Online Decentralized Leverage Score Sampling for Streaming Multidimensional Time Series

Abstract

Estimating the dependence structure of multidimensional time series data in real-time is challenging. With large volumes of streaming data, the problem becomes more difficult when the multidimensional data are collected asynchronously across distributed nodes, which motivates us to sample representative data points from streams. We propose a *leverage score sampling* (LSS) method for efficient online inference of the streaming vector autoregressive (VAR) model. We define the leverage score for the streaming VAR model so that the LSS method selects informative data points in real-time with statistical guarantees of parameter estimation efficiency. Moreover, our LSS method can be directly deployed in an asynchronous decentralized environment, e.g., a sensor network without a fusion center, and produce asynchronous consensus online parameter estimation over time. By exploiting the temporal dependence structure of the VAR model, the LSS method selects samples independently on each dimension and thus is able to update the estimation asynchronously. We illustrate the effectiveness of the LSS method in synthetic, gas sensor and seismic datasets.

1 INTRODUCTION

Understanding the dependence structure of streaming multidimensional time series in real-time is a “space-time” challenge due to (1) the temporal dependency and infinite sample size of data streams in time, and (2) cross-correlation among multidimensional streams and information transition in the data acquisition network on space. The multidimensional streaming data are commonly collected from a network system with each node corresponding to one marginal dimension of the streams. The multidimensional streams contain complex temporal and cross-sectional dependency, usually along with a huge volume of data. Accurately and efficiently estimating the dependence structure is crucial, especially for real-time inference tasks, but the estimating process is time-consuming. Sampling is a natural and efficient way to reduce the data size and speed up the computation. Meanwhile, when the multidimensional streams are collected across distributed nodes asynchronously, it is not practical to transfer all data to one computing node and process them with an increasing data volume. One reasonable approach to retrieve dependence information is to perform asynchronous consensus estimation on each node in the decentralized computing framework [50, 51]. Sampling can relief the storage pressure and minimize the communication cost in such decentralized network.

The vector autoregressive (VAR) model, one of the most popular and fundamental time series models, provides a mechanism for capturing complex temporal dependency and cross-correlation among the multidimensional time series. Inferring these dependencies requires both efficient methodology and intensive computational efforts. Precisely understanding theses dependencies facilitates the interpretation of the model and improves prediction accuracy.

In this work, we introduce a *leverage score sampling* (LSS) method that can efficiently estimate the dependence structure from asynchronous multidimensional streaming time series. By exploiting the VAR model, we parameterize the temporal dependence (auto/cross-correlation) structure and propose the streaming statistical leverage scores for streaming sampling. We also seek to directly deploy this method to an asynchronous decentralized network, which has limited energy, memory and processing resource. In these cases, finding the informative data points is highly desired for accelerating the estimation process and boosting the transmission of the streaming data in the decentralized network system.

Challenges: In this paper, we focus on designing a sampling strategy that can improve the parameter estimation accuracy and maintain the computation efficiency. We address a few specific challenges in sampling streaming multidimensional time series. First, how do we find a subset of samples that efficiently capture the

Preliminary work. Under review by AISTATS 2019. Do not distribute.

temporal structure under a multidimensional setting? The proposed sampling method aims to find influential data points, which are highly efficient for estimating the parameter matrix of the VAR model, in real-time to reduce evaluation times without losing too much accuracy. Second, how do we adapt the importance sampling method to the streaming and decentralized environment? We utilize the VAR model to decompose the dependence structure and distribute it to each node so that the sampling method can be applied on each node independently.

Prior Work: Sampling is an important data reduction approach for *reducing the computational cost and memory usage*, and it is widely used in matrix approximation or sketching [15, 14, 3, 53], kernel approximation [34, 1], graph sparsification [44, 27], linear regression [31, 13, 38], and etc. Especially, sampling method based on leverage score is one of the most popular techniques [36, 11, 33]. Random sampling with probability proportional to exact or approximated leverage scores can yield high accuracy on model parameter estimation for linear regression [32, 38], logistic regression [46] and kernel ridge regression [2].

On the other hand, sampling as the subset selection-which *optimizes a specified objective function* leads to numerous applications in image, video, speech summarization [17, 18, 21, 42, 30, 28], and bioinformatics [49, 25]. Most of the existing methods treat the samples independently and ignore the dependence information among the samples, except the most recent work of [18] that selects the sequential data based on Markov models.

Meanwhile, literatures on sampling for *streaming data* have focused on column sampling [11], spectral sparsification or subgraph sampling for graph streams [27, 10], data management [12, 16], and clustering [43]. To the best of our knowledge, the study on sampling with an objective of recovering the dependence information of streaming data is still lacking.

Paper Contributions: In this paper, we develop a novel sampling method for estimating temporal dependence structure of multidimensional streaming multidimensional time series. Our leverage score sampling (LSS) method is based on the statistical leverage score of vector autoregressive model for online selecting representative data points, which are later used to estimate the VAR model parameter matrix.

1. The LSS differs substantially from other leverage-based sampling methods. The LSS focuses on selecting informative data points that contribute to the estimation efficiency of the VAR model parameter matrix, which is a model-based surrogate for temporal dependence structure of the multidimensional time series streams.

2. We provide a theoretical guarantee that the LSS method yields a better estimation efficiency for the VAR model parameter matrix than naive sampling methods.

3. Not only is the LSS method fast and accurate for estimating temporal dependence structure, but it can also be applied in an asynchronous decentralized environment where traditional leverage-based sampling methods cannot.

As an illustration, we present a single-pass streaming sampling algorithm on the asynchronous decentralized framework for consensus optimization. We demonstrate the practical effectiveness with such asynchronous decentralized environment, both in parameter estimation on K-dimensional VAR(p) synthetic data streams, as well as in real large-scale sensor data prediction tasks.

2 BACKGROUND

2.1 Notation

A curly capital letter $\mathcal{A}$ is used for set and collection of sets. The upper-case letters $\mathbf{A}$ or A are used for matrices and operators. A lower-case bold letter $\mathbf{a}$ is used for vector and a lower-case letter a is used for scalar. Specifically, we reserve $\mathbb{E}(\cdot)$ to denote the expectation operator. The integers are denoted by $\mathbb{Z}$, and real numbers are denoted by $\mathbb{R}$. We denote the identity matrix of dimension n by $\mathbf{I}_n \in \mathbb{R}^{n\times n}$. We use $1_{\{\cdot\}}$ to denote the indicator function. We use $\mathbf{\Sigma}_1 \prec \mathbf{\Sigma}_2$, for two non-negative-definite matrices $\mathbf{\Sigma}_1$ and $\mathbf{\Sigma}_2$, to indicate that $\mathbf{\Sigma}_2 - \mathbf{\Sigma}_1$ is positive definite. We denote the transpose of a matrix $\mathbf{A}$ as $\mathbf{A}'$, the determinant of a matrix $\mathbf{A}$ as $\det(\mathbf{A})$ and vectorization operator as $\mathrm{vec}(\cdot)$. The Kronecker product is denoted by $\otimes$. Finally, $\xrightarrow{d}$ denotes the convergence in distribution, $\triangleq$ means *defined to be equal to*, $||\cdot||_2$ denotes the vector ℓ_2-norm, and $||\cdot||_F$ denotes the matrix Frobenius norm.

