S., Qiu, P., and Mukherjee, A. (2015), "Editorial to the special issue: Nonparametric statistical process control charts," *Quality and Reliability Engineering International*, **31**, 1–2.

Centers for Disease Control and Prevention (CDC) "COVID-19 Timeline," *David J. Sencer CDC Museum*. https://www.cdc.gov/museum/timeline/covid19.html.

Deng, H., Runger, G., and Tuv, E. (2012), "System monitoring with real-time contrasts," *Journal of Quality Technology*, **44**, 9–27.

Epanechnikov, V.A. (1969) "Non-parametric estimation of a multivariate probability density," *Theory of Probability and its Applications*, **14**, 153–158.

Göb, R. (2006), "Data mining and statistical control - a review and some links," in *Frontiers in Statistical Quality Control* (edited by H.J. Lenz and P.T. Wilrich), **8**, 285–308, Heidelberg: Physica-Verlag.

Hastie, T., Tibshirani, R., and Friedman, J. (2001), *The Elements of Statistical Learning - Data Mining, Inference, and Prediction*. Berlin: Springer-Verlag.

Hawkins, D.M. (1987), "Self-starting cusums for location and scale," *The Statistician*, **36**, 299–315.

He, S., Jiang, W., and Deng, H. (2018), "A distance-based control chart for monitoring multivariate processes using support vector machines," *Annals of Operations Research*, **263**, 191–207.

Hu, J., and Runger, G. (2010), "Time-based detection of changes to multivariate patterns," *Annals of Operations Research*, **174**, 67–81.

Jin, R. and Liu, K. (2013), "Multimode variation modeling and process monitoring for serial-parallel multistage manufacturing processes," *IIE Transactions*, **45**, 617–629.

Li, W., Tsung, F., Song, Z., Zhang, K., and Xiang, D. (2021a), "Multi-sensor based landslide monitoring via transfer learning," *Journal of Quality Technology*, **53**, 474–487.

Li, W., Zhang, C., Tsung, F., and Mei, Y. (2021b), "Nonparametric monitoring of multivariate data via KNN learning," *International Journal of Production Research*, **59**, 6311–6326.

Megahed, F.M., and Jones-Farmer, L.A. (2015), "Statistical perspectives on 'big data'," In *Frontiers in Statistical Quality Control* (eds Knoth S. and Schmid W.), **11**, 29–47, Cham: Springer.

Qiu, P. (2014), *Introduction to Statistical Process Control*, Boca Raton, FL: CRC Press.

Qiu, P. (2024a), "Machine learning approaches for statistical process Control," *Wiley StatsRef: Statistics Reference Online*, DOI: 10.1002/9781118445112.stat08469.

Qiu, P. (2024b), *Statistical Methods for Dynamic Disease Screening and Spatio-Temporal Disease Surveillance*, Boca Raton, FL: CRC Press.

Qiu, P., and Hawkins, D.M. (2001), "A rank based multivariate CUSUM procedure," *Technometrics*, **43**, 120–132.

Qiu, P., Li, W., and Li, J. (2020), "A new process control chart for monitoring short-range serially correlated data," *Technometrics*, **62**, 71–83.

Qiu, P., and Xie, P. (2022), "Transparent sequential learning for statistical process control of serially correlated data," *Technometrics*, **64**(4), 487–501.

Qiu, P., and Yang, K. (2023), "spatial-temporal process monitoring using exponentially weighted spatial LASSO," *Journal of Quality Technology*, **55**, 163-180.

Sukchotrat, T., Kim, S. B., and Tsung, F. (2009), "One-class classification-based control charts for multivariate process monitoring," *IIE Transactions*, **42**, 107–120.

Sun, R., and Tsung, F. (2003), "A kernel-distance-based multivariate control chart using support vector methods," *International Journal of Production Research*, **41**, 2975–2989.

Yeganeh, A., Shadman, A., and Abbasi, S. A. (2022a), "Enhancing the detection ability of control charts in profile monitoring by adding RBF ensemble model," *Neural Computing and Applications*, **34**, 9733–9757

Yeganeh, A., Abbasi, S. A., Pourpanah, F., Shadman, A., Johannssen, A., and Chukhrova, N. (2022b), "An ensemble neural network framework for improving the detection ability of a base control chart in non-parametric profile monitoring," *Expert Systems with Applications*, **204**, 117572.

Tuv, E., and Runger, G. (2003), "Learning patterns through artificial contrasts with application to process control," *Transactions on Information and Communications Technologies*, **29**, 63–72.

Xie, X., and Qiu, P. (2023a), "Dynamic process monitoring using machine learning control charts," In *Artificial Intelligence for Smart Manufacturing - Methods, Applications, and Challenges* (editted by Kim Phuc Tran), 65–82, Springer Nature.

Xie, X., and Qiu, P. (2023b), "Control chart for dynamic process monitoring with an application to air pollution surveillance," *Annals of Applied Statistics*, **17**, 47–66.

Xie, X., Qian, N., and Qiu, P. (2024), "Online monitoring of air quality using PCA-based sequential learning," *Annals of Applied Statistics*, **18**, 600–625.

Yang, K., and Qiu, P. (2018), "spatialtemporal incidence rate data analysis by nonparametric regression," *Statistics in Medicine*, **37**, 996–1012.

Yang, K., and Qiu, P. (2019), "Nonparametric estimation of the spatial-temporal covariance structure," *Statistics in Medicine*, **38**, 2800-2814.

Yang, K., and Qiu, P. (2020), "Online sequential monitoring of spatial-temporal disease incidence rates," *IISE Transactions*, **52**, 1218–1233.

You, L., and Qiu, P. (2019), "Fast computing for dynamic screening systems when analyzing correlated data," *Journal of Statistical Computation and Simulation*, **89**, 379–394.