Multivariate online regression analysis with heterogeneous streaming data

Lan Luo[*]and Peter X.-K. Song[†]

62M20 Inference from stochastic processes and prediction; filtering

September 21, 2021

Abstract

New data collection and storage technologies have given rise to a new field of streaming data analytics, including real-time statistical methodology for online data analyses. Most existing online learning methods are based on homogeneity assumptions that require the samples in a sequence to be independent and identically distributed. However, inter data batch correlation and dynamically evolving, batch-specific effects are among the key defining features of real-world streaming data such as electronic health records and mobile health data. This article is built under a state-space mixed model framework in which the observed data stream is driven by a latent state process that follows a Markov process. In this setting, online maximum likelihood estimation is made challenging by high-dimensional integrals and complex covariance structures. In this article, we develop a real-time Kalman-filter-based regression analysis method that updates both point estimates and their standard errors for fixed population average effects while adjusting for dynamic hidden effects. Both theoretical justification and numerical experiments demonstrate that our proposed online method has similar statistical properties to its offline counterpart and enjoys great computational efficiency. We also apply this method to analyze an electronic health record dataset.

1 Introduction

The advent of distributed cluster-computing paradigms such as Apache Spark (Bifet et al., 2015) has motivated new developments in data analytics for large-scale data processing. Such innovations enable effective analyses of streaming data assembled through, for example, national disease registries, mobile health consortia and infectious disease surveillance programs. One of the defining features of streaming data is that, typically, observations become available sequentially over time at a high velocity. Researchers learn from the sequence

[*]Corresponding author. lan-luo@uiowa.edu, Department of Statistics and Actuarial Science, University of Iowa, Iowa City, IA 52242-1409, U.S.A.

[†]pxsong@umich.edu, Department of Biostatistics, University of Michigan, Ann Arbor, MI 48109-2029, U.S.A.

of data batches to update answers to questions of scientific interest, including assessing disease biomarkers, monitoring product safety and validating drug efficacy and side-effects in phase IV clinical trials, among others.

This article is primarily motivated by a large-scale electronic health record database managed by Scientific Registry of Transplant Recipients (SRTR) since 1984. This database is constantly updated, with new patients added to the transplant wait list in the U.S.A. every ten minutes, resulting in a yearly average of over 25,000 transplant entries since the mid-2000s. Due to the lack of suitable data analytic methods, data collected in real time have been analyzed in a static fashion, leading to latency in the translation of data into clinical knowledge. In addition, conventional static data-analytic approaches are often complicated by limitations in data storage and computational capacity when dealing with high-throughput electronic health records. These analytic and computational challenges call for reliable and efficient, real-time statistical methodologies that promote the timely processing of data to improve clinical decision making, in terms of both learning and inference.

The motivating data in this article consists of a sequence of a kidney transplant datasets updated yearly during the 24-year period from 1994 to 2017. Our analysis uses a total of 158,204 kidney transplant recipients in the U.S.A. with complete personal clinical information. A primary analytic interest is updating learning and inference for important risk factors related to post-transplant serum creatinine, a key biomarker of renal function used to monitor the post-transplant graft condition of a transplanted organ. Primarily, we aim to update the estimated effects of risk factors on a regular basis, namely, when new data becomes available each year. Traditional static data analysis would start with a very large data file composed of both old and new data and then analyze this dataset using suitable statistical methods and software. This traditional approach is not computationally efficient: if we plan to run the same analysis annually over the next 24 years, we need to repeatedly overcome the same logistic barriers such as maintaining and renewing a data use agreement and accessibility to raw data annually throughout the 24-year period. Additionally, repeating the same data cleaning, pre-processing and analysis procedures 24 times with expanded data are clearly laborious, expensive and time-consuming. Thus, it is appealing to develop a smarter solution to conduct this type of data analysis, particularly for streaming data that arrive at a fast rate in large volumes, such as mobile health data.

Most existing methods such as stochastic gradient descent (Robbins and Monro, 1951; Sakrison, 1965; Toulis and Airoldi, 2015) and other online estimation and inference methods (Schifano et al., 2016; Luo and Song, 2020) are built under the homogeneity assumption: all data batches are generated from the same underlying mechanism, and observations arriving over time are independently sampled. Arguably, this widely adopted assumption of independent and identically distributed (i.i.d.) for streaming data is only for mathematical convenience and may be violated in many real-world applications. In practice, different data batches are often heterogeneous and correlated over different sampling points. In the kidney transplant dataset, it is clinically more so where associations between risk predictors and post-transplant serum creatinine evolves dynamically rather than remains constant over the 24-year period. For example, constantly improving organ-matching strategies can alter the effects of risk factors over time. Thus, improvements in medical care or clinical facilities over time may be modelled as temporal confounding variables, while risk factors (e.g., age, sex and body mass index) of primary interest may be assessed as population-average fixed effects.

In the literature, continuous data streams are structured as time series data, for instance, as collected from traffic sensors (Chen et al., 2005), health sensors (Dias and Cunha, 2018), transaction logs (Zhang et al., 2009) and activity logs (Ciuciu et al., 2008). Incorporating dynamic heterogeneity and correlation into the analysis of data streams leads to increased complexity in modelling and statistical inference and is a difficult problem, even in offline settings (L'Heureux et al., 2017; Sadik et al., 2018). It is of great interest to generalize the renewable estimation and incremental inference for i.i.d. samples in Luo and Song (2020) to scenarios with both correlation and dynamic temporal effects.

State-space models, also termed dynamic models, are a very flexible class of models for analyzing time series or longitudinal data when the number of repeated observations is large (Kitagawa, 1987; West and Harrison, 1997; Jørgensen et al., 1999). It is widely used in many applied areas such as economics, engineering and biology. Classical state-space models refer to a class of hierarchical models where an observation process is driven by a latent state process that may incorporate trend, seasonal or time-varying covariate effects.

State-space models appear very flexible in the modelling of certain stochastic behaviors where a latent state process may account for both inter data batch correlation and time-varying heterogeneity in a sequence of observed data batches. This latent process represents temporally or spatially evolving batch-specific effects. In most cases, learning the latent states via, say, filtering or smoothing is a primary goal of statistical analyses. However, our analytic needs in streaming data analysis are based on real time regression, where we focus primarily on updating parameter estimation and inference for population-average fixed effects of key clinical risk factors that are shared across data batches. This type of state-space model, with the addition of population-average effects, is termed a state-space mixed model by Czado and Song (2008).

In applications with large volumes of streaming data, existing offline approaches to fit state-space models require large amounts of computing memory and storage, and repeatedly fitting such models offline may become computationally expensive and even infeasible. For the case where the sample space of the latent state is finite, such as in hidden Markov models, an efficient online expectation–maximization (EM) algorithm (Dempster et al., 1977) based on sufficient statistics has been developed by Cappé (2011). But this algorithm is greatly challenged from a computational perspective by state-space models where the sample space of the latent process is infinite, leading to the introduction of Monte Carlo approximations (Cappé and Moulines, 2009). It is worth noting that most online methods for fitting state-space models are built in a Bayesian paradigm where inference on the latent process, rather than on the fixed effects, is of primary interest. One such example is the streaming variational Bayes method (Broderick et al., 2013) developed for Gaussian process state-space models (Frigola et al., 2014). There is a lack of online regression analysis via state-space models that focus on estimation and inference for population-average fixed effects and adjust for dynamic covariate effects governed by the latent process. Population-average fixed effects are of primary interest in many clinical studies examining the relationship between an outcome and covariates. State-space regression models that contain both deterministic and random predictors have been widely studied in many static settings, for example, in the analysis of longitudinal count data (Jørgensen et al., 1999) and binomial responses (Czado and Song, 2008).

In this article, we develop a new Kalman filter along with an online estimation procedure for linear state-space mixed models. This multivariate online regression analysis (MORA) method enables real-time estimation of both fixed effects and their standard errors. In an online regression paradigm based on linear state-space mixed models, we generalize the renewable estimation and incremental inference methodology proposed in Luo and Song (2020) to estimate fixed effects with both statistical and computational efficiency. Inter data batch heterogeneity is modelled by a batch-specific latent effect that follows a stationary AR(1) process. A crucial step in the proposed methodology is obtaining the conditional distribution of state variables given the data and other model parameters, which we do in a spirit similar to the E step in the EM algorithm. Maximum likelihood estimation is challenging due to the lack of closed form expressions for likelihood functions that typically involve high-dimensional integrals. In our setting, the dimensions of these integrals become infinite when data batches arrive perpetually over time. Thus, approaches permitting approximations are necessary. This will be discussed in detail in this article.

Approximation via the Monte Carlo method is less appealing as far as computational burden is concerned. Instead, we develop an analytic solution through the best linear unbiased predictor in this article, which has been given as an extension of the classical Kalman filter recursion (Harvey, 1981; Song, 2007). The seminal Kalman filter is known as a computationally efficient method that utilizes the first-order Markovian properties of latent states to calculate conditional moments recursively. The resulting recursive data analytics meet the sequential processing needs of MORA where historical, subject-level data are not retrievable and thus, not used. The proposed inference procedure resembles the offline Kalman estimating equation (Song, 2007). The Kalman estimating equation is a generalization of the EM algorithm in which the E step is based on a recursive best linear unbiased predictor and the M step solves an augmented estimating equation. Kalman estimating equations avoid the use of Monte Carlo estimation in the E step, and instead adopt analytic recursive estimation using a Kalman filter. Our proposed MORA method further generalizes the ideals of offline Kalman estimating equations by accommodating heterogeneity in streaming data. Our generalization consists of two new technical elements: the first uses our new online Kalman filter in the E step to recursively update the conditional means of dynamic latent states and the second updates population-average fixed effects using summary statistics from historical data rather than historical, individual-level data, similar to the renewable estimation procedure proposed by Luo and Song (2020). In the setting of a linear state-space mixed model, solving a Kalman estimating equation for the fixed effects has a closed-form solution that is linearly separable by data batches. This separability turns the generalized offline Kalman estimating equation into an online version applicable to streaming data, so the resulting online procedure is scalable to larger volumes of heterogeneous streaming data.

The organization of this article is as follows. Section 2 begins with a brief overview of model assumptions and recursive formulas relevant to the Kalman filter. Section 3 presents key analytic derivations and establishes theoretical guarantees for our proposed MORA method. Section 5 concerns the architecture for the implementation of MORA via the expanded Lambda architecture in Spark (Luo and Song, 2020). Simulation experiments are given in Section 6 to evaluate the performance of MORA. We apply MORA to analyze the kidney transplant dataset, adjusting for some time effects, in Section 7. Finally, we make some concluding remarks in Section 8. A detailed proof of the large-sample property presented in Section 4 is included in the appendix.

2 Model

This section consists of three parts: we introduce state-space mixed models, the Kalman filter estimation procedure, and the mean square error matrix that will be used in online statistical inference.

2.1 Formulation

At a time point $b \geq 2$, a sequence of b data batches, each with a sample size of n_j, for $j = 1, \ldots, b$, arrives sequentially, with a cumulative sample size $N_b = \sum_{j=1}^{b} n_j$. The j-th data batch is denoted by $D_j = \{\boldsymbol{y}_j, \boldsymbol{X}_j, \boldsymbol{Z}_j\}$, where $\boldsymbol{y}_j = (y_{j1}, \ldots, y_{jn_j})^\top \in \mathbb{R}^{n_j \times 1}$, $\boldsymbol{X}_j = (\boldsymbol{x}_{j1}, \ldots, \boldsymbol{x}_{jn_j})^\top \in \mathbb{R}^{n_j \times p}$ and $\boldsymbol{Z}_j = (\boldsymbol{z}_{j1}, \ldots, \boldsymbol{z}_{jn_j})^\top \in \mathbb{R}^{n_j \times q}$, for $j = 1, \ldots, b$ are the vector of responses and the associated covariate matrices for the observed and latent processes, respectively. Here, $n_j = |D_j|$. The cumulative outcome vector and covariate matrices are denoted by $\vec{\boldsymbol{y}}_b = (\boldsymbol{y}_1^\top, \ldots, \boldsymbol{y}_b^\top)^\top \in \mathbb{R}^{N_b \times 1}$, $\vec{\boldsymbol{X}}_b = (\boldsymbol{X}_1^\top, \ldots, \boldsymbol{X}_b^\top)^\top \in \mathbb{R}^{N_b \times p}$ and $\vec{\boldsymbol{Z}}_b = (\boldsymbol{Z}_1^\top, \ldots, \boldsymbol{Z}_b^\top)^\top \in \mathbb{R}^{N_b \times q}$. Let $D_b^\star = \{D_1, \ldots, D_b\}$ be the cumulative data up to batch b, with $N_b = |D_b^\star|$. Note that, in a streaming data setting, the batch size n_b is not supposed to diverge to infinity, but the cumulative sample size N_b is. For simplicity, D_b may be taken as a set of indices. In the framework of state-space mixed models, we postulate a first-order Markov process $\{\boldsymbol{\beta}_b : b \geq 1\}$ to account for cross-batch heterogeneity. We assume that the two series $\{D_b : b \geq 1\}$ and $\{\boldsymbol{\beta}_b : b \geq 1\}$ follow a hierarchical dynamic system defined as follows and as shown in Figure 1:

(A1) given $\boldsymbol{\beta}_b$, the outcome vector $\boldsymbol{y}_b$ is conditionally independent of the other $\boldsymbol{y}_b$s;

(A2) $\{\boldsymbol{\beta}_b : b \geq 1\}$ is a first-order Markov process, with the initial $\boldsymbol{\beta}_1$ assumed to be a fixed, unknown parameter;

(A3) $\boldsymbol{y}_b = \boldsymbol{X}_b\boldsymbol{\alpha} + \boldsymbol{Z}_b\boldsymbol{\beta}_b + \boldsymbol{\epsilon}_b$ with $\boldsymbol{\epsilon}_b \overset{ind}{\sim} \mathcal{N}_{n_b}(\boldsymbol{0}, \phi\boldsymbol{I})$, where $\boldsymbol{\alpha}$ is the vector of population-average fixed effects for the covariates $\boldsymbol{X}_b$, $\boldsymbol{\beta}_b$ is the vector of random effects for the covariates $\boldsymbol{Z}_b$, ϕ is the dispersion parameter and $\boldsymbol{I}$ denotes an identity matrix;

(A4) $\boldsymbol{\beta}_{b+1} = \boldsymbol{B}_b\boldsymbol{\beta}_b + \boldsymbol{\xi}_b$, where $\boldsymbol{B}_b$ is a $q \times q$ transition matrix and $\boldsymbol{\xi}_b \overset{i.i.d.}{\sim} \mathcal{N}_q(\boldsymbol{0}, \delta\boldsymbol{I}_q)$ is Gaussian white noise, with $\boldsymbol{\xi}_b$ and $\boldsymbol{\epsilon}_b$ independent.