2.2 Vector Autoregressive Model

Time series data, one of the most representative classes of dependent data, which contain temporal dependence structure among samples, are often modeled by the *vector autoregressive* (VAR) models [4, 24, 47]. VAR model represents a family of time series models that offers a broad framework for capturing complex temporal and longitudinal interrelationship among the multidimensional time series data. A time series $\mathbf{y}_t \in \mathbb{R}^K$ follows a K-dimensional vector autoregressive model of order p, i.e., VAR(p), if

$$\mathbf{y}_t = \sum_{i=1}^{p} \mathbf{\Phi}_i \mathbf{y}_{t-i} + \mathbf{e}_t, \quad t \in \mathbb{Z}, \tag{1}$$

where $\mathbf{\Phi}_i$'s are $K \times K$ matrices, and $\mathbf{e}_t$ is a sequence of independent and identically distributed (i.i.d.) random vectors with mean zero and and finite non-singular covariance matrix $\mathbb{E}[\mathbf{e}_t\mathbf{e}_t'] = \mathbf{\Psi}$. Let $\mathbf{\Phi}(z) = \mathbf{I}_K - \sum_{j=1}^{p} \mathbf{\Phi}_j z^j$ be the the associated characteristic matrix polynomial, where $\mathbf{I}_p$ is the $p \times p$ identity matrix. We assume that $\det(\mathbf{\Phi}(z)) \neq 0$ on the complex unit disk $\{z \in \mathbb{C} : \ |z| \leq 1\}$. In this case, there is a unique stationary solution $\mathbf{y}_t$ of (1), which is expressed as a causal linear filter of $(\mathbf{e}_t)$ (Theorem 11.3.1 of [5]).

We rewrite the VAR(p) model as

$$\mathbf{y}_t' = \mathbf{x}_t'\mathbf{B} + \mathbf{e}_t', \quad t \in \mathbb{Z}, \tag{2}$$

where $\mathbf{B} = [\Phi_1', \Phi_2', \cdots, \Phi_p']'$ is the $Kp \times K$ model parameter matrix, $\mathbf{x}_t = (\mathbf{y}_{t-1}', \mathbf{y}_{t-2}', \cdots, \mathbf{y}_{t-p}')'$ is a column vector of length $m = Kp$. We assume that the covariance matrix $\Gamma = \mathbb{E}[\mathbf{x}_t\mathbf{x}_t']$ is non-singular. The model parameter estimation can be done through Ordinary Least-Squares (OLS) estimate [45], which is $\hat{\mathbf{B}}_{OLS} = \left(\sum_{t=1}^{T} \mathbf{x}_t\mathbf{x}_t'\right)^{-1} \sum_{t=1}^{T} \mathbf{x}_t\mathbf{y}_t'$. The computational cost of estimating the model parameter matrix $\mathbf{B}$ is $O(TK^2p^2)$.

2.3 Statistical Leverage Score

The VAR(p) model can be expressed in the form of linear model. In general, consider the parameter estimation of a linear model

$$\mathbf{y} = X\boldsymbol{\beta}_0 + \boldsymbol{\varepsilon}, \tag{3}$$

where $\mathbf{y}$ is the response vector, X is the design matrix, $\boldsymbol{\beta}_0$ is the parameter vector, and $\boldsymbol{\varepsilon}$ is the i.i.d. noise vector. The unknown parameter $\boldsymbol{\beta}_0$ in this model can be estimated as

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} ||\mathbf{y} - X\boldsymbol{\beta}||_2^2 = (X'X)^{-1}X'\mathbf{y}, \tag{4}$$

in which case the predicted value is $\hat{\mathbf{y}} = H\mathbf{y}$ with the so called Hat Matrix $H = X(X'X)^{-1}X'$. The i-th diagonal element of H,

$$\ell_{ii} = \mathbf{x}_i'(X'X)^{-1}\mathbf{x}_i, \tag{5}$$

is the *statistical leverage score* of the i-th sample, where $\mathbf{x}_i'$ is the i-th row of X. Alternatively, the leverage score can be expressed as

$$\ell_{ii} = ||\mathbf{u}_i||_2^2,$$

where $\mathbf{u}_i'$ is the i-th row of U, which can be any orthogonal basis for the column space of X, i.e., $H = UU'$ [14]. The statistical leverage scores have been used to regression diagnostics and to quantify the influential observations, which is critical for the leverage-based importance sampling.

3 LEVERAGE SCORE SAMPLING FOR TIME SERIES DATA

The *leverage score sampling* (LSS) method for streaming time series data utilizes the structure information of the underlying dynamic model to efficiently select informative samples. The information contained in multidimensional time series are projected onto a one-dimensional space through the LSS procedure, which results in an easy-to-implement and efficient sampling criterion.

We use the VAR(p) model to characterize the temporal dependence structure of K-dimensional time series, which keeps the interoperability, compatibility and avoids the overparameterization. For a fixed-sample-size case, suppose we observe the K-dimensional time series at T time points, $\{\mathbf{y}_t | t = 1, \ldots, T\}$. The dependence structure between data points can be modeled through a VAR(p) model. Our goal is to select a set of samples that well represent the underlying times series. This can be formulated as finding a Borel subset $\mathcal{E}$ of $\mathbb{R}^m$, $m = Kp$, so that $\mathbf{y}_t$ is selected whenever $\mathbf{x}_t \in \mathcal{E}$. Then the set of time points selected is $\mathcal{S} = \{1 \leq t \leq T : \ \mathbf{x}_t \in \mathcal{E}\}$. The least square estimator of $\boldsymbol{B}$ based on the selected sample then becomes [23]

$$\begin{aligned}\hat{\mathbf{B}}_S &= \left(\sum_{t\in\mathcal{S}} \mathbf{x}_t\mathbf{x}_t'\right)^{-1} \left(\sum_{t\in\mathcal{S}} \mathbf{x}_t\mathbf{y}_t'\right) \\ &= \left(\sum_{t=1}^{T} 1_{\{\mathbf{x}_t\in\mathcal{E}\}}\mathbf{x}_t\mathbf{x}_t'\right)^{-1} \left(\sum_{t=1}^{T} 1_{\{\mathbf{x}_t\in\mathcal{E}\}}\mathbf{x}_t\mathbf{y}_t'\right).\end{aligned}$$

The *leverage score sampling* method which finds subset $\mathcal{E}_{lev}$ of $\mathbb{R}^m$ and corresponding $\mathcal{S}_{lev}$ is according to the sampling rule

$$h_{ii} \triangleq \mathbf{x}_t'\mathbf{\Gamma}^{-1}\mathbf{x}_t > r^2 \tag{6}$$

for some carefully chosen sampling criterion r, where we recall that $\mathbf{\Gamma} = \mathbb{E}[\mathbf{x}_t\mathbf{x}_t']$ is the covariance matrix. The choice of r is based on the quantile of a desirable asymptotic probability distribution of normalized data points. In practice, the sample covariance matrix $\hat{\mathbf{\Gamma}}$ is used as an estimator of $\mathbf{\Gamma}$, and the data vector $\mathbf{x}_t$ is constructed based on the VAR model. From the definition (5), the quadratic form $h_{ii} = \mathbf{x}_t'\hat{\mathbf{\Gamma}}^{-1}\mathbf{x}_t$ is the unscaled statistical leverage score, $|\mathcal{S}|\ell_{ii}$, of the t-th data point for the VAR model, where $|\mathcal{S}|$ is the sample size.

The leverage score sampling can be summarized as, if for sample stretch $(\mathbf{x}_t, \mathbf{y}_t)$, the Mahalanobis distance satisfies $\sqrt{\mathbf{x}_t'\hat{\mathbf{\Gamma}}^{-1}\mathbf{x}_t} > r$, then we include t in subset $\mathcal{S}_{lev}$. As illustrated in Fig. 1, the normalized data points outside the ellipse are selected into $\mathcal{E}_{lev}$, where

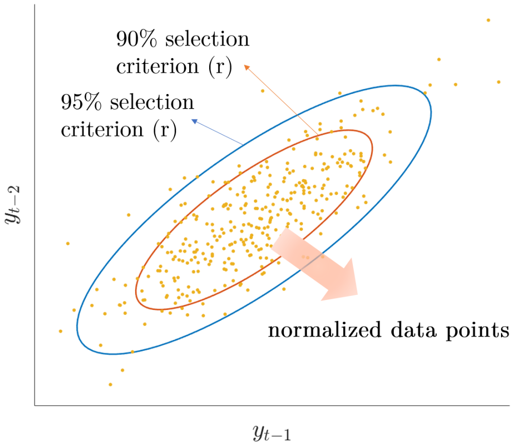


Figure 1: *Illustration of sampling criterion: One-dimensional AR(2) time series $\{y_t\}_{t\in\mathbb{Z}}$ are plotted with axes lag-2 values y_{t-2} vs. lag-1 values y_{t-1}. Sampling criterion r is the quantile of a desirable chi-squared sampling probability distribution. The normalized data points outside the ellipses (orange: 90-th percentile; blue: 95-th percentile) will be selected by the LSS.*

the normalization is based on their statistical leverage scores. The rate of sampling, $|\mathcal{S}_{lev}|/T$, is determined by the quantile r that measures the proportion of information selected rather than a prespecified sample size.

LSS simultaneously achieves the following goals

1. Improving the estimation efficiency of $\hat{\mathbf{B}}_{\mathcal{S}_{lev}}$ by reducing its estimation uncertainty;
2. Selecting a small set of samples to improve the computational efficiency;
3. Preserving the dependence structure since the data stretch $(\mathbf{x}_t, \mathbf{y}_t) = ((\mathbf{y}'_{t-1}, \mathbf{y}'_{t-2}, \cdots, \mathbf{y}'_{t-p})', \mathbf{y}_t)$ is the smallest sampling unit for any $t = 1, \ldots, T$;
4. Leading to streaming and decentralized algorithms.

3.1 Leverage Score Sampling for Streaming Time Series

The fundamental characteristic of a sampling method in the streaming setting, which distinguishes itself from its off-line version, is that streaming sampling requires a real-time decision-making mechanism. The LSS method utilizes a single-pass streaming procedure that calculates the leverage score in real-time so that one can make an immediate decision on sampling the current data point or not. In streaming setting, we replace the symmetric matrix $\hat{\mathbf{\Gamma}}_t^{-1}$ in h_{ii}, the inverse of the sample covariance matrix, by a robust estimation of the precision matrix $\mathbf{\Gamma}^{-1}$, denoted as Ω, see e.g. [9, 6]. Due to the stationarity of the VAR model, both $\hat{\mathbf{\Gamma}}_t^{-1}$ and Ω are consistent estimators for the model precision matrix $\mathbf{\Gamma}^{-1}$. Replacing $\hat{\mathbf{\Gamma}}_t^{-1}$ by Ω will no longer require the renormalization in the calculation of the leverage score, but also gain some statistical robustness and estimation efficiency [9]. For streaming time series, our LSS method thus requires a set of pilot samples to calculate the estimator Ω, which is a one-time operation. Subsequently, we can calculate the streaming leverage score and the corresponding sampling criterion

$$\tilde{h}_{tt} \triangleq \mathbf{x}'_t \Omega \mathbf{x}_t > r^2 \tag{7}$$

to select the important data point in real time, which is a single-pass procedure and only requires linear computation time with respect to the model dimension Kp.

Streaming time series also requires an online method to continuously aggregates past data, updating the current estimate of parameter to incorporate the information obtained from the new data. As the streaming data comes in sequentially, we would like to update the estimate of the parameter $\mathbf{B}$, sequentially as well. With a slight abuse of notation, we use $\mathbf{B}_t$ to denote the estimate of the parameter $\mathbf{B}$ using LSS method at time t. Hence, for each time point t in the selected subset $\mathcal{S}_{lev}$ up to current time T, we find the estimate $\mathbf{B}_t$ through optimizing the ℓ_2 loss,

$$\min_{\mathbf{B}_t} \sum_t ||\mathbf{y}'_t - \mathbf{x}'_t \mathbf{B}_t||_2^2, \quad \forall\, t \in \mathcal{S}_{lev} \tag{8}$$

which is in the form of dynamic linear model (DLM) [29, 37], where the observation vector at time t becomes, $\mathbf{y}_t = \mathbf{B}'_t \mathbf{x}_t + \mathbf{e}_t$ and the underlying state vector satisfies $\mathbb{E}\,\mathbf{B}_t = \mathbb{E}\,\mathbf{B}_{t-1}$.

There are plenty choices to solve the DLM in (8) , for example the classical Kalman filter. The Kalman filter [26, 22] updates the state vector vec($\mathbf{B}_t$) for $\forall\, t \in \mathcal{S}_{lev}$. The updates of the parameter vec($\mathbf{B}_t$) depends on accumulating the corresponding values themselves while streaming, and do not require accessing previous data points. It is important to note that, our LSS method is independent of the choice of the DLM solver in (8). The leverage score sampling for streaming time series can, therefore, run in constant memory and at a computational cost constant in time.