In particular, for a stationary AR(1) process, $\boldsymbol{B}_b = \text{diag}(\rho_1, \ldots, \rho_q)$ with $|\rho_s| < 1$ for all $s = 1, \ldots, q$, where ρ_s is the autocorrelation coefficient between the s-th components in $\boldsymbol{\beta}_{b+1}$ and $\boldsymbol{\beta}_b$. For a random walk process, $\boldsymbol{B}_b = \boldsymbol{I}_q$, so $\boldsymbol{\beta}_{b+1} = \boldsymbol{\beta}_b + \boldsymbol{\xi}_b$, where $\delta = 0$ leads to the homogeneous case $\boldsymbol{\beta}_{b+1} = \boldsymbol{\beta}_b$ used extensively in current literature on online regression analysis.

Among many types of state-space models, in this article we focus on a class of linear state-space models with stationary latent processes, that is, a class of models satisfying assumptions (A3) and (A4) with $\boldsymbol{B}_b$ having a bounded spectrum norm, i.e., $\|\boldsymbol{B}_b\|_2 \leq 1$. Obviously, this condition is easily satisfied when $\boldsymbol{B}_b$ is a diagonal matrix of stationary AR(1) processes.

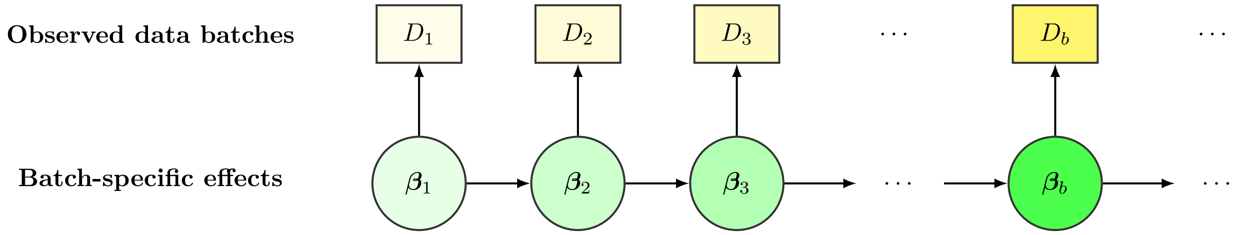

Figure 1: A comb structure for a hierarchical dynamic system. Data batches $\{D_b, b \geq 1\}$ are generated from a state space mixed model with common effect $\boldsymbol{\alpha}$ and batch-specific latent effects $\{\boldsymbol{\beta}_b, b \geq 1\}$ governed by a Markov process.

2.2 Kalman filter

A Kalman filter is used to calculate the conditional mean and variance of the latent state variable or batch-specific effects $\{\boldsymbol{\beta}_b : b \geq 1\}$. Under (A1)–(A4), given the prediction at data batch b, the conditional mean $\boldsymbol{m}_{b-1}$ and covariance $\boldsymbol{C}_{b-1}$, the Kalman filter proceeds recursively as follows.

(i) Compute two types of predictions

$$\boldsymbol{\beta}_b \mid D^\star_{b-1} \sim \mathcal{N}_q(\boldsymbol{B}_{b-1}\boldsymbol{m}_{b-1}, \boldsymbol{H}_b) \text{ and } \boldsymbol{y}_b \mid D^\star_{b-1} \sim \mathcal{N}_{n_b}(\boldsymbol{f}_b, \boldsymbol{Q}_b),$$

where

$$\begin{aligned} \boldsymbol{H}_b &= \operatorname{var}(\boldsymbol{\beta}_b \mid D^\star_{b-1}) = \boldsymbol{B}_{b-1}\boldsymbol{C}_{b-1}\boldsymbol{B}^\top_{b-1} + \delta \boldsymbol{I}_q, \\ \boldsymbol{f}_b &= \mathbb{E}(\boldsymbol{y}_b \mid D^\star_{b-1}) = \boldsymbol{Z}_b\boldsymbol{B}_{b-1}\boldsymbol{m}_{b-1} + \boldsymbol{X}_b\boldsymbol{\alpha} \\ \text{and } \boldsymbol{Q}_b &= \operatorname{var}(\boldsymbol{y}_b \mid D^\star_{b-1}) = \phi \boldsymbol{I}_{n_b} + \boldsymbol{Z}_b\boldsymbol{H}_b\boldsymbol{Z}^\top_b. \end{aligned}$$

(ii) Let $\boldsymbol{K}_b = \boldsymbol{H}^\top_b \boldsymbol{Z}^\top_b \boldsymbol{Q}^{-1}_b$ and update the prediction $\boldsymbol{\beta}_b \mid D^\star_b \sim \mathcal{N}_p(\boldsymbol{m}_b, \boldsymbol{C}_b)$, where

$$\begin{aligned} \boldsymbol{m}_b &= \mathbb{E}(\boldsymbol{\beta}_b \mid D^\star_b) = \boldsymbol{B}_{b-1}\boldsymbol{m}_{b-1} + \boldsymbol{H}^\top_b \boldsymbol{Z}^\top_b \boldsymbol{Q}^{-1}_b(\boldsymbol{y}_b - \boldsymbol{f}_b) \\ \text{and } \boldsymbol{C}_b &= \operatorname{var}(\boldsymbol{\beta}_b \mid D^\star_b) = (\boldsymbol{I}_q - \boldsymbol{K}_b\boldsymbol{Z}_b)\boldsymbol{H}_b. \end{aligned}$$

Consequently, the two inferential quantities needed in our online regression method can be updated by the Kalman filter with the form

$$\mathbb{E}(\boldsymbol{\beta}_b \mid D^\star_b, \tilde{\boldsymbol{\alpha}}_{b-1}, \tilde{\boldsymbol{\zeta}}_{b-1}) = \boldsymbol{m}_b, \text{ and } \operatorname{var}(\boldsymbol{\beta}_b \mid D^\star_b, \tilde{\boldsymbol{\alpha}}_{b-1}, \tilde{\boldsymbol{\zeta}}_{b-1}) = \boldsymbol{C}_b, \tag{1}$$

where $\boldsymbol{\zeta} = (\phi, \rho, \delta)^\top$ is a vector of nuisance parameters.

2.3 Mean square error matrix

Let $\vec{\boldsymbol{m}}_b = (\boldsymbol{m}^\top_1, \boldsymbol{m}^\top_2, \ldots, \boldsymbol{m}^\top_b)^\top$ and $\vec{\boldsymbol{\beta}}_b = (\boldsymbol{\beta}^\top_1, \boldsymbol{\beta}^\top_2, \ldots, \boldsymbol{\beta}^\top_b)^\top$. Assume that

$$\vec{\boldsymbol{\beta}}_b \mid D^\star_b \sim \mathcal{N}_{bp}(\vec{\boldsymbol{m}}_b, \boldsymbol{\Sigma}_b),$$

where $\boldsymbol{\Sigma}_b = \mathbb{E}\left\{(\vec{\boldsymbol{\beta}}_b - \vec{\boldsymbol{m}}_b)(\vec{\boldsymbol{\beta}}_b - \vec{\boldsymbol{m}}_b)^\top\right\}$ is the mean square error matrix of size $bq \times bq$. The block diagonal elements of the matrix $\boldsymbol{\Sigma}_b$ are $\boldsymbol{\Sigma}_b(j,j) = \boldsymbol{C}_j$ $(j = 1, \dots, b)$ and the off-diagonal blocks are $\boldsymbol{\Sigma}_b(j, j+h) = \boldsymbol{\Sigma}_b^\top(j+h, j) = \mathbb{E}\left\{(\boldsymbol{\beta}_j - \boldsymbol{m}_j)(\boldsymbol{\beta}_{j+h} - \boldsymbol{m}_{j+h})^\top\right\}$. Following similar algebra given in Jørgensen and Song (2007),

$$\boldsymbol{\Sigma}_b(j, j+h) = (\boldsymbol{B}_j \boldsymbol{C}_j^\top \boldsymbol{W}_j^{-1})(\boldsymbol{B}_{j+1} \boldsymbol{C}_{j+1}^\top \boldsymbol{W}_{j+1}^{-1}) \cdots (\boldsymbol{B}_{j+h-1} \boldsymbol{C}_{j+h-1}^\top \boldsymbol{W}_{j+h-1}^{-1}) \boldsymbol{C}_{j+h},$$

where $\boldsymbol{W}_j = \mathrm{var}(\boldsymbol{\beta}_{j+1} - \boldsymbol{B}_j \boldsymbol{m}_j) = \delta \boldsymbol{I}_q + \boldsymbol{B}_j \boldsymbol{C}_j \boldsymbol{B}_j^\top$. In particular, $\boldsymbol{\Sigma}_b(j, j+1) = \boldsymbol{B}_j \boldsymbol{C}_j^\top \boldsymbol{W}_j^{-1} \boldsymbol{C}_{j+1}$.

3 Online Regression Analysis

In this section, we first describe the online estimation procedure for regression coefficients of population-average fixed effects, which is our primary interest. Then, we explain how to estimate the nuisance parameters, including the dispersion and correlation parameters.

3.1 Estimation of population-average fixed effects

In this article, we focus on online estimation and inference for the common fixed effect $\boldsymbol{\alpha}$, which will benefit from the accumulation of data batches. For the batch-specific effects $\{\boldsymbol{\beta}_b : b \geq 1\}$, we just report results from a single batch based analysis.

To proceed with maximum likelihood estimation, we first write out the marginal likelihood function for the parameters of interest, $(\boldsymbol{\alpha}, \boldsymbol{\zeta})$:

$$L(\boldsymbol{\alpha}, \boldsymbol{\zeta} \mid D_b^\star) = \int_{\mathbb{R}^{q(b-1)}} P(\boldsymbol{y}_j \mid \boldsymbol{\beta}_j, \boldsymbol{\alpha}, \boldsymbol{\zeta}) P(\boldsymbol{\beta}_j \mid \boldsymbol{\beta}_{j-1}, \boldsymbol{\zeta}) d\boldsymbol{\beta}_2 d\boldsymbol{\beta}_3 \cdots d\boldsymbol{\beta}_b,$$

where the integral is $q(b-1)$-dimensional and both $P(\boldsymbol{y}_j \mid \boldsymbol{\beta}_j, \boldsymbol{\alpha}, \boldsymbol{\zeta})$ and $P(\boldsymbol{\beta}_j \mid \boldsymbol{\beta}_{j-1}, \boldsymbol{\zeta})$ are multivariate normal distributions.

Treating $\boldsymbol{\beta}_b$ as missing data, we obtain the augmented log-likelihood

$$\ell(\boldsymbol{\alpha}, \boldsymbol{\zeta} \mid D_b^\star, \vec{\boldsymbol{\beta}}_b) = \sum_{j=1}^{b} \log P(\boldsymbol{y}_j \mid \boldsymbol{\beta}_j, \boldsymbol{\alpha}, \boldsymbol{\zeta}) + \sum_{j=1}^{b-1} \log P(\boldsymbol{\beta}_{j+1} \mid \boldsymbol{\beta}_j, \boldsymbol{\zeta}).$$

In order to use the EM algorithm (Dempster et al., 1977) to perform maximum likelihood estimation, we maximize the Q-function $Q(\boldsymbol{\alpha}, \boldsymbol{\zeta} \mid \boldsymbol{\alpha}', \boldsymbol{\zeta}') = \mathbb{E}\{\ell(\boldsymbol{\alpha}, \boldsymbol{\zeta} \mid D_b^\star, \vec{\boldsymbol{\beta}}_b)\}$, where the expectation is taken under the conditional distribution $P(\vec{\boldsymbol{\beta}}_b \mid D_b^\star, \boldsymbol{\alpha}', \boldsymbol{\zeta}')$. Here, $\boldsymbol{\alpha}'$ and $\boldsymbol{\zeta}'$ are updated parameter values from the previous iteration. This maximization can be carried out by solving the augmented score equations

$$\boldsymbol{U}_{b,1}^\star(\boldsymbol{\alpha}, \boldsymbol{\zeta}) = \sum_{j=1}^{b} \boldsymbol{X}_j^\top \{\boldsymbol{y}_j - \boldsymbol{X}_j \boldsymbol{\alpha} - \boldsymbol{Z}_j \mathbb{E}(\boldsymbol{\beta}_j \mid D_b^\star, \boldsymbol{\alpha}', \boldsymbol{\zeta}')\} = \boldsymbol{0}$$

$$\text{and } \boldsymbol{U}_{b,2}^\star(\boldsymbol{\alpha}, \boldsymbol{\zeta}) = \sum_{j=1}^{b-1} \{\boldsymbol{\beta}_{j+1} - \boldsymbol{B}_j \mathbb{E}(\boldsymbol{\beta}_j \mid D_b^\star, \boldsymbol{\alpha}', \boldsymbol{\zeta}')\} = \boldsymbol{0}.$$

Instead of using the Monte Carlo techniques to compute the conditional mean $\mathbb{E}(\boldsymbol{\beta}_j \mid D_b^\star, \boldsymbol{\alpha}', \boldsymbol{\zeta}')$, the best linear unbiased predictor (BLUP) (Robinson, 1991) can be used to speed up computation. An obvious advantage of the BLUP is that it can be quickly computed via the recursive Kalman formula. In our proposed online regression analysis method, since historical subject-level data are not available, we adopt the Kalman filter $\mathbb{E}(\boldsymbol{\beta}_b \mid D_b, \tilde{\boldsymbol{\alpha}}_{b-1}, \tilde{\boldsymbol{\zeta}}_{b-1})$, which is recursively updated using only individual-level data in the current data batch D_b rather than the historical cumulative data $D_{b-1}^\star$. Upon the arrival of one data batch, following Titterington (1984) and Cappé and Moulines (2009), we perform a one-step recursive update via the EM algorithm rather than iteratively until convergence.