4 DECENTRALIZED LEVERAGE SCORE SAMPLING

When the multidimensional streams are observed in a decentralized environment, the LSS method can be

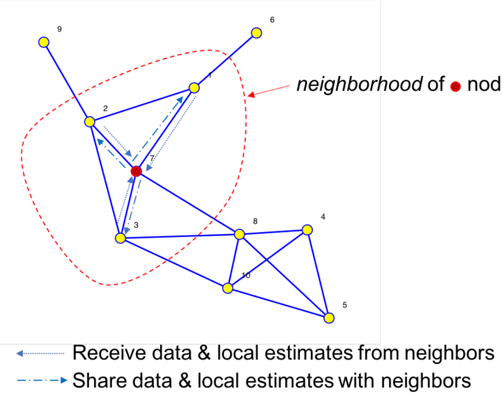

9
6
1
2
7
3
8
4
10
5
neighborhood of ● node
Receive data & local estimates from neighbors
Share data & local estimates with neighbors

Figure 2: *Diffusion strategy of the decentralized network. At every time t, node j collects a measurement $y_t^{(j)}$ and neighborhood data.*

efficiently applied in parallel into asynchronous decentralized optimization algorithm by exploiting to VAR model structure. The leverage score and sampling criterion defined in (7) can be computed on each dimension in parallel and asynchronously under the decentralized setting.

The decentralized architecture is needed as long as the streams dimension K is large or distributed physically apart in a network that accessing the data streams on a single machine is impossible. More specifically, a decentralized system lacks a fusion center (a centralized computing node) and may be communication-restricted, which requires a communication-efficient information diffusion strategy in the design of the decentralized algorithm. As illustrated in Fig. 2, we use neighborhood-based communication strategy in our sampling method and parameter estimation.

Note that the problem of (8) can be decomposed into K subproblems by taking advantage of the VAR model structure. We assume that, without loss of generality, each node in the network observes one dimension data of the multidimensional streams. The selection criterion $\mathcal{S}_{lev}^{(j)}$ for node j becomes,

$$\tilde{h}_{tt}^{(j)} = \mathbf{x}'_{\tau_j}\Omega\mathbf{x}_{\tau_j} > r^2,$$

as long as the node j receives its local copy of data $\mathbf{x}_{\tau_j}$ at local time τ_j. We express the parameter matrix $\mathbf{B}$ as a block matrix with column vectors

$$\mathbf{B} = [\boldsymbol{\beta}^{(1)}, \boldsymbol{\beta}^{(2)}, \dots, \boldsymbol{\beta}^{(K)}]$$

with $\boldsymbol{\beta}^{(j)}$ being the jth column of $\mathbf{B}$ for $j = 1, \dots, K$. Hence for node j with its local time τ_j, the j-th subproblem is

$$\min_{\boldsymbol{\beta}_{\tau_j}^{(j)}} \sum_{\tau_j} ||y_{\tau_j}^{(j)} - \mathbf{x}'_{\tau_j}\boldsymbol{\beta}_{\tau_j}^{(j)}||_2^2, \quad \forall \tau_j \in \mathcal{S}_{lev}^{(j)}, \tag{9}$$

where $y_{\tau_j}^{(j)}$ is the jth element of $\mathbf{y}_{\tau_j}$, $\boldsymbol{\beta}_{\tau_j}^{(j)}$ is the estimate of $\boldsymbol{\beta}^{(j)}$ at time τ_j, for $j = 1, \dots, K$. Those $\boldsymbol{\beta}^{(j)}$ can be estimated at each corresponding node locally as soon as $\mathbf{x}_{\tau_j}$ is completed at local time τ_j. From (9), we see that the sampling, parameter estimation and communication of nodes are uncoordinated. Each node j has its own local time τ_j and a global clock is not needed, resulting in an asynchronous algorithm. The algorithm then is running over the *ad hoc* network topology [48], i.e. decentralized network without fusion/data center to aggregate data asynchronously, where the nodes communicate with their neighbors and perform the local computation. The data from neighbors arrived sequentially with delay depends on the distance in the network due to the limited communication, see Fig. 2.

Algorithm 1 Online Asynchronous Decentralized Leverage Score Sampling

```
Require: Precision matrix $\Omega$, quantile $r$.
    Broadcast initial value of parameter $\mathbf{B}_0$ and covariance $\mathbf{P}_0$.
while $t > 0$ do
    while node $j \in [1, \dots, K]$ in parallel do
        Receive the local data $y_t^{(j)}$ without delay, and the neighborhood data with arbitrary delay
        Send out the local data $y_t^{(j)}$ to neighbors
        Wait until $\mathbf{x}_{\tau_j}$ is complete for some $\tau_j \leq t$
        if $\tilde{h}_{tt}^{(j)} = \mathbf{x}'_{\tau_j}\Omega\mathbf{x}_{\tau_j} > r^2$ then        ▷ LSS
            Update $\boldsymbol{\beta}_{\tau_j}^{(j)}$ and $\mathbf{P}_{\tau_j}$ according to the local Kalman filter (10) and (11)
        else
            $\boldsymbol{\beta}_{\tau_j}^{(j)} = \beta_{\tau_j - 1}^{(j)}$ and $P_{\tau_j} = P_{\tau_j - 1}$
        end if
        Transmit the local estimate $\boldsymbol{\beta}^{(j)}$ to neighbors and receive neighbors' estimation
        Set $\tau_j \leftarrow \tau_j + 1$
        return $\mathbf{B}_{\tau_j} = [\boldsymbol{\beta}_{\tau_j}^{(1)}, \dots, \boldsymbol{\beta}_{\tau_j}^{(j)}, \dots, \boldsymbol{\beta}_{\tau_j}^{(K)}]$
    end while nodes
end while $t$
```