To further speed up the algorithm, instead of solving $\boldsymbol{U}_{b,2}^\star = \boldsymbol{0}$, we propose using method of moments estimators for $\boldsymbol{\zeta}$. In effect, as the cumulative sample size N_b increases, the choice of the estimator for $\boldsymbol{\zeta}$ becomes less critical.

In summary, the online estimation procedure is conducted as follows.

- Step 1: Choose initial values for the parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\zeta}$, denoted by $\tilde{\boldsymbol{\alpha}}_0$ and $\tilde{\boldsymbol{\zeta}}_0$.
- Step 2: For $b \geq 1$, given $\sqrt{N_{b-1}}$-consistent estimates $\tilde{\phi}_{b-1}$,$\tilde{\rho}_{b-1}$, and $\tilde{\delta}_{b-1}$ from the previous iteration, we update the fixed effects $\tilde{\boldsymbol{\alpha}}_{b-1}$ to $\tilde{\boldsymbol{\alpha}}_b$ by solving the unbiased aggregated Kalman estimating equation (KEE)

$$\tilde{\boldsymbol{U}}_b(\boldsymbol{\alpha}) = \sum_{i=1}^{N_b} \boldsymbol{U}_i(\boldsymbol{\alpha}) = \sum_{j=1}^{b} \boldsymbol{X}_j^\top (\boldsymbol{y}_j - \boldsymbol{X}_j\boldsymbol{\alpha} - \boldsymbol{Z}_j\boldsymbol{m}_j) = \boldsymbol{0}, \tag{2}$$

  where $\boldsymbol{m}_b = \mathbb{E}(\boldsymbol{\beta}_b \mid D_b, \tilde{\boldsymbol{\alpha}}_{b-1}, \tilde{\boldsymbol{\zeta}}_{b-1})$ is the Kalman filter obtained upon the arrival of D_b using the previous updates $\tilde{\boldsymbol{\alpha}}_{b-1}$ and $\tilde{\boldsymbol{\zeta}}_{b-1}$.
- Step 3: Given $\tilde{\boldsymbol{\alpha}}_b$, update the parameter vector $\tilde{\boldsymbol{\zeta}}_{b-1}$ to $\tilde{\boldsymbol{\zeta}}_b$ by the method of moments given in Section 3.2.

In the Gaussian linear model considered in this article, Equation (2) has the closed-form solution

$$\tilde{\boldsymbol{\alpha}}_b = \left(\sum_{j=1}^{b} \boldsymbol{X}_j^\top \boldsymbol{X}_j \right)^{-1} \left\{ \sum_{j=1}^{b} \boldsymbol{X}_j^\top (\boldsymbol{y}_j - \boldsymbol{Z}_j\boldsymbol{m}_j) \right\}, \text{ for } b \geq 1.$$

3.2 Estimation of dispersion and correlation parameters

We invoke the method of moments to estimate both the dispersion and correlation parameters $\boldsymbol{\zeta} = (\phi, \rho, \delta)^\top$. First, note that the equation $\mathrm{var}(\boldsymbol{y}_j - \boldsymbol{X}_j\boldsymbol{\alpha} - \boldsymbol{Z}_j\boldsymbol{m}_j) = \phi\boldsymbol{I}_{n_j} + \boldsymbol{Z}_j\boldsymbol{C}_j\boldsymbol{Z}_j^\top$ leads to the moment estimator for the dispersion parameter ϕ,

$$\hat{\phi}_b^\star = \frac{1}{N_b} \sum_{j=1}^{b} (\boldsymbol{y}_j - \boldsymbol{X}_j\hat{\boldsymbol{\alpha}}_b^\star - \boldsymbol{Z}_j\boldsymbol{m}_j)^\top (\boldsymbol{y}_j - \boldsymbol{X}_j\hat{\boldsymbol{\alpha}}_b^\star - \boldsymbol{Z}_j\boldsymbol{m}_j) - \frac{1}{N_b} \sum_{j=1}^{b} \sum_{i \in D_j} \boldsymbol{P}_j(i, i),$$

where $\boldsymbol{P}_j = \boldsymbol{Z}_j \boldsymbol{C}_j \boldsymbol{Z}_j^\top$, with $\boldsymbol{P}_j(i,i)$ corresponding to the i-th diagonal block of $\boldsymbol{P}_j$. Additionally, note that

$$\begin{aligned}
\delta \boldsymbol{I}_q &= \mathrm{var}(\boldsymbol{\beta}_{j+1} - \boldsymbol{B}_j \boldsymbol{\beta}_j) \\
&= \mathrm{var}\left\{\boldsymbol{\beta}_{j+1} - \boldsymbol{m}_{j+1} - \boldsymbol{B}_j(\boldsymbol{\beta}_j - \boldsymbol{m}_j)\right\} + \mathrm{var}(\boldsymbol{m}_{j+1} - \boldsymbol{B}_j \boldsymbol{m}_j) \\
&= \boldsymbol{C}_{j+1} + \boldsymbol{B}_j \boldsymbol{C}_j \boldsymbol{B}_j^\top - 2\boldsymbol{\Sigma}_b(j+1,j)\boldsymbol{B}_j^\top + \mathrm{var}(\boldsymbol{m}_{j+1} - \boldsymbol{B}_j \boldsymbol{m}_j).
\end{aligned}$$

Similarly, let $\boldsymbol{E}_j = \boldsymbol{C}_{j+1} + \boldsymbol{B}_j \boldsymbol{C}_j \boldsymbol{B}_j^\top - 2\boldsymbol{\Sigma}_b(j+1,j)\boldsymbol{B}_j^\top$, with $\boldsymbol{E}_j(i,i)$ denoting the i-th diagonal block of $\boldsymbol{E}_j$. A moment estimator of δ is

$$\hat{\delta}_b^\star = \frac{1}{bq}\sum_{j=1}^{b}(\boldsymbol{m}_{j+1} - \boldsymbol{B}_j \boldsymbol{m}_j)^\top(\boldsymbol{m}_{j+1} - \boldsymbol{B}_j \boldsymbol{m}_j) + \frac{1}{bq}\sum_{j=1}^{b}\sum_{i=1}^{q}\boldsymbol{E}_j(i,i).$$

These $\sqrt{N_b}$-consistent online estimators of ϕ and δ, for $b \geq 1$, are updated by

$$\tilde{\phi}_b = \frac{N_{b-1}}{N_b}\tilde{\phi}_{b-1} + \frac{n_b}{N_b}\hat{\phi}_b \text{ and } \tilde{\delta}_b = \frac{b-2}{b-1}\tilde{\delta}_{b-1} + \frac{1}{b-1}\hat{\delta}_b,$$

where $\hat{\phi}_b = \frac{1}{n_b}(\boldsymbol{y}_b - \boldsymbol{X}_b\tilde{\boldsymbol{\alpha}}_b - \boldsymbol{Z}_b\boldsymbol{m}_b)^\top(\boldsymbol{y}_b - \boldsymbol{X}_b\tilde{\boldsymbol{\alpha}}_b - \boldsymbol{Z}_b\boldsymbol{m}_b) - \frac{1}{n_b}\sum_{i\in D_b}\boldsymbol{P}_b(i,i)$, $\hat{\delta}_b = \frac{1}{q}\|\boldsymbol{m}_b - \boldsymbol{B}_{b-1}\boldsymbol{m}_{b-1}\|^2 + \frac{1}{q}\sum_{i=1}^{q}\boldsymbol{E}_b(i,i)$.

An estimate of $\boldsymbol{B} = \mathrm{diag}\left(\rho_1, \ldots, \rho_q\right)$ is obtained by the moment conditions

$$\mathrm{cov}(\boldsymbol{m}_b, \boldsymbol{m}_{b-1}) = \boldsymbol{B}\mathrm{var}(\boldsymbol{m}_{b-1}) + \boldsymbol{C}_{b-1}^\top \mathrm{cov}(\boldsymbol{Y}_b - \boldsymbol{f}_b, \boldsymbol{m}_{b-1}) = \boldsymbol{B}\boldsymbol{C}_{b-1}.$$

Therefore, the lag-one autocorrelation of the standardized filter may serve as an estimator of $\boldsymbol{B}$. We carry out online updates using the following estimator via the building blocks $\left(\sum_{j=2}^{b}\boldsymbol{m}_j^\top\boldsymbol{m}_j\right)$ and $\left(\sum_{j=1}^{b}\boldsymbol{m}_j^\top\boldsymbol{m}_{j+1}\right)$, which are clearly separable across the sequence of Kalman filters $\{\boldsymbol{m}_b : b \geq 1\}$:

$$\tilde{\boldsymbol{B}}_b = \left(\sum_{j=2}^{b}\boldsymbol{m}_j^\top\boldsymbol{m}_j\right)^{-1}\left(\sum_{j=1}^{b}\boldsymbol{m}_j^\top\boldsymbol{m}_{j+1}\right), \text{ for } b \geq 2, \text{ with } \tilde{\boldsymbol{B}}_1 = \boldsymbol{0}.$$

4 Theoretical Guarantees

In this section, we establish large-sample properties of the online estimators of the population-average fixed effects $\boldsymbol{\alpha}$ proposed in Section 3. Let $\mathbb{N}_\epsilon(\boldsymbol{\alpha}_0) = \{\boldsymbol{\alpha} : \|\boldsymbol{\alpha} - \boldsymbol{\alpha}_0\|_2 \leq \epsilon\}$ be a neighbourhood around the true value $\boldsymbol{\alpha}_0$. Let $\boldsymbol{U}(\boldsymbol{\alpha})$ be generic notation for the score vector for a single observation, and let the population sensitivity and variability matrices be denoted by $\mathbb{S}(\boldsymbol{\alpha}) = \mathbb{E}_{\boldsymbol{\alpha}}\left\{-\frac{\partial \boldsymbol{U}(\boldsymbol{\alpha})}{\partial \boldsymbol{\alpha}^\top}\right\}$ and $\mathbb{V}(\boldsymbol{\alpha}) = \mathbb{E}_{\boldsymbol{\alpha}}\left\{\boldsymbol{U}(\boldsymbol{\alpha})\boldsymbol{U}^\top(\boldsymbol{\alpha})\right\}$, respectively. We assume the following regularity conditions:

(C1) the true parameter value $\boldsymbol{\alpha}_0$ lies in the interior of parameter space of $\boldsymbol{\alpha}$, denoted by Θ, a compact subset of $\mathbb{R}^p$;

(C2) $\mathbb{E}_{\boldsymbol{\alpha}}\{\boldsymbol{U}(\boldsymbol{\alpha})\} = \boldsymbol{0}$ if and only if $\boldsymbol{\alpha} = \boldsymbol{\alpha}_0$;

(C3) the score vector $\boldsymbol{U}(\boldsymbol{\alpha})$ is twice continuously differentiable with respect to $\boldsymbol{\alpha}$, and the sensitivity matrix $\mathbb{S}(\boldsymbol{\alpha})$ is of full column rank for $\boldsymbol{\alpha} \in \Theta$; and
(C4) the variability matrix $\mathbb{V}(\boldsymbol{\alpha})$ is positive definite for $\boldsymbol{\alpha} \in \mathbb{N}_\epsilon(\boldsymbol{\alpha}_0)$.

Remark The unbiasedness condition (C2) is required for consistency: it implies the $\boldsymbol{\zeta}$-insensitivity of the estimating equation (Song, 2007, Chapter. 12), namely, that $\mathbb{E}\left\{\frac{\partial \boldsymbol{U}(\boldsymbol{\alpha})}{\partial \boldsymbol{\zeta}^\top}\right\} = \boldsymbol{0}$, where $\boldsymbol{\zeta}$ is the nuisance parameter. This property ensures that the efficiency of the nuisance parameter estimator has little influence on the estimation of $\boldsymbol{\alpha}$. Conditions (C3) and (C4) are required to establish both estimation consistency and asymptotic normality. In linear models, the regularity conditions (C2)–(C4) hold automatically.

Theorem 4.1 *Under the regularity conditions (C1)–(C4), for fixed ρ, ϕ and δ, $\tilde{\boldsymbol{\alpha}}_b$ is consistent and asymptotically normal, namely,*

$$\sqrt{N_b}(\tilde{\boldsymbol{\alpha}}_b - \boldsymbol{\alpha}_0) \xrightarrow{d} \mathcal{N}_p\left\{\boldsymbol{0}, \mathbb{J}^{-1}(\boldsymbol{\alpha}_0)\right\} \text{ as } N_b = \sum_{j=1}^{b} n_j \to \infty,$$

where $\mathbb{J}(\boldsymbol{\alpha}_0) = \mathbb{S}^\top(\boldsymbol{\alpha}_0)\mathbb{V}^{-1}(\boldsymbol{\alpha}_0)\mathbb{S}(\boldsymbol{\alpha}_0)$ is the Godambe information matrix of the inference function in Equation (2).

The estimated asymptotic covariance matrix for $\tilde{\boldsymbol{\alpha}}_b$ is $\operatorname{var}(\tilde{\boldsymbol{\alpha}}_b) = \left(\tilde{\boldsymbol{S}}_b^\top \tilde{\boldsymbol{V}}_b^{-1} \tilde{\boldsymbol{S}}_b\right)^{-1}$, where $\tilde{\boldsymbol{S}}_b$ and $\tilde{\boldsymbol{V}}_b$ are calculated as follows. It is easy to see that the $p \times p$ sensitivity matrix $\tilde{\boldsymbol{S}}_b = \sum_{j=1}^{b} \boldsymbol{X}_j^\top \{\boldsymbol{X}_j + \boldsymbol{Z}_j \boldsymbol{L}_j(\tilde{\boldsymbol{\alpha}}_j)\}$, where $\boldsymbol{L}_b(\tilde{\boldsymbol{\alpha}}_b) = \mathbb{E}\left(\partial \boldsymbol{m}_b / \partial \boldsymbol{\alpha}^\top\right) = (\boldsymbol{I}_q - \boldsymbol{K}_b \boldsymbol{Z}_b)\boldsymbol{B}_{b-1}\boldsymbol{L}_{b-1}(\tilde{\boldsymbol{\alpha}}_{b-1}) - \boldsymbol{K}_b \boldsymbol{X}_b$ and $\boldsymbol{L}_0(\tilde{\boldsymbol{\alpha}}_0) = \boldsymbol{0}$.

The variability matrix is updated as

$$\tilde{\boldsymbol{V}}_b \approx \sum_{j=1}^{b} \boldsymbol{X}_j^\top (\tilde{\phi}_j \boldsymbol{I}_{n_j} + \boldsymbol{Z}_j \boldsymbol{C}_j \boldsymbol{Z}_j^\top)\boldsymbol{X}_j - 2\sum_{j=1}^{b-1} \boldsymbol{X}_j^\top \boldsymbol{Z}_j \widetilde{\boldsymbol{\Sigma}}_b(j, j+1)\boldsymbol{Z}_{j+1}^\top \boldsymbol{X}_{j+1},$$

where $\widetilde{\boldsymbol{\Sigma}}_b(j, j+1)$ is the $(j, j+1)$-th off-diagonal block in the estimated mean square error matrix in Section 2.3 with $\tilde{\delta}_j$ and $\tilde{\boldsymbol{B}}_j$, for $j = 1, \ldots, b$. It is worth noting that this is one computational advantage of our proposed online inference method: it only requires the storage of the (j, j)-th diagonal blocks for $j = 1, \ldots, b$ and the $(j, j+1)$-th off-diagonal blocks for $j = 1, \ldots, b-1$. All these blocks are of dimension $q \times q$, so related calculations are scalable with respect to increasing b.

5 Implementation

Apache Spark is a unified data analytics platform for large-scale data processing. Built on a distributed computing paradigm, it offers high performance for both batch and streaming data. Its Lambda architecture is designed to achieve efficient communication and coordination between batch and speed layers to handle streaming data. To implement our proposed online regression analysis method, we expand the speed layer in Spark's existing

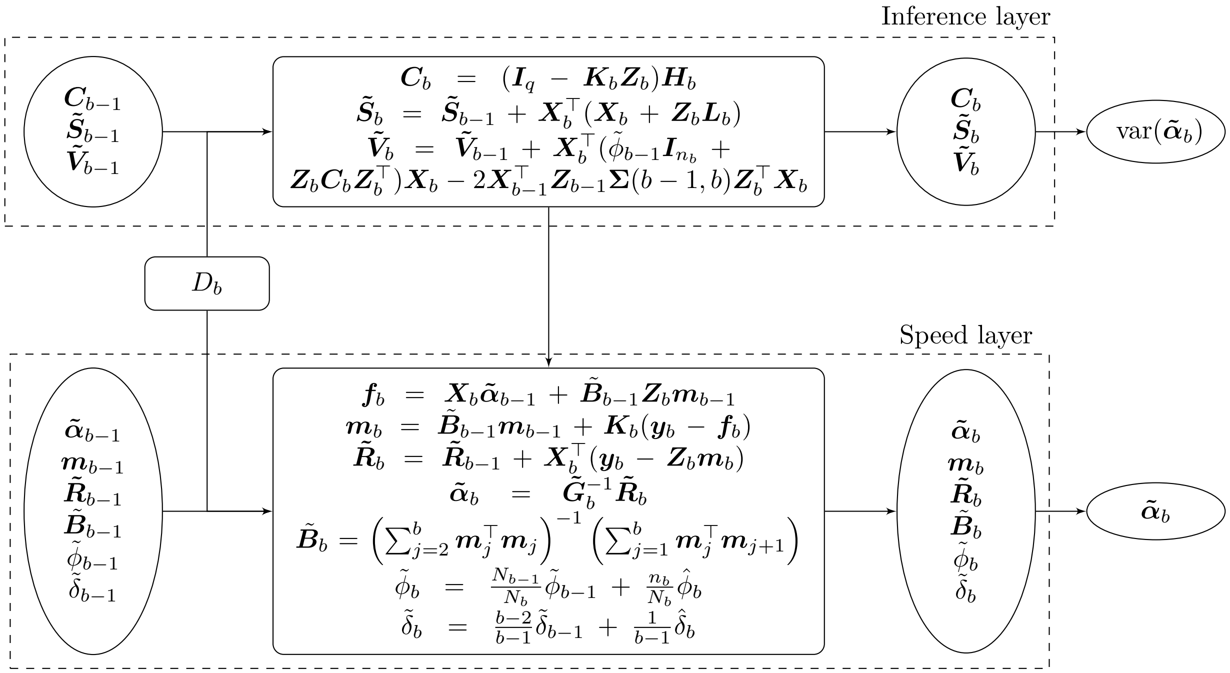
Inference layer
$\boldsymbol{C}_{b-1}$
$\tilde{\boldsymbol{S}}_{b-1}$
$\tilde{\boldsymbol{V}}_{b-1}$
$\boldsymbol{C}_b = (\boldsymbol{I}_q - \boldsymbol{K}_b\boldsymbol{Z}_b)\boldsymbol{H}_b$
$\tilde{\boldsymbol{S}}_b = \tilde{\boldsymbol{S}}_{b-1} + \boldsymbol{X}_b^\top(\boldsymbol{X}_b + \boldsymbol{Z}_b\boldsymbol{L}_b)$
$\tilde{\boldsymbol{V}}_b = \tilde{\boldsymbol{V}}_{b-1} + \boldsymbol{X}_b^\top(\tilde{\phi}_{b-1}\boldsymbol{I}_{n_b} + \boldsymbol{Z}_b\boldsymbol{C}_b\boldsymbol{Z}_b^\top)\boldsymbol{X}_b - 2\boldsymbol{X}_{b-1}^\top\boldsymbol{Z}_{b-1}\boldsymbol{\Sigma}(b-1,b)\boldsymbol{Z}_b^\top\boldsymbol{X}_b$
$\boldsymbol{C}_b$
$\tilde{\boldsymbol{S}}_b$
$\tilde{\boldsymbol{V}}_b$
$\mathrm{var}(\tilde{\boldsymbol{\alpha}}_b)$
D_b
Speed layer
$\tilde{\boldsymbol{\alpha}}_{b-1}$
$\boldsymbol{m}_{b-1}$
$\tilde{\boldsymbol{R}}_{b-1}$
$\tilde{\boldsymbol{B}}_{b-1}$
$\tilde{\phi}_{b-1}$
$\tilde{\delta}_{b-1}$
$\boldsymbol{f}_b = \boldsymbol{X}_b\tilde{\boldsymbol{\alpha}}_{b-1} + \tilde{\boldsymbol{B}}_{b-1}\boldsymbol{Z}_b\boldsymbol{m}_{b-1}$
$\boldsymbol{m}_b = \tilde{\boldsymbol{B}}_{b-1}\boldsymbol{m}_{b-1} + \boldsymbol{K}_b(\boldsymbol{y}_b - \boldsymbol{f}_b)$
$\tilde{\boldsymbol{R}}_b = \tilde{\boldsymbol{R}}_{b-1} + \boldsymbol{X}_b^\top(\boldsymbol{y}_b - \boldsymbol{Z}_b\boldsymbol{m}_b)$
$\tilde{\boldsymbol{\alpha}}_b = \tilde{\boldsymbol{G}}_b^{-1}\tilde{\boldsymbol{R}}_b$
$\tilde{\boldsymbol{B}}_b = \left(\sum_{j=2}^b \boldsymbol{m}_j^\top\boldsymbol{m}_j\right)^{-1}\left(\sum_{j=1}^b \boldsymbol{m}_j^\top\boldsymbol{m}_{j+1}\right)$
$\tilde{\phi}_b = \frac{N_{b-1}}{N_b}\tilde{\phi}_{b-1} + \frac{n_b}{N_b}\hat{\phi}_b$
$\tilde{\delta}_b = \frac{b-2}{b-1}\tilde{\delta}_{b-1} + \frac{1}{b-1}\hat{\delta}_b$
$\tilde{\boldsymbol{\alpha}}_b$
$\boldsymbol{m}_b$
$\tilde{\boldsymbol{R}}_b$
$\tilde{\boldsymbol{B}}_b$
$\tilde{\phi}_b$
$\tilde{\delta}_b$
$\tilde{\boldsymbol{\alpha}}_b$

Figure 2: Diagram of the expanded Lambda architecture in which the online estimators $\tilde{\boldsymbol{\alpha}}_{b-1}$ and $\tilde{\boldsymbol{\zeta}}_{b-1}$ are updated to $\tilde{\boldsymbol{\alpha}}_b$ and $\tilde{\boldsymbol{\zeta}}_b$ at the speed layer and the online information matrices $\tilde{\boldsymbol{S}}_{b-1}$ and $\tilde{\boldsymbol{V}}_{b-1}$ are updated to $\tilde{\boldsymbol{S}}_b$ and $\tilde{\boldsymbol{V}}_b$ at the inference layer.

Lambda architecture to accommodate inferential statistics such as sensitivity and variability matrices together with other needed quantities in the recursive Kalman filter calculation. Consequently, the resulting architecture consists of a speed layer and an inference layer responsible for the iterative calculation detailed in Section 3. As shown in Figure 2, when a new data batch D_b arrives, the inference layer calculates the matrices involved in both the Kalman filter and inferential statistics. These quantities are then sent to the speed layer to update the point estimates of $\boldsymbol{\alpha}$ and $\boldsymbol{\zeta}$. Finally, the outputs from both layers are combined to generate online regression analysis results.

Algorithm 1 gives pseudocode implementing online regression analysis with dynamic heterogeneity in the expanded Lambda architecture.