Solving (9) can be done by various decentralized consensus optimization, e.g. decentralized gradient descent [52], decentralized ADMM [41], decentralized Kalman filter [35, 7] and references therein, etc. We use diffusion strategies in [7] as an illustration to handle the parameter estimation and sampling, which allow asynchrony and delay in the decentralized consensus optimization. We use the local Kalman filter to estimate the local parameter $\boldsymbol{\beta}_{\tau_j}^{(j)}$ for j-th node and $\tau_j \in \mathcal{S}_{lev}$

$$\mathbf{P}_{\tau_j} = \mathbf{P}_{\tau_j - 1} - \mathbf{k}_{\tau_j}\mathbf{x}'_{\tau_j}\mathbf{P}_{\tau_j - 1} \tag{10}$$

$$\boldsymbol{\beta}_{\tau_j}^{(j)} = \boldsymbol{\beta}_{\tau_j - 1}^{(j)} + [y_{\tau_j}^{(j)} - \mathbf{x}'_{\tau_j}\boldsymbol{\beta}_{\tau_j - 1}^{(j)}]\mathbf{k}_{\tau_j}, \tag{11}$$

where $\mathbf{k}_{\tau_j} \triangleq \gamma_{\tau_j}^{-1}\mathbf{P}_{\tau_j - 1}\mathbf{x}_{\tau_j}$, and $\gamma_{\tau_j} \triangleq 1 + \mathbf{x}'_{\tau_j}\mathbf{P}_{\tau_j - 1}\mathbf{x}_{\tau_j}$ with $\mathbf{P}_{\tau_j}$ as the j-th local estimate of the precision matrix at local time τ_j. After getting the local estimate $\boldsymbol{\beta}^{(j)}_{\tau_j}$, the node exchanges the local estimate with neighbors to form a complete estimate of $\mathbf{B}_{\tau_j}$ at time τ_j. The theoretical guarantee of the consensus result of the algorithm can be found in [7]. The algorithm is summarized in Algorithm 1.

5 THEORETICAL JUSTIFICATION OF LEVERAGE SCORE SAMPLING

The goal of this section is to provide the theoretical justification on the superiority of the LSS method over the Bernoulli sampling method. In Bernoulli sampling, we take the simple random sampling over time, i.e., conduct Bernoulli trail to with success probability q at each time t to select samples.

The following theorem[1] establishes the asymptotic normality of the estimate $\hat{\mathbf{B}}_{\mathcal{S}_{lev}}$ based on the LSS samples $\mathcal{S} = \{1 \le t \le T: \ \mathbf{x}_t \in \mathcal{E}\}$.

Theorem 5.1. *Let $m = Kp$ and let $K \times K$ matrix $\Psi = \mathbb{E}[\mathbf{e}_t\mathbf{e}_t']$. Define the $m \times m$ matrix*

$$\Gamma(\mathcal{E}) = \mathbb{E}\left[1_{\{\mathbf{x}_t \in \mathcal{E}\}}\mathbf{x}_t\mathbf{x}_t'\right].$$

and suppose that it is non-singular. Then as $T \to \infty$,

$$\sqrt{T}(\text{vec}(\hat{\mathbf{B}}_{\mathcal{S}}) - \text{vec}(\mathbf{B})) \xrightarrow{d} N(\mathbf{0}, \Psi \otimes \Gamma(\mathcal{E})^{-1}). \quad (12)$$

In view of Theorem 5.1, the asymptotic covariance matrix of $\text{vec}(\hat{\mathbf{B}}_{\mathcal{S}})$ dropping the scaling T^{-1} is

$$\Psi \otimes \Gamma(\mathcal{E})^{-1}. \quad (13)$$

Our goal is to compare this covariance matrix with those arising from some naive sampling approaches. One option is to directly use a consecutive sample $(\mathbf{x}_t, \mathbf{y}_t)_{1 \le t \le Tq}$, $q \in (0, 1)$. Another option is to employ an i.i.d. Bernoulli sampling: for each $t \in \{1 \ldots, T\}$, the sample $(\mathbf{x}_t, \mathbf{y}_t)$ is selected for regression with probability q independently. It turns out that these two options lead to the same asymptotic covariance matrix:

Theorem 5.2. *Under either the consecutive sampling or the i.i.d. Bernoulli sampling described above, we have as $T \to \infty$,*

$$\sqrt{T}(\text{vec}(\hat{\mathbf{B}}) - \text{vec}(\mathbf{B})) \xrightarrow{d} N(\mathbf{0}, q^{-1}\Psi \otimes \Gamma^{-1}), \quad (14)$$

where

$$\Gamma = \mathbb{E}[\mathbf{x}_t\mathbf{x}_t'] = \Gamma(\mathbb{R}^m). \quad (15)$$

To have a fair comparison with a leveraged-based sampling approach, we shall set

$$q = Q(\mathcal{E}) = \Pr(\mathbf{X}_t \in \mathcal{E}).$$

This ensures that the average sampling proportions across the different approaches are the same. Now the asymptotic covariance matrix (dropping n^{-1}) of the consecutive or Bernoulli sampling approaches is

$$Q(\mathcal{E})^{-1}\Psi \otimes \Gamma^{-1}. \quad (16)$$

Comparing (13) with (16), we want the selection of $\mathcal{E}$ to achieve

$$\Psi \otimes \Gamma(\mathcal{E})^{-1} \prec Q(\mathcal{E})^{-1}\Psi \otimes \Gamma^{-1}. \quad (17)$$

Relation (17) is equivalent to

$$Q(\mathcal{E})\Gamma \prec \Gamma(\mathcal{E}). \quad (18)$$

See items 10.51(b) and 11.1(i) of [40].

Under the Gaussian assumption and a choice of $\mathcal{E}$, the following theorem provides an expression for the minimum eigenvalue of $Q(\mathcal{E})\Gamma - \Gamma(\mathcal{E})$, and thus yields a criterion for (18) or equivalently (17) to hold.

Theorem 5.3. *Suppose in (2) that $\mathbf{e}_t$'s are i.i.d. $N(\mathbf{0}, \Psi)$. Let $m = Kp$ and let $\Gamma^{1/2} = P'\Lambda^{1/2}P$ be a square root of the covariance matrix Γ in (15), where P is an orthogonal matrix and $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_m)$ is a diagonal matrix of the eigenvalues of Γ. Let $\mathcal{D}_r = \{\mathbf{x} \in \mathbb{R}^m : \|\mathbf{x}\| > r\}$, $r \ge 0$. Define $\mathcal{E}_{lev}$ as the complement of an ellipsoid:*

$$\mathcal{E}_{lev}(r) \triangleq \mathcal{E}_r = \Gamma^{1/2}\mathcal{D}_r.$$

(a) The sampling probability is

$$Q(\mathcal{E}_r) = Q(m, r) = \Pr(\chi^2_m > r^2), \quad (19)$$

where χ^2_m denotes a chi-squared random variable with m degrees of freedom.

(b) The minimum eigenvalue of $Q(\mathcal{E}_r)\Gamma - \Gamma(\mathcal{E}_r)$ is

$$\lambda_{\min}\left[T(m, r) - Q(m, r)\right],$$

where $\lambda_{\min} = \min(\lambda_1, \ldots, \lambda_m) > 0$ and

$$T(m, r) = \frac{1}{m}\,\mathbb{E}[\chi^2_m 1_{\{\chi^2_m > r^2\}}].$$

Corollary 5.4. *Under the setup of Theorem 5.3, the relation (17), namely, the asymptotic superiority of leverage score sampling using $\mathcal{E}_{lev}(r) \triangleq \mathcal{E}_r$ over the Bernoulli sampling holds, if*

$$T(m, r) > Q(m, r).$$

Remark 1. One can show that $T(m, r) > Q(m, r)$ holds if

$$r^2 > m. \quad (20)$$

This criterion is distribution-free. If $\mathbf{e}_t$ is non-Gaussian, then some symmetry explored in the proof of Theorem 5.3 is unavailable. Nevertheless, (17) is expected hold under (20) with a moderate departure from normality.