Algorithm 1: Online regression analysis for heterogeneous streaming data via our expanded Lambda architecture.

```
**Inputs:** sequentially arriving datasets $D_1$,...,$D_b$,... ;
**Outputs:** $\tilde{\boldsymbol{\alpha}}_b$, var($\tilde{\boldsymbol{\alpha}}_b$), $\tilde{\boldsymbol{B}}_b$, $\tilde{\phi}_b$, $\tilde{\delta}_b$, $\boldsymbol{m}_b$ and $\boldsymbol{C}_b$, for $b = 1, 2, \dots$ ;
**Initialize:** set initial values $\tilde{\boldsymbol{\alpha}}_0 = \mathbf{0}_{p\times 1}$, $\tilde{\boldsymbol{R}}_0 = \mathbf{0}_{p\times 1}$, $\tilde{\boldsymbol{G}}_0 = \tilde{\boldsymbol{S}}_0 = \tilde{\boldsymbol{V}}_0 = \mathbf{0}_{p\times p}$,
    $\boldsymbol{L}_0 = \mathbf{0}_{q\times p}$, $\tilde{\boldsymbol{B}}_0 = 10^{-3}\boldsymbol{I}_{q\times q}$ and $\tilde{\phi}_0 = \tilde{\delta}_0 = 10^{-3}$ ;
**for** $b = 1, \dots$ **do**
    Read in the dataset $D_b$;
    At the inference layer, calculate $\boldsymbol{H}_b = \tilde{\boldsymbol{B}}_{b-1}\boldsymbol{C}_{b-1}\tilde{\boldsymbol{B}}_{b-1}^\top + \tilde{\delta}_{b-1}\boldsymbol{I}_q$,
      $\boldsymbol{Q}_b = \tilde{\phi}_{b-1}\boldsymbol{I}_{n_b} + \boldsymbol{Z}_b\boldsymbol{H}_b\boldsymbol{Z}_b^\top$, $\boldsymbol{K}_b = \boldsymbol{H}_b^\top\boldsymbol{Z}_b^\top\boldsymbol{Q}_b^{-1}$, $\boldsymbol{C}_b = (\boldsymbol{I}_q - \boldsymbol{K}_b\boldsymbol{Z}_b)\boldsymbol{H}_b$,
      $\boldsymbol{L}_b = (\boldsymbol{I}_q - \boldsymbol{K}_b\boldsymbol{Z}_b)\boldsymbol{B}_{b-1}\boldsymbol{L}_{b-1} - \boldsymbol{K}_b\boldsymbol{X}_b$, $\tilde{\boldsymbol{S}}_b = \tilde{\boldsymbol{S}}_{b-1} + \boldsymbol{X}_b^\top(\boldsymbol{X}_b + \boldsymbol{Z}_b\boldsymbol{L}_b)$,
      $\boldsymbol{W}_{b-1} = \tilde{\boldsymbol{B}}_{b-1}\boldsymbol{C}_{b-1}\tilde{\boldsymbol{B}}_{b-1}^\top + \tilde{\delta}_{b-1}\boldsymbol{I}_q$, $\boldsymbol{\Sigma}(b-1,b) = \tilde{\boldsymbol{B}}_{b-1}\boldsymbol{C}_b\boldsymbol{W}_{b-1}^{-1}\boldsymbol{C}_{b-1}$,
      $\tilde{\boldsymbol{V}}_b = \tilde{\boldsymbol{V}}_{b-1} + \boldsymbol{X}_b^\top(\tilde{\phi}_{b-1}\boldsymbol{I}_{n_b} + \boldsymbol{Z}_b\boldsymbol{C}_b\boldsymbol{Z}_b^\top)\boldsymbol{X}_b - 2\boldsymbol{X}_{b-1}^\top\boldsymbol{Z}_{b-1}\boldsymbol{\Sigma}(b-1,b)\boldsymbol{Z}_b^\top\boldsymbol{X}_b$,
      $\boldsymbol{P}_b = \boldsymbol{Z}_b\boldsymbol{C}_b\boldsymbol{Z}_b^\top$ and $\boldsymbol{E}_b = \boldsymbol{C}_b + \tilde{\boldsymbol{B}}_{b-1}\boldsymbol{C}_{b-1}\tilde{\boldsymbol{B}}_{b-1}^\top - 2\tilde{\boldsymbol{B}}_{b-1}\boldsymbol{\Sigma}(b-1,b)$;
    At the speed layer, calculate
    $\boldsymbol{f}_b = \boldsymbol{X}_b\tilde{\boldsymbol{\alpha}}_{b-1} + \tilde{\boldsymbol{B}}_{b-1}\boldsymbol{Z}_b\boldsymbol{m}_{b-1}$, $\boldsymbol{m}_b = \tilde{\boldsymbol{B}}_{b-1}\boldsymbol{m}_{b-1} + \boldsymbol{K}_b(\boldsymbol{y}_b - \boldsymbol{f}_b)$,
      $\tilde{\boldsymbol{R}}_b = \tilde{\boldsymbol{R}}_{b-1} + \boldsymbol{X}_b^\top(\boldsymbol{y}_b - \boldsymbol{Z}_b\boldsymbol{m}_b)$, $\tilde{\boldsymbol{G}}_b = \tilde{\boldsymbol{G}}_{b-1} + \boldsymbol{X}_b^\top\boldsymbol{X}_b$, $\tilde{\boldsymbol{\alpha}}_b = \tilde{\boldsymbol{G}}_b^{-1}\tilde{\boldsymbol{R}}_b$,
      $\tilde{\boldsymbol{B}}_b = \left(\sum_{j=2}^{b}\boldsymbol{m}_j^\top\boldsymbol{m}_j\right)^{-1}\left(\sum_{j=1}^{b}\boldsymbol{m}_j^\top\boldsymbol{m}_{j+1}\right)$,
      $\hat{\phi}_b = \frac{1}{n_b}(\boldsymbol{y}_b - \boldsymbol{X}_b\tilde{\boldsymbol{\alpha}}_b - \boldsymbol{Z}_b\boldsymbol{m}_b)^\top(\boldsymbol{y}_b - \boldsymbol{X}_b\tilde{\boldsymbol{\alpha}}_b - \boldsymbol{Z}_b\boldsymbol{m}_b) - \frac{1}{n_b}\sum_{i\in D_b}\boldsymbol{P}_b(i,i)$,
      $\hat{\delta}_b = \frac{1}{q}\|\boldsymbol{m}_b - \tilde{\boldsymbol{B}}_b\boldsymbol{m}_{b-1}\|^2 + \frac{1}{q}\sum_{i=1}^{q}\boldsymbol{E}_b(i,i)$,
    and then update $\tilde{\phi}_b$ and $\tilde{\delta}_b$.
    Save $\{\tilde{\boldsymbol{\alpha}}_b, \boldsymbol{m}_b, \tilde{\boldsymbol{R}}_b, \tilde{\boldsymbol{G}}_b, \tilde{\boldsymbol{B}}_b, \tilde{\phi}_b, \tilde{\delta}_b\}$ and $\{\boldsymbol{C}_b, \tilde{\boldsymbol{S}}_b, \tilde{\boldsymbol{V}}_b\}$ at the speed and inference
      layers, respectively;
    Release the dataset $D_b$ from memory.
**end**
**Return** $\tilde{\boldsymbol{\alpha}}_b$, var($\tilde{\boldsymbol{\alpha}}_b$) $= \tilde{\boldsymbol{S}}_b^\top\tilde{\boldsymbol{V}}_b^{-1}\tilde{\boldsymbol{S}}_b$, $\tilde{\boldsymbol{B}}_b$, $\tilde{\phi}_b$, $\tilde{\delta}_b$, $\boldsymbol{m}_b$ and $\boldsymbol{C}_b$, for $b = 1, 2, \dots$.
```

6 Simulation Studies

This section begins with the setup of our numerical experiments. Then we compare our proposed MORA method with other methods under two scenarios: (i) a fixed total sample size N_B but a varying data batch size n_b and (ii) a fixed data batch size n_b but an increasing number of data batches B.

6.1 Setup

We conduct simulation studies to assess the performance of our proposed MORA method. We compare our method with the naive linear regression model (LM) from the R package `glm` without considering either inter data batch correlation or heterogeneity, and the offline Kalman estimating equation (KEE) estimator obtained by processing the entire data once. The evaluation criteria for parameter estimation and inference for α include (a) average absolute bias (α.ABIAS), (b) average estimated standard error (α.ASE), (c) empirical standard error (α.ESE) and (d) coverage probability (α.CP). Computational efficiency is assessed by (e) computation time (C.Time) and (f) running time (R.Time). C.Time includes time spent on both loading data and running the algorithm while R.Time accounts for only algorithm execution time.

In simulation experiments, we set a terminal point B. Consider the data batch $D_b = \{\boldsymbol{y}_b, \boldsymbol{X}_b\}$ with the outcome $\boldsymbol{y}_b = (y_{b1}, \ldots, y_{bn_b})^\top$, covariates for population-average effects $\boldsymbol{X}_b = (\boldsymbol{x}_{b1}, \ldots, \boldsymbol{x}_{bn_b})^\top$, and batch-specific covariates $\boldsymbol{Z}_b = (\boldsymbol{z}_{b1}, \ldots, \boldsymbol{z}_{bn_b})^\top$. Outcomes $\boldsymbol{y}_b \mid \boldsymbol{X}_b, \boldsymbol{Z}_b$ are independently sampled from a Gaussian distribution with a mean of $\boldsymbol{\mu}_b = (\mu_{b1}, \ldots, \mu_{bn_b})^\top$ and a variance of ϕI such that $\mu_{bi} = \mathbb{E}(y_{bi} \mid \boldsymbol{x}_{bi}, \boldsymbol{z}_{bi}) = \boldsymbol{x}_{bi}^\top \boldsymbol{\alpha} + \boldsymbol{z}_{bi}^\top \boldsymbol{\beta}_b$ and variance $\mathrm{var}(y_{bi} \mid \boldsymbol{x}_{bi}, \boldsymbol{z}_{bi}) = \phi$. We consider a two-dimensional stationary vector AR(1) process to characterize batch-specific heterogeneity with regression coefficients satisfying $\boldsymbol{\beta}_{b+1} = \boldsymbol{B}_b \boldsymbol{\beta}_b + \boldsymbol{\xi}_b$, where $\boldsymbol{B}_b = \mathrm{diag}(\rho_1, \rho_2)$ is the transition matrix with the respective autocorrelation coefficients ρ_1 and ρ_2, and $\boldsymbol{\xi}_b \overset{iid}{\sim} \mathcal{N}_2(0, \delta \boldsymbol{I})$ is noise, for $b = 1, \ldots, B$.

We choose the true regression coefficient parameters by generating $\boldsymbol{\alpha}_0 \sim \mathcal{N}_5(\mathbf{0}, \boldsymbol{I}_5)$, where $\boldsymbol{I}_5$ is the 5×5 identity matrix. We set the initial value for the dynamic coefficients as $\boldsymbol{\beta}_1 = 0$. Covariates are independently sampled from $\boldsymbol{x}_i \overset{i.i.d}{\sim} \mathcal{N}_5(\mathbf{0}, \boldsymbol{V}_5)$ for $i = 1, \ldots, N_b$, where $\boldsymbol{V}_5$ is a 5×5 compound symmetry covariance matrix with a correlation parameter $\rho_x = 0.5$. The variance parameters of the two covariance matrices are set as $\phi = 1$ and $\delta = 1$. As far as the online procedure is concerned, we only consider the correlation between adjacent data batches. Thus, we examine performance under different correlation coefficients $\rho_1 = 0.1$, 0.5, 0.9 while ρ_2 is fixed at 0.5.

6.2 Fixed N_B and varying batch size n_b

We begin with evaluating the effect of data batch size n_b on the performance of the MORA method's parameter estimation and computational efficiency. There are B data batches, each with size n_b. The total sample size is $N_B = |D_B^\star| = 10,000$. These samples are generated in data batches from the linear state-space mixed model specified in Section 6.1. Table 1 reports the evaluation criteria, averaged over 500 replications.

Bias and coverage probability in $\boldsymbol{\alpha}$. Between the offline KEE and MORA methods, as shown in Table 1, estimation bias and coverage probability are very close to each other, and neither changes with varying batch sample size n_b. This confirms the theoretical results given in Theorem 4.1. In other words, statistical inference by the MORA method depends only on the cumulative sample size N_B. However, in the naive LM method, where outcomes are treated as independent, $\boldsymbol{\alpha}$.ABIAS, $\boldsymbol{\alpha}$.ASE, and $\boldsymbol{\alpha}$.ESE are all larger than in either the offline KEE or MORA methods due to the loss of statistical efficiency. The coverage probability in the LM method is still around 95% because `glm` in `R` uses iteratively weighted least squares, where extra variability is accounted for by an empirical weighting matrix. Additionally, considering correlation between only adjacent data batches shows very marginal effects on inference performance on $\boldsymbol{\alpha}$: the coverage probabilities are close to the nominal 95% level under different values of the autocorrelation parameter ρ_1.

Computation time. Computational efficiency is assessed in Table 1 by C.Time and

	$\rho_1 = 0.1,\ \rho_2 = 0.5$								
$B \times n_b$	$5 \times 2,000$			50×200			500×20		
	LM	KEE	MORA	LM	KEE	MORA	LM	KEE	MORA
$\boldsymbol{\alpha}$.ABIAS$\times 10^{-3}$	16.49	10.56	10.96	18.84	10.45	10.55	18.66	10.73	10.76
$\boldsymbol{\alpha}$.ASE$\times 10^{-3}$	20.68	12.92	13.66	23.31	12.97	13.19	23.58	13.58	13.64
$\boldsymbol{\alpha}$.ESE$\times 10^{-3}$	21.14	13.25	13.81	23.51	12.99	13.19	23.38	13.36	13.38
$\boldsymbol{\alpha}$.CP	0.946	0.945	0.947	0.953	0.953	0.953	0.950	0.952	0.951
C.Time (s)	0.04	44.91	16.47	0.08	1.97	0.57	0.45	1.87	0.49
R.Time (s)	0.03	44.90	16.46	0.04	1.93	0.54	0.09	1.46	0.31

	$\rho_1 = 0.5,\ \rho_2 = 0.5$								
$B \times n_b$	$5 \times 2,000$			50×200			500×20		
	LM	KEE	MORA	LM	KEE	MORA	LM	KEE	MORA
$\boldsymbol{\alpha}$.ABIAS$\times 10^{-3}$	16.29	10.55	10.95	19.47	10.45	10.54	19.80	10.73	10.76
$\boldsymbol{\alpha}$.ASE$\times 10^{-3}$	20.50	12.92	13.67	24.24	12.98	13.20	24.66	13.58	13.64
$\boldsymbol{\alpha}$.ESE$\times 10^{-3}$	20.98	13.25	13.81	24.39	12.99	13.19	24.76	13.36	13.38
$\boldsymbol{\alpha}$.CP	0.946	0.946	0.947	0.952	0.953	0.953	0.948	0.952	0.951
C.Time (s)	0.06	43.13	17.12	0.10	2.16	0.57	0.51	2.11	0.55
R.Time (s)	0.04	43.12	17.10	0.05	2.10	0.54	0.04	1.64	0.36

	$\rho_1 = 0.9,\ \rho_2 = 0.5$								
$B \times n_b$	$5 \times 2,000$			50×200			500×20		
	LM	KEE	MORA	LM	KEE	MORA	LM	KEE	MORA
$\boldsymbol{\alpha}$.ABIAS$\times 10^{-3}$	16.55	10.55	10.94	24.03	10.45	10.55	28.20	10.74	10.77
$\boldsymbol{\alpha}$.ASE$\times 10^{-3}$	20.86	12.92	13.67	30.41	12.97	13.20	34.86	13.58	13.66
$\boldsymbol{\alpha}$.ESE$\times 10^{-3}$	21.38	13.20	13.81	30.61	12.99	13.19	35.31	13.36	13.39
$\boldsymbol{\alpha}$.CP	0.948	0.946	0.946	0.951	0.953	0.953	0.944	0.952	0.952
C.Time (s)	0.06	43.85	16.39	0.10	2.49	0.64	0.45	1.87	0.50
R.Time (s)	0.04	43.83	16.10	0.04	2.44	0.60	0.04	1.46	0.32

Table 1: Simulation results under the linear state-space mixed model, summarized over 500 replications with $N_B = 10,000$, $p = 5$ and varying batch size n_b.

R.Time, which refer to total algorithm execution times, respectively. As expected, MORA is more efficient than offline KEE and provides similar statistical performance. Additionally, while maintaining similar bias and coverage probabilities, our proposed MORA method is around three-fold faster than the offline KEE method and is computationally more efficient in processing data with a small data batch size n_b.

6.3 Fixed batch size n_b and increasing B

Now we consider a scenario where a sequence of data batches arrives with high speed. For convenience, we fix the data batch size as $n_b = 100$ but let B increase from 10 to 1000. Table 2 summarizes simulation results under the same model, as specified in Section 6.1.

Bias and coverage probability in $\boldsymbol{\alpha}$. Similar to what we observed in Table 1, the MORA method gives a similar level of bias and coverage probability as the offline KEE method: as

$\rho_1 = 0.1,\ \rho_2 = 0.5,\ n_b = 100$									
B	10			100			1,000		
	LM	KEE	MORA	LM	KEE	MORA	LM	KEE	MORA
$\boldsymbol{\alpha}$.ABIAS$\times 10^{-3}$	56.22	34.00	34.59	18.48	10.50	10.55	5.91	3.32	3.32
$\boldsymbol{\alpha}$.ASE$\times 10^{-3}$	70.13	41.26	43.23	23.46	13.05	13.18	7.46	4.12	4.13
$\boldsymbol{\alpha}$.ESE$\times 10^{-3}$	70.76	42.70	43.83	23.20	13.06	13.12	7.41	4.16	4.17
$\boldsymbol{\alpha}$.CP	0.947	0.944	0.949	0.955	0.951	0.952	0.949	0.948	0.948
C.Time (s)	0.01	0.08	0.03	0.07	0.62	0.18	5.89	17.63	2.68
R.Time (s)	0.01	0.07	0.02	0.02	0.57	0.15	0.38	12.12	2.26

$\rho_1 = 0.5,\ \rho_2 = 0.5,\ n_b = 100$									
B	10			100			1,000		
	LM	KEE	MORA	LM	KEE	MORA	LM	KEE	MORA
$\boldsymbol{\alpha}$.ABIAS$\times 10^{-3}$	56.80	34.00	34.55	19.26	10.50	10.56	6.16	3.32	3.32
$\boldsymbol{\alpha}$.ASE$\times 10^{-3}$	70.80	41.26	43.26	24.47	13.05	13.19	7.81	4.12	4.13
$\boldsymbol{\alpha}$.ESE$\times 10^{-3}$	71.77	42.69	43.76	24.16	13.06	13.12	7.74	4.16	4.17
$\boldsymbol{\alpha}$.CP	0.947	0.944	0.950	0.960	0.951	0.952	0.952	0.948	0.948
C.Time (s)	0.01	0.11	0.04	0.12	1.07	0.29	4.12	16.51	2.66
R.Time (s)	0.01	0.10	0.03	0.04	0.98	0.24	0.39	11.48	2.25

$\rho_1 = 0.9,\ \rho_2 = 0.5,\ n_b = 100$									
B	10			100			1,000		
	LM	KEE	MORA	LM	KEE	MORA	LM	KEE	MORA
$\boldsymbol{\alpha}$.ABIAS$\times 10^{-3}$	60.08	34.00	34.55	25.31	10.50	10.56	8.73	3.32	3.32
$\boldsymbol{\alpha}$.ASE$\times 10^{-3}$	74.68	41.26	43.34	32.46	13.05	13.19	11.18	4.12	4.13
$\boldsymbol{\alpha}$.ESE$\times 10^{-3}$	76.37	42.70	43.73	31.95	13.06	13.12	11.01	4.16	4.17
$\boldsymbol{\alpha}$.CP	0.949	0.944	0.951	0.960	0.951	0.952	0.949	0.948	0.948
C.Time (s)	0.01	0.13	0.04	0.11	1.02	0.28	5.79	17.78	3.07
R.Time (s)	0.01	0.12	0.03	0.04	0.94	0.24	0.44	12.42	2.59

Table 2: Simulation results under the linear state-space mixed model, summarized over 500 replications, with $n_b = 100$, $p = 5$, and B increasing from 10 to $1,000$.

the number of data batches B increases from 10 to 1000, $\boldsymbol{\alpha}$.bias decreases at an empirical rate of approximating $O(\sqrt{N_B})$, which further confirms the large-sample property given in Theorem 4.1. The coverage probability robustly stays around 95%. Similar to Section 6.2, estimation bias and coverage probability in our online regression method are robust across different ρ_1, but larger ρ_1 leads to slightly larger bias in the LM method due to its ignorance of dependence.

Computation time. As for computational efficiency, with a fixed data batch size n_b, both C.Time and R.Time in MORA increase linearly with B. When B is small, C.Time is lower for naive LM than for our online regression method, but this relationship reverses once B reaches $1,000$ due to the large data loading time. It is worth noting that both C.Time and R.Time in the offline KEE method are almost 10 times those in our online regression method. This further demonstrates the strong computational advantage of the MORA method, especially after a large sample size has accumulated over time.

6.4 Scalability

To elucidate the scalability of MORA in dealing with large-scale online regression analyses, here we show some numerical evidence regarding the computational efficiency of MORA with large p. In the simulation studies, we fix the total sample size as $N_B = 10^5$ and the number of data batches as $B = 25$, for a data batch size of $n_b = 4,000$. The dimensions of the observed and latent processes increase up to (i) $p = 1,000$ and $q = 5$ and (ii) $p = 2,000$ and $q = 10$, with individual autocorrelation coefficients $\rho_s \overset{i.i.d.}{\sim}$ Uniform$(0,1)$ for $s = 1,\dots,q$. As shown by the simulation results summarized in Table 3, our proposed MORA method is more than four-fold faster than the offline KEE method with no loss of statistical efficiency. This finding is similar to that in the low-dimensional simulation experiments.

$N_B = 10,000$	$p = 1000,\ q = 5$			$p = 2000,\ q = 10$		
	LM	KEE	MORA	LM	KEE	MORA
$\boldsymbol{\alpha}$.ABIAS$\times 10^{-3}$	14.24	3.59	3.64	15.03	3.61	3.88
$\boldsymbol{\alpha}$.ASE$\times 10^{-3}$	17.87	4.49	4.62	18.84	4.52	5.31
$\boldsymbol{\alpha}$.ESE$\times 10^{-3}$	18.36	4.49	4.56	19.02	4.52	5.58
$\boldsymbol{\alpha}$.CP	0.951	0.950	0.952	0.949	0.950	0.953
C.Time (min)	3.05	71.84	16.67	11.54	153.50	36.22
R.Time (min)	2.77	71.55	16.59	10.55	152.50	36.05

Table 3: Simulation results under the linear state-space mixed model with $p = 1,000$ and $p = 2,000$, summarized over 200 replications, with $N_B = 10^5$, $B = 25$ and $n_b = 4,000$.

7 SRTR Data Example

In an analysis of the kidney transplant data collected by the Scientific Registry of Transplant Recipients, we aim to evaluate the effects of certain key risk factors on serum creatinine levels one year post-transplantation. Many studies have found that post-transplant renal function in the first year is highly related to long-term kidney-transplant survival (Sundaram et al., 2002). We consider the scenario where transplant data batches arrive yearly during the 24-year period from 1994 to 2017, with $B = 24$ and $N_B = 158204$ recipients whose creatinine measurements are recorded in the first post-transplant year with no missing data are log-transformed and included in our analysis.

We apply the proposed linear mixed state-space model with the following risk factors as fixed effects: donor and recipient age (standardized), donor–recipient sex (1 for a homosexual pair and 0 otherwise), donor and recipient BMI (1 for obese and 0 for not obese), donor-recipient height ratio (1 for greater than 1 and 0 otherwise), donor–recipient weight ratio (1 for greater than 0.9 and 0 otherwise), donor–recipient race (1 for a homoracial pair and 0 otherwise), and duration of dialysis (0 for less than three years and 1 otherwise). We first perform a preliminary analysis by fitting a cross-sectional linear regression model to yearly individual data batches separately: see Figure 3. We plot the corresponding autocorrelation and partial correlation plots in Figure 4. It is clear that the estimated effects of time (in year) and donor age show autoregressive trends with an order-one correlation structure.

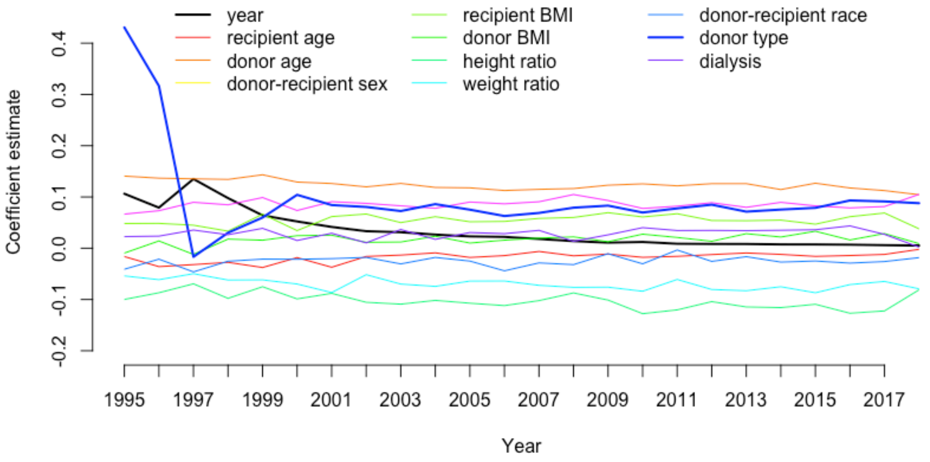

Figure 3: Preliminary cross-sectional analysis results showing trends in individual regression coefficient estimates obtained by fitting a linear regression model to each yearly data batch.

Therefore, we model these two risk factors as dynamic batch-specific effects that account for underlying heterogeneity over the sequence of data batches: see the estimated trace plots in Figure 5. Such an analysis can hardly be done via the offline KEE method due to the intensive computational burden incurred by both the large data batch size n_b and the cumulative sample size N_B. Therefore, we apply our proposed MORA method to sequentially update parameter estimates and standard errors.

Table 4 reports results from fitting a linear state-space mixed model using our proposed online regression method at the terminal year, 2017. Due to the large cumulative sample size in this streaming data setting, all P-values are too small to be useful for making conclusions (see Figure 6). Thus, we focus on point estimates, standard errors, and z values in Table 4, which allows us to rank the risk factors. The major findings are as follows. (i) Donor–recipient height ratio and donor–recipient weight ratio are the top two risk factors. Such an association between donor–recipient weight mismatch (donor<recipient) and graft failure has also been found in Miller et al. (2017); Tillmann et al. (2019). (ii) Recipients with younger ages and matched-race transplants show better graft function. (iii) Donor death, higher recipient or donor BMI, homosexual transplantation, and a longer dialysis duration may have negative effects on post-transplant renal function. This may provide health practitioners some insights on how to correctly analyze these types of cumulative electronic health records while accounting for dynamics and dependence. Additionally, the dynamic changes in the time effect and donor age effects are also shown in Figure 5. It is clear that baseline serum creatinine levels decrease from 1994 to 2003 before stabilizing, and that donor age also shows a slowly decreasing trend. These trends might be related to the FDA's approval of immunosuppressive drugs such as CellCept in 1995 and Tacrolimus in 1997 for use in kidney transplantation.