[1] The proofs of all theorems can be found in Supplementary Material.

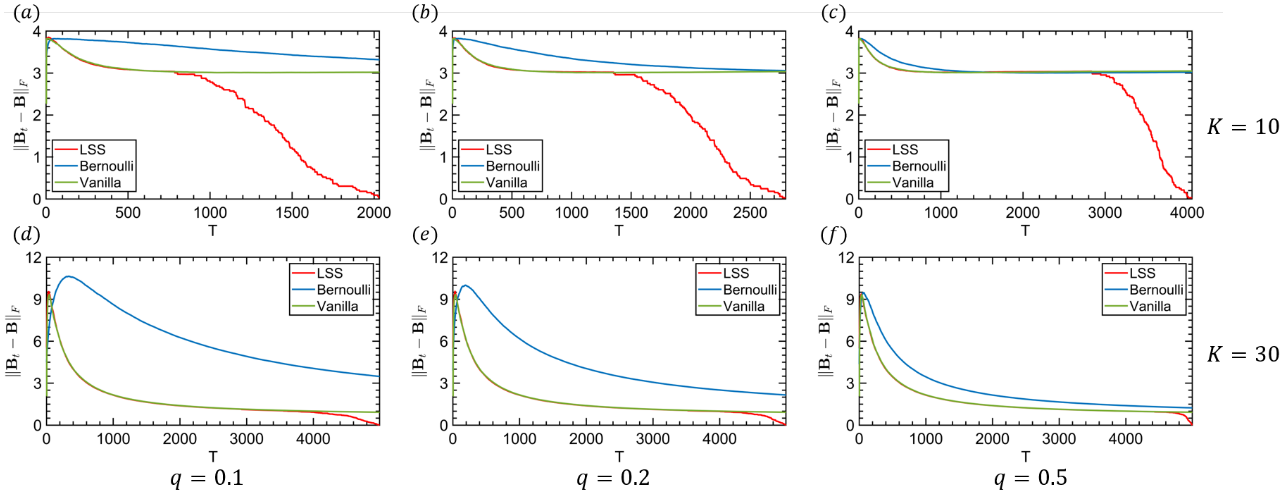


Figure 3: *Each column shows the comparison of estimation error with different sampling rate (a): $q = 0.1$, (b): $q = 0.2$, and (c): $q = 0.5$. Fig.(a)-(c) show the results with a 10-dimensional stationary VAR(3) process and Fig.(d)-(f) show the results with a 30-dimensional stationary VAR(1) process. The estimation error, $||\mathbf{B_t} - \mathbf{B}||_F$ of LSS (red), Bernoulli (blue) and Vanilla (green) methods are plotted against time T with total time steps 5000.*

6 EXPERIMENTS

In this section, we demonstrate the applicability of the LSS method on three experiments: the synthetic data with various settings and two real multidimensional streaming data. In all experiments, we compare our proposed LSS method against Bernoulli sampling method (hereafter, Bernoulli) and vanilla Kalman filter method (hereafter, Vanilla) [7] in a decentralized setting with fixed network topology structure. The Vanilla method uses full observed data, while the LSS and Bernoulli methods take samples accordingly with the same sampling rate q. We assume that the current node can only access its own data in real time, and the data transition from other nodes is delayed by the distance in the network connectivity to the current node. The results show the distinguishing features of our LSS method: accurate in parameter estimation with faster and better convergence at different sampling rate q, and computation efficiency with shorter execution time.

6.1 Synthetic Data

To compare accuracy and efficiency of parameter estimation in the streaming setting at different sampling rates q, dimensions K and lags p, we perform simulation study on synthetic data and report the estimation error $||\mathbf{B}_t - \mathbf{B}||_F$. The simulation data is generated by two settings: the first one (used in Fig. 3 (a)-(c)) is a 10-dimensional stationary VAR(3) process for 10 nodes, i.e., $K = 10,\ p = 3$ and the second one (used in Fig. 3 (d)-(f)) is a 30-dimensional stationary VAR(1) process for 30 nodes, i.e., $K = 30,\ p = 1$. The topology structure and the connectivity of nodes is created randomly at the beginning of the simulation and then applied to all methods [39]. The first 200 data points from all nodes are used as pilot samples to obtain the estimate of Ω for each setting. In each subplot of Fig. 3, the result is compared by the estimation error, $||\mathbf{B}_t - \mathbf{B}||_F$, against time T, with 100 independent replicates, on different sampling rate $q \in [0.1, 0.2, 0.5]$ and two settings of K and p.

Fig. 3 shows that our method converges significantly faster (high accuracy and efficiency) than Bernoulli method, and converge as fast or slightly faster than the Vanilla method that uses full data points in all test cases. In addition, LSS takes fewer computational steps (require fewer samples) than the Bernoulli method to achieve convergence. Fig. 5(d) shows the average elapsed time of 100 trials of the three methods. It can be seen that the time consumption of LSS is much smaller than vanilla Kalman filter and similar to the Bernoulli, while LSS achieves better estimation results than both of the other two methods, especially comparing to Bernoulli sampling. From Fig. 3(a) and (c), the advantages of the LSS method are more obvious when the sample size is small.

6.2 Real Data

The LSS, Bernoulli and Vanilla methods are implemented on two real datasets to compare the prediction error, $||\mathbf{y}_t - \hat{\mathbf{y}}_t||_2$, since the VAR model parameters are unknown for real data. In both experiments, the first 2000 data points are used as pilot samples.