Figure 6 shows the trajectories of $-\log_{10}(P)$ values over 24 years: the 10-base log P-

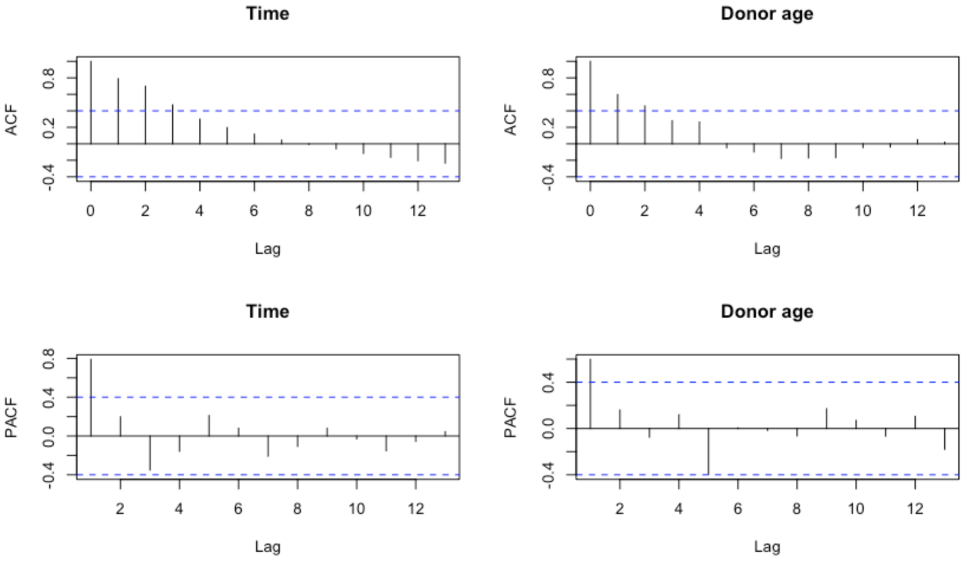

Figure 4: Empirical ACF and PACF plots for the regression coefficient estimates in the preliminary analysis. It is clear that the risk factors year effect and donor age follow a stationary AR(1) process.

values of the z-test for each regression coefficients are zero. Among all these risk factors, donor–recipient height ratio turns out to have the largest effect. To characterize the overall significance level for each covariate over the 24-year period, we calculate a summary statistic as the area under the P-value curve. Intuitively, a larger area under the curve indicates a stronger association with the outcome. We use this metric to rank predictors instead of claiming statistical significance at the traditional cutoff $P = 0.05$ because most risk factors have P-values smaller than 0.05 due to the large sample size. Ranking gives more-important information about and is a more-desirable evaluation of outcome–covariate associations than a binary decision of rejection or acceptance based on a universal cutoff. For most of these curves, the ranking of overall significance by these areas is well aligned with the ranking of the P-values obtained at the terminal year, 2017, except for recipient age and donor–recipient weight ratio, which cross over at around 2014. This also happens to donor–recipient race and donor type. By looking into these trajectories rather than only the end-point P-values, we can see that recipient age has, overall, a more significant association with post-transplant renal function than does weight ratio. This summary statistic provides useful evidence in addition to the terminal P-values.

8 Concluding Remarks

As streaming data becomes one of the most pervasive data collection schemes in the field of data science, there is a surge increase in the number of applications that require real-time processing of massive data arriving with high velocity. Conventional offline techniques suffer from many limitations when applied to streaming data-analytic tasks. Online learning

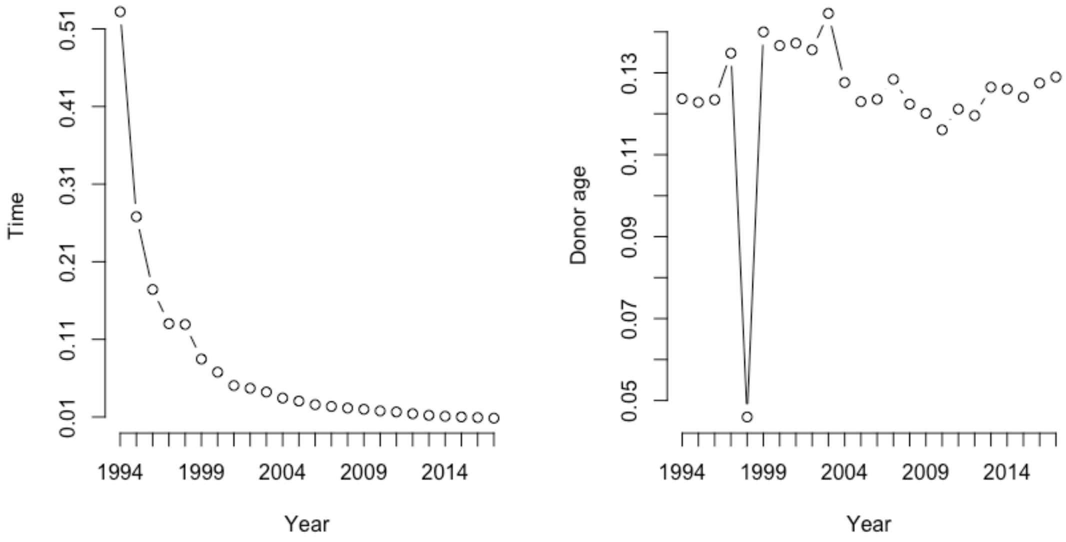

Figure 5: Trace plots of the dynamic effects of time and donor age over the 24-year period.

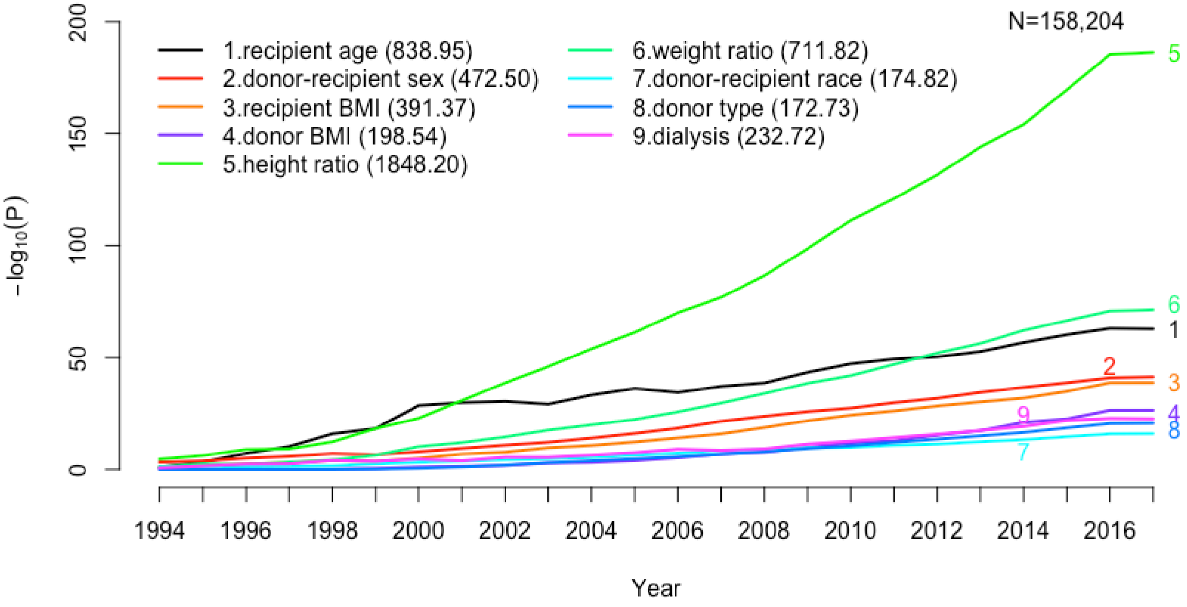

Figure 6: Trajectories of $-\log_{10}(P)$ over yearly data batches from 1994 to 2017, each for one risk factor. Numbers on the left y-axis are the negative logarithm of P-values obtained by z tests and labels on the x-axis correspond to the end of each year. The values in the brackets next to the covariate names denote respective areas under the P-value curves.

	Estimate	Std. Err $\times 10^{-3}$	z value
Recipient age	-0.015	0.87	-16.83
Donor–recipient sex	0.069	5.11	13.59
Recipient BMI	0.049	3.69	13.15
Donor BMI	0.021	1.97	10.79
Donor–recipient height ratio	-0.120	4.11	-29.17
Donor–recipient weight ratio	-0.085	4.74	-17.95
Donor–recipient race	-0.041	4.92	-8.31
Donor type	0.058	6.01	9.50
Duration of dialysis	0.025	2.51	9.95

Table 4: Results from fitting a linear state-space mixed model with our proposed MORA method at the end of 2017. The total sample size is $N_B = 158,204$, with $p = 9$, $q = 2$ and $B = 24$.

techniques are promising for tackling the emerging challenges of data stream mining. The history of sequential processing may be dated back to the 1950s when Robbins and Monro (1951) proposed a theory of stochastic approximation. A variety of online learning methods such as the stochastic gradient descent algorithm were developed thereafter (Sakrison, 1965; Duchi et al., 2011; Toulis and Airoldi, 2015). However, there are two major issues that are not fully addressed by methods in this area of research. (i) Most methods are motivated by applications in the field of engineering where point estimation or prediction rather than statistical inference is the main focus, while in biomedical research. (ii) There are only fixed, common parameters in model specifications, which prevents the analysis from addressing dynamic heterogeneity over data streams. As shown in the marginal linear regression analysis, ignoring serial heterogeneity may lead to large estimation bias and low statistical efficiency.

These technical gaps were partially filled in by Luo and Song (2020) in the setting of cross-sectional data with homogeneity assumptions on model parameters. To account for dynamic heterogeneity, we propose a new framework of linear state-space models in which dynamically changing regression coefficients are allowed to follow a Markov process (e.g., an AR(1) process). The main idea underlying our estimation method is rooted in the EM algorithm, where the E step is calculated using the Kalman recursive technique, and, in the M step, summary statistics rather than historical subject-level data are used to facilitate the efficiency of online regression analysis, as in Luo and Song (2020). Both the proposed statistical methodology and computational algorithms have been investigated for theoretical guarantees and examined numerically via extensive simulation studies. The proposed MORA method with data heterogeneity is computationally more efficient with smaller data batch sizes and has no loss of statistical efficiency in comparison to the offline oracle method.

It is worth noting that our method is robust against different ways of data splitting schemes that give rise to certain latent process dynamics over data batches that may be different from the true dynamics. By an analogy to the notion of working correlation structures in generalized estimating equations (GEE) (Liang and Zeger, 1986), we term the resulting specification of the Markov transitions as the "working dynamics" in our framework. In the presence of a discrepancy between the working dynamics and the true dynamics, as long as the mean model is properly specified, our MORA method still enjoys estimation consistency and valid statistical inference because it is constructed with unbiased estimating functions for the fixed effects of interest. Similar to misspecified working correlation structures in GEEs (Wang and Carey, 2003), it is not surprising that there could be a loss of statistical efficiency. We ran some additional simulation experiments (results are not shown here) that have confirmed these points of view. To examine mean model misspecification, a goodness-of-fit test such as the generalized method of moments (GMM) (Hansen, 1982) may be invoked. This requires an extension to the MORA method by following the line of, for example, quadratic inference functions (QIF) (Qu et al., 2000), which will be considered in our future work.

It is also noteworthy that our proposed incremental inference procedure only offers continuously updated standard errors of parameter estimates rather than a valid rejection rule based on some test statistic. With a fixed number of data batches, alpha spending functions used in sequential clinical trials provide a promising procedure to properly control type I error in sequential testing (Lan and Demets, 1983). A more-challenging technical problem to be solved is how to develop a proper alpha spending function suitable for a number of data batches diverging to infinity. Another direction worthy of further exploration is the case of nonstationary latent processes such as random walks. One technical challenge pertains to the fact that inter data batch correlation does not decay over the sequence of data batches, a case beyond ϕ-mixing processes. In this article, large-sample properties are established in a ϕ-mixing process framework. A related problem of interest is in testing the stationarity of the underlying latent process, that is $H_0 : \rho = 1$ versus $H_1 : 0 < \rho < 1$, where ρ is the autocorrelation parameter. This is a difficult problem because the hypothetical value in the null hypothesis is on the boundary of the parameter space. In this article, we start with the linear state-space model with Gaussian outcomes. This framework may be relaxed to non-Gaussian responses to analyze other types of streaming data. For example, in biomedical fields where data streams are captured by wearable devices, data may be discrete physical activity counts, binary or highly skewed physiological measurements such as body temperature. Therefore, further extensions to handle non-Gaussian streaming data represent an important future research area as part of new analytic tools for high frequency mobile health data.

Acknowledgements

The authors are grateful to the editor, associate editor and two anonymous referees for their helpful comments and suggestions, which led to a substantial improvement of this article. The authors also thank Dr. Alfred O. Hero for his constructive discussion on an early version of this article. Dr. Peter X.-K. Song's research was supported by the National Science Foundation grants DMS 1811734 and DMS 2113564.

Appendix A.

In this appendix we prove the following theorem from Section 4.

Theorem 1 *Under the regularity conditions (C1)–(C4), for fixed ρ, ϕ and δ, $\tilde{\boldsymbol{\alpha}}_b$ is consistent and asymptotically normal, namely,*

$$\sqrt{N_b}(\tilde{\boldsymbol{\alpha}}_b - \boldsymbol{\alpha}_0) \xrightarrow{d} \mathcal{N}_p\left\{\mathbf{0}, \mathbb{J}^{-1}(\boldsymbol{\alpha}_0)\right\} \ \text{ as } N_b = \sum_{j=1}^{b} n_j \to \infty,$$

where $\mathbb{J}(\boldsymbol{\alpha}_0) = \mathbb{S}^\top(\boldsymbol{\alpha}_0)\mathbb{V}^{-1}(\boldsymbol{\alpha}_0)\mathbb{S}(\boldsymbol{\alpha}_0)$ is the Godambe information matrix of the inference function in Equation (2).

Proof: We take the first-order Taylor expansion of the aggregated estimating equation $\tilde{\boldsymbol{U}}_b(\tilde{\boldsymbol{\alpha}}_b)$ around $\boldsymbol{\alpha}_0$, $\tilde{\boldsymbol{U}}_b(\tilde{\boldsymbol{\alpha}}_b) = \tilde{\boldsymbol{U}}_b(\boldsymbol{\alpha}_0) + \frac{\partial \tilde{\boldsymbol{U}}_b(\boldsymbol{\alpha})}{\partial \boldsymbol{\alpha}^\top}(\tilde{\boldsymbol{\alpha}}_b - \boldsymbol{\alpha}_0) = \mathbf{0}$. It follows that

$$\sqrt{N_b}(\tilde{\boldsymbol{\alpha}}_b - \boldsymbol{\alpha}_0) = \left\{-\frac{1}{N_b}\frac{\partial \tilde{\boldsymbol{U}}_b(\boldsymbol{\alpha})}{\partial \boldsymbol{\alpha}^\top}\right\}^{-1}\left\{\frac{1}{\sqrt{N_b}}\tilde{\boldsymbol{U}}_b(\boldsymbol{\alpha}_0)\right\}, \tag{3}$$

where $\tilde{\boldsymbol{S}}_b(\boldsymbol{\alpha}) = -\frac{\partial \tilde{\boldsymbol{U}}_b(\boldsymbol{\alpha})}{\partial \boldsymbol{\alpha}^\top} = \sum_{j=1}^{b} \boldsymbol{X}_j^\top\left\{\boldsymbol{X}_j + \boldsymbol{Z}_j\boldsymbol{L}_j(\boldsymbol{\alpha})\right\}$.

The second factor on the right-hand side of Equation (3) may be written as

$$\frac{1}{\sqrt{N_b}}\tilde{\boldsymbol{U}}_b(\boldsymbol{\alpha}) = \frac{1}{\sqrt{N_b}}\sum_{j=1}^{b}\boldsymbol{X}_j^\top(\boldsymbol{y}_j - \boldsymbol{X}_j\boldsymbol{\alpha} - \boldsymbol{Z}_j\boldsymbol{m}_j).$$

Denote $\boldsymbol{U}_j = \boldsymbol{X}_j^\top(\boldsymbol{y}_j - \boldsymbol{X}_j\boldsymbol{\alpha} - \boldsymbol{Z}_j\boldsymbol{m}_j) = \sum_{i\in D_j}\boldsymbol{u}_{ji} = \sum_{i\in D_j}\boldsymbol{x}_{ji}(y_{ji} - \boldsymbol{x}_{ji}^\top\boldsymbol{\alpha} - \boldsymbol{z}_{ji}^\top\boldsymbol{m}_j)$. Then, $\tilde{\boldsymbol{U}}_b = \sum_{j=1}^{b}\boldsymbol{U}_j$. Let $\mathscr{F}_j$ represent the σ-field generated by $D_j^\star$. It is easy to show that $\mathbb{E}[\boldsymbol{U}_j \mid \mathscr{F}_{j-1}] = \mathbf{0}$. Then $\{(\boldsymbol{U}_j, \mathscr{F}_j) : j = 1, 2, \ldots\}$ forms a sequence of martingale differences with means of $\mathbf{0}$.

To derive the joint distribution of $\tilde{\boldsymbol{U}}_b$, we apply the Cramér-Wold theorem (Cramér and Wold, 1936). For any nonrandom, nonzero vector $\boldsymbol{a} = (a_1, \ldots, a_p)^\top \in \mathbb{R}^p$, letting $\boldsymbol{u}_{ji} = (u_{ji,1}, \ldots, u_{ji,p})^\top$, we write

$$\boldsymbol{a}^\top\tilde{\boldsymbol{U}}_b = \sum_{i=1}^{N_b}\sum_{d=1}^{p} a_d u_{i,d} = \sum_{i=1}^{N_b} u_i^\star. \tag{4}$$

Since $\{\boldsymbol{\beta}_b\}$ is a stationary AR(1) process, it is a ϕ-mixing process (Billingsley, 1968). Given $\boldsymbol{\beta}_j$, $\{\boldsymbol{u}_{ji}\}$ in Equation (4) is conditionally independent of each other with $\mathbb{E}[\boldsymbol{u}_{ji}] = \mathbf{0}$, and thus, $\{\boldsymbol{u}_{ji}\}$ is a centred ϕ-mixing centred process (Billingsley, 1968). It follows that $\{u_i^\star\}_{i=1}^{N_b}$ is also a stationary ϕ-mixing centred stochastic process whose second moments are given by

$$\sigma_{N_b}^2 = \text{var}\left(\sum_{i=1}^{N_b} u_i^\star\right) \to \infty \text{ as } N_b \to \infty.$$

Now we check the Lindeberg condition. For any $\epsilon > 0$,

$$\begin{aligned}\sum_{i=1}^{N_b}\mathbb{E}\left\{(u_i^\star)^2\mathbf{1}[|u_i^\star| > \epsilon\sigma_{N_b}]\right\} &= \sum_{i=1}^{N_b}\mathbb{E}\left\{\left(\sum_{d=1}^{p} a_d u_{i,d}\right)^2\mathbf{1}\left[\sum_{d=1}^{p}|a_d u_{i,d}| > \epsilon\sigma_{N_b}\right]\right\} \\ &\leq \sum_{i=1}^{N_b}\sum_{d=1}^{p} a_d^2\mathbb{E}\left\{u_{i,d}^2\mathbf{1}\left[\sum_{d=1}^{p}|u_{i,d}| > \frac{\epsilon\sigma_{N_b}}{\max_d |a_d|}\right]\right\},\end{aligned}$$

where $\mathbf{1}[\cdot]$ is an indicator function. Since $\sigma_{N_b} \to \infty$ and $\max|a_d| < \infty$, we have that $\mathbf{1}\left[\sum_{d=1}^{p} |u_{i,d}| > \frac{\epsilon\sigma_{N_b}}{\max\limits_{i} |a_i|}\right] \overset{a.s.}{\to} 0$. Additionally, because $\mathbb{E}[u_{i,d}^2] < \infty$ and $P(u_{i,d} = \infty) = 0$, it follows that

$$\sum_{i=1}^{N_b} \mathbb{E}\left\{(u_i^\star)^2 \mathbf{1}[|u_i^\star| > \epsilon\sigma_{N_b}]\right\} \to 0 \text{ as } N_b \to \infty,$$

so the Lindeberg condition holds for $\{u_i^\star\}$.

The central limit theorem for the ϕ-mixing stochastic process $\{u_i^\star\}$ (Peligrad, 1986) implies that

$$\frac{\sum_{i=1}^{N_b} u_i^\star}{\sigma_{N_b}} \overset{d}{\to} \mathcal{N}(0, 1),$$

where $\sigma_{N_b}^2 = \boldsymbol{a}^\top \text{var}[\tilde{\boldsymbol{U}}_b]\boldsymbol{a}$. Moreover, applying the Cramér-Wold theorem, we have that

$$\frac{1}{\sqrt{N_b}} \tilde{\boldsymbol{U}}_b \overset{d}{\to} \mathcal{N}_p\left\{\mathbf{0}, \mathbb{V}(\boldsymbol{\alpha}_0)\right\},$$

where $\mathbb{V}(\boldsymbol{\alpha}_0) = \lim\limits_{b\to\infty} \frac{1}{N_b} \vec{\boldsymbol{X}}_b^\top \text{var}[\tilde{\boldsymbol{U}}_b] \vec{\boldsymbol{X}}_b = \lim\limits_{b\to\infty} \tilde{\boldsymbol{V}}_b$, where $\vec{\boldsymbol{X}}_b = (\boldsymbol{X}_1^\top, \ldots, \boldsymbol{X}_b^\top)^\top$ is a matrix of combined covariates with dimension $N_b \times p$.

Applying the above arguments to Equation (3), by the central limit theorem and Slutsky's theorem, we obtain that

$$\sqrt{N_b}(\boldsymbol{\alpha} - \boldsymbol{\alpha}_0) \overset{d}{\to} \mathcal{N}_p\left\{\mathbf{0}, \mathbb{J}^{-1}(\boldsymbol{\alpha}_0)\right\} \text{ as } N_b \to \infty,$$

where $\mathbb{J}(\boldsymbol{\alpha}_0) = \mathbb{S}^\top(\boldsymbol{\alpha}_0)\mathbb{V}(\boldsymbol{\alpha}_0)^{-1}\mathbb{S}(\boldsymbol{\alpha}_0)$.

References

Bifet, A., Maniu, S., Qian, J., Tian, G., He, C., and Fan, W. (2015). Streamdm: Advanced data mining in spark streaming. In *IEEE International Conference on Data Mining Workshop, ICDMW 2015, Atlantic City, NJ, USA, November 14–17, 2015*, pages 1608–1611.

Billingsley, P. (1968). *Convergence of Probability Measures.* Wiley, New York.

Broderick, T., Boyd, N., Wibisono, A., Wilson, A. C., and Jordan, M. I. (2013). Streaming variational bayes. *in Advances in Neural Information Processing Systems*, pages 1727–1735.

Cappé, O. (2011). Online em algorithm for hidden markov models. *Journal of Computational and Graphical Statistics*, 20(3):728–749.

Cappé, O. and Moulines, E. (2009). Online expectation-maximization algorithm for latent data models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(3):593–613.

Chen, W., Chen, L., Chen, Z., and Tu, S. (2005). A realtime dynamic traffic control system based on wireless sensor network. In *2005 International Conference on Parallel Processing Workshops (ICPPW‘05)*, pages 258–264.

Ciuciu, P., Abry, P., Rabrait, C., and Wendt, H. (2008). Log wavelet leaders cumulant based multifractal analysis of evi fmri time series: Evidence of scaling in ongoing and evoked brain activity. *IEEE Journal of Selected Topics in Signal Processing*, 2(6):929–943.

Cramér, H. and Wold, H. (1936). Some theorems on distribution functions. *Journal of the London Mathematical Society*, 11(4):290–294.

Czado, C. and Song, P. X.-K. (2008). State space mixed models for longitudinal observations with binary and binomial responses. *Statistical Papers*, 49:691–714.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 39(1):1–38.

Dias, D. and Cunha, J. a. P. S. (2018). Wearable health devices—vital sign monitoring, systems and technologies. *Sensors (Basel)*, 18(8):2414.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159.

Frigola, R., Chen, Y., and Rasmussen, C. E. (2014). Variational gaussian process state-space models. *in Advances in Neural Information Processing Systems*, pages 3680–3688.

Hansen, L. P. (1982). Large sample properties of generalized method of moments estimators. *Econometrica*, 50(4):1029–1054.

Harvey, A. C. (1981). *Time Series Models.* Allan, Oxford.

Jørgensen, B., Lundbye-Christensen, S., Song, P. X.-K., and Sun, L. (1999). A state-space model for multivariate longitudinal count data. *Biometrika*, 86:169–181.

Jørgensen, B. and Song, P. X.-K. (2007). Stationary state space models for longitudinal data. *The Canadian Journal of Statistics*, 35(4):461–483.

Kitagawa, G. (1987). Non-gaussian state-space modeling of nonstationary time series (with discussion). *Journal of the American Statistical Association*, 82:1032–1063.

Lan, K. K. G. and Demets, D. L. (1983). Discrete sequential boundaries for clinical trials. *Biometrika*, 70(3):659–663.

Liang, K.-Y. and Zeger, S. (1986). Longitudinal data analysis using generalized linear models. *Biometrika*, 73:13–22.

Luo, L. and Song, P. X.-K. (2020). Renewable estimation and incremental inference in generalized linear models with streaming data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 82:69–97.

L'Heureux, A., Grolinger, K., Elyamany, H. F., and Capretz, M. A. M. (2017). Machine learning with big data: Challenges and approaches. *IEEE Access*, 5:7776–7797.

Miller, A. J., Kiberd, B. A., Alwayn, I. P., Odutayo, A., and Tennankore, K. K. (2017). Donor-recipient weight and sex mismatch and the risk of graft loss in renal transplantation. *Clinical Journal of the American Society of Nephrology*, 12(4):669–676.

Peligrad, M. (1986). *Recent advances in the central limit theorem and its weak invariance principle for mixing sequences of random variables (a survey)*. Birkhäuser, Boston, MA.

Qu, A., Lindsay, B., and Li, B. (2000). Improving generalised estimating equations using quadratic inference functions. *Biometrika*, 87:823–76.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407.

Robinson, G. K. (1991). That blup is a good thing: The estimation of random effects. *Statistical Science*, 6(1):15–32.

Sadik, S., Gruenwald, L., and Leal, E. (2018). Wadjet: Finding outliers in multiple multi-dimensional heterogeneous data streams. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 1232–1235.

Sakrison, D. J. (1965). Efficient recursive estimation: Application to estimating the parameter of a covariance function. *International Journal of Engineering Science*, 3(4):461–483.

Schifano, E. D., Wu, J., Wang, C., Yan, J., and Chen, M.-H. (2016). Online updating of statistical inference in the big data setting. *Technometrics*, 58(3):393–403.

Song, P. X.-K. (2007). *Correlated data analysis: modeling, analytics, and applications.* Springer-Verlag, New York.

Sundaram, H., Maureen, A., S.Cherikh, W., B.Tolleris, C., A.Bresnahan, B., and P.Johnson, C. (2002). Post-transplant renal function in the first year predicts long-term kidney transplant survival. *Kidney International*, 62(1):311–318.

Tillmann, F.-P., Quack, I., Woznowski, M., and Rump, L. C. (2019). Effect of recipient-donor sex and weight mismatch on graft survival after deceased donor renal transplantation. *PLoS One*, 14(3).

Titterington, D. M. (1984). Recursive parameter estimation using incomplete data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 46(2):257–267.

Toulis, P. and Airoldi, E. M. (2015). Scalable estimation strategies based on stochastic approximations: Classical results and new insights. *Statistics and Computing*, 25(4):781–795.

Wang, Y.-G. and Carey, V. (2003). Working correlation structure misspecification, estimation and covariate design: Implications for generalised estimating equations performance. *Biometrika*, 90(1):29–41.

West, M. and Harrison, P. J. (1997). *Bayesian Forecasting and Dynamic Models*. Springer-Verlag, New York, 2nd edition.

Zhang, Y., Jansen, B. J., and Spink, A. (2009). Time series analysis of a web search engine transaction log. *Information Processing and Management*, 45(2):230–245.