Seismic Data: We consider the seismic data that records the wave amplitude (mm/s) from earthquake sequences in Oklahoma collected on October 26, 2014

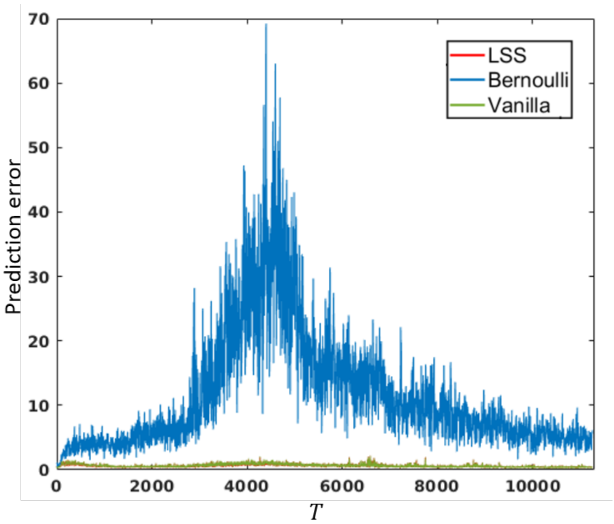
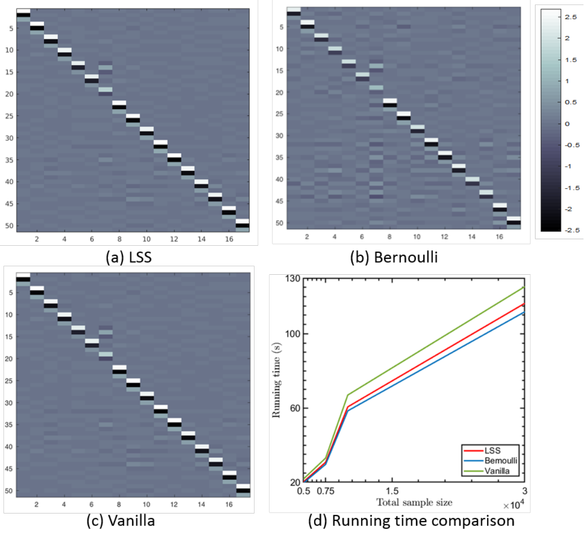


Figure 4: *Prediction error from seismic data. The LSS (red) and Vanilla (green) error are tangled together in bottom of the plot.*

[8]. The data contains 17 sensors with 17,698 time-steps. The VAR(3) model was chosen based on the analysis of the pilot sample. Fig. 4 shows the prediction error of the seismic data estimation. LSS outperforms the Bernoulli method, and it can achieve comparable or better prediction than the Vanilla method. From the first-order parameter matrices $\mathbf{\Phi}_1$ shown in Fig. 5, we see that LSS (a) and vanilla Kalman filter (c) perform similar estimation, while Bernoulli method has several off-diagonal unusual patterns. Combine Fig. 5 and Fig. 4, we see that Bernoulli method failed to capture the correlation information in the seismogram so that the prediction reflects a severer bias and delay.



(a) LSS

(b) Bernoulli

(c) Vanilla

(d) Running time comparison

Figure 5: *Seismic Data: Fig.(a)-(c) show first-order estimated parameter matrices* $\mathbf{\Phi}_1$ *at time* $t = 8500$. *Fig.(d) is the average elapsed time (seconds) of LSS(red), Bernoulli(blue) and Vanilla(green) methods over 100 replicates.*

Gas Sensor Array: We do another experiment on the UCI dynamic gas mixtures dataset [19, 20]. The data uses 16 chemical sensors at a sampling frequency of 100 Hz and records 4, 208, 261 time-steps of Ethylene and CO mixture in air. For our experiments, we use data from 15 sensors to build a VAR(3) model[2]. A snapshot of the prediction error is shown in Fig. 6. It is clear that LSS captures the correct patterns in streams and performed superior or comparable to the Vanilla method that using the full data points.

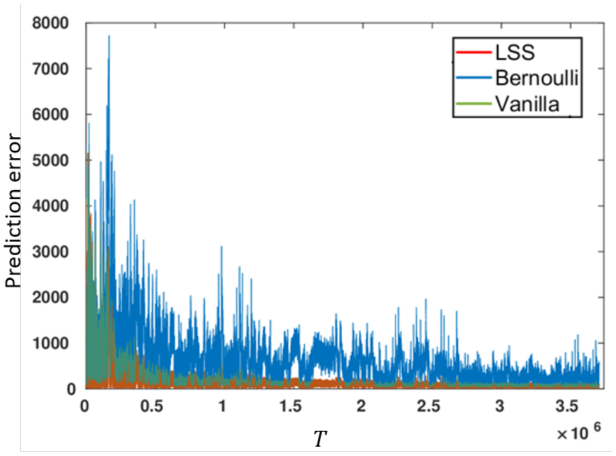


Figure 6: *Prediction error from gas sensor data.*

7 CONCLUSION

We develop a novel online leverage score sampling method for efficiently estimating the temporal dependence of streaming multidimensional time series in an asynchronous decentralized environment. We prove that leverage score sampling yields a lower parameter estimation variance by selecting informative samples in infinite-sample streaming time series. Our future work includes, from the theoretical perspective, finding an optimal selection criterion under a more general (such as nonlinear or nonparametric) dynamic streaming model, and from the application perspective, extending the sampling scope to irregular-sampled high-dimensional random field streams, such as medical imaging real-time diagnosis, video and audio summarization, and environmental monitoring.

References

[1] D. Achlioptas, F. McSherry, and B. Schölkopf. Sampling techniques for kernel methods. In *Advances in neural information processing systems*, pages 335–342, 2002.

[2] A. Alaoui and M. W. Mahoney. Fast randomized kernel ridge regression with statistical guarantees.

[2] We drop data from one sensor due to incomplete observation and low quality of the data.

In *Advances in Neural Information Processing Systems*, pages 775–783, 2015.

[3] C. Boutsidis, P. Drineas, and M. Magdon-Ismail. Near-optimal column-based matrix reconstruction. *SIAM Journal on Computing*, 43(2):687–717, 2014.

[4] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[5] P. J. Brockwell and R. A. Davis. *Time series: theory and methods*. Springer Science & Business Media, 2013.

[6] T. T. Cai, Z. Ren, H. H. Zhou, et al. Estimating structured high-dimensional covariance and precision matrices: Optimal rates and adaptive estimation. *Electronic Journal of Statistics*, 10(1):1–59, 2016.

[7] F. S. Cattivelli and A. H. Sayed. Diffusion strategies for distributed kalman filtering and smoothing. *IEEE Transactions on automatic control*, 55(9):2069–2084, 2010.

[8] X. Chen, R. E. Abercrombie, C. Pennington, X. Meng, and Z. Peng. Source parameter validations using multiple-scale approaches for earthquake sequences in Oklahoma: implications for earthquake triggering processes. In *2016 AGU Fall Meeting, San Francisco, California*, 2016.

[9] X. Chen, M. Xu, W. B. Wu, et al. Covariance and precision matrix estimation for high-dimensional time series. *The Annals of Statistics*, 41(6):2994–3021, 2013.

[10] R. Chitnis, G. Cormode, H. Esfandiari, M. Hajiaghayi, A. McGregor, M. Monemizadeh, and S. Vorotnikova. Kernelization via sampling with applications to finding matchings and related problems in dynamic graph streams. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 1326–1344. Society for Industrial and Applied Mathematics, 2016.

[11] M. B. Cohen, C. Musco, and C. Musco. Input sparsity time low-rank approximation via ridge leverage score sampling. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1758–1777. SIAM, 2017.

[12] G. Cormode, S. Muthukrishnan, K. Yi, and Q. Zhang. Continuous sampling from distributed streams. *Journal of the ACM (JACM)*, 59(2):10, 2012.

[13] M. Derezinski and M. K. Warmuth. Unbiased estimates for linear regression via volume sampling. In *Advances in Neural Information Processing Systems*, pages 3087–3096, 2017.

[14] P. Drineas, M. Magdon-Ismail, M. W. Mahoney, and D. P. Woodruff. Fast approximation of matrix coherence and statistical leverage. *Journal of Machine Learning Research*, 13(Dec):3475–3506, 2012.

[15] P. Drineas, M. W. Mahoney, and S. Muthukrishnan. Relative-error cur matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30(2):844–881, 2008.

[16] P. S. Efraimidis. Weighted random sampling over data streams. In *Algorithms, Probability, Networks, and Games*, pages 183–195. Springer, 2015.

[17] E. Elhamifar and M. Kaluza. Online summarization via submodular and convex optimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[18] E. Elhamifar and M. C. D. P. Kaluza. Subset selection and summarization in sequential data. In *Advances in Neural Information Processing Systems*, pages 1036–1045, 2017.

[19] J. Fonollosa and R. Huerta. Gas sensor array under dynamic gas mixtures data set, 2015.

[20] J. Fonollosa, S. Sheik, R. Huerta, and S. Marco. Reservoir computing compensates slow response of chemosensor arrays exposed to fast varying gas concentrations in continuous monitoring. *Sensors and Actuators B: Chemical*, 215:618–629, 2015.

[21] B. Gong, W.-L. Chao, K. Grauman, and F. Sha. Diverse sequential subset selection for supervised video summarization. In *Advances in Neural Information Processing Systems*, pages 2069–2077, 2014.

[22] M. S. Grewal. Kalman filtering. In *International Encyclopedia of Statistical Science*, pages 705–708. Springer, 2011.

[23] J. Hamilton. *Time Series Analysis*. Princeton University Press, 1994.

[24] A. C. Harvey. *Forecasting, structural time series models and the Kalman filter*. Cambridge university press, 1990.

[25] R. Jörnsten and B. Yu. Simultaneous gene clustering and subset selection for sample classification via MDL. *Bioinformatics*, 19(9):1100–1109, 2003.

[26] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[27] M. Kapralov, Y. T. Lee, C. Musco, C. Musco, and A. Sidford. Single pass spectral sparsification in dynamic streams. *SIAM Journal on Computing*, 46(1):456–477, 2017.

[28] A. Kulesza, B. Taskar, et al. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.

[29] T. L. Lai and C. Z. Wei. Least squares estimates in stochastic regression models with applications to identification and control of dynamic systems. *The Annals of Statistics*, pages 154–166, 1982.

[30] H. Lin and J. Bilmes. Learning mixtures of submodular shells with application to document summarization. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, pages 479–490. AUAI Press, 2012.

[31] P. Ma, M. W. Mahoney, and B. Yu. A statistical perspective on algorithmic leveraging. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

[32] P. Ma, M. W. Mahoney, and B. Yu. A statistical perspective on algorithmic leveraging. *The Journal of Machine Learning Research*, 16(1):861–911, 2015.

[33] M. W. Mahoney and P. Drineas. Cur matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106(3):697–702, 2009.

[34] C. Musco and C. Musco. Recursive sampling for the nystrom method. In *Advances in Neural Information Processing Systems*, pages 3836–3848, 2017.

[35] R. Olfati-Saber. Distributed kalman filter with embedded consensus filters. In *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*, pages 8179–8184. IEEE, 2005.

[36] D. Papailiopoulos, A. Kyrillidis, and C. Boutsidis. Provable deterministic leverage score sampling. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 997–1006. ACM, 2014.

[37] A. Pole, M. West, and J. Harrison. *Applied Bayesian forecasting and time series analysis*. Chapman and Hall/CRC, 1994.

[38] G. Raskutti and M. Mahoney. Statistical and algorithmic perspectives on randomized sketching for ordinary least-squares. In *International Conference on Machine Learning*, pages 617–625, 2015.

[39] A. H. Sayed et al. Adaptation, learning, and optimization over networks. *Foundations and Trends® in Machine Learning*, 7(4-5):311–801, 2014.

[40] G. A. Seber. *A matrix handbook for statisticians*, volume 15. John Wiley & Sons, 2008.

[41] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin. On the linear convergence of the admm in decentralized consensus optimization.

[42] I. Simon, N. Snavely, and S. M. Seitz. Scene summarization for online image collections. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[43] Z. Song, L. F. Yang, and P. Zhong. Sensitivity sampling over dynamic geometric data streams with applications to k-clustering. *arXiv preprint arXiv:1802.00459*, 2018.

[44] D. A. Spielman and N. Srivastava. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6):1913–1926, 2011.

[45] R. S. Tsay. *Multivariate time series analysis: with R and financial applications*. John Wiley & Sons, 2013.

[46] H. Wang, R. Zhu, and P. Ma. Optimal subsampling for large sample logistic regression. *Journal of the American Statistical Association*, (just-accepted), 2017.

[47] M. West, P. J. Harrison, and H. S. Migon. Dynamic generalized linear models and bayesian forecasting. *Journal of the American Statistical Association*, 80(389):73–83, 1985.

[48] J. Wu and I. Stojmenovic. Ad hoc networks. *Computer*, 37(2):29–31, 2004.

[49] P.-Y. Wu and M. D. Wang. The selection of quantification pipelines for illumina rna-seq data using a subsampling approach. In *Biomedical and Health Informatics (BHI), 2016 IEEE-EMBS International Conference on*, pages 78–81. IEEE, 2016.

[50] T. Wu, K. Yuan, Q. Ling, W. Yin, and A. H. Sayed. Decentralized consensus optimization with asynchrony and delays. *IEEE Transactions on Signal and Information Processing over Networks*, 4(2):293–307, 2018.

[51] F. Xiao and L. Wang. Asynchronous consensus in continuous-time multi-agent systems with switching topology and time-varying delays. *IEEE Transactions on Automatic Control*, 53(8):1804–1816, 2008.

[52] K. Yuan, Q. Ling, and W. Yin. On the convergence of decentralized gradient descent. *SIAM Journal on Optimization*, 26(3):1835–1854, 2016.

[53] K. Zhang, C. Liu, J. Zhang, H. Xiong, E. Xing, and J. Ye. Randomization or condensation?: Linear-cost matrix sketching via cascaded compression sampling. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 615–623. ACM, 2017.