Submitted 3/20; Revised 10/21; Published 1/22

# LSAR: Efficient Leverage Score Sampling Algorithm for the Analysis of Big Time Series Data

**Ali Eshragh** ALI.ESHRAGH@NEWCASTLE.EDU.AU
*School of Information and Physical Sciences, University of Newcastle, Australia*
*International Computer Science Institute, Berkeley, CA, USA*

**Fred Roosta** FRED.ROOSTA@UQ.EDU.AU
*School of Mathematics and Physics, University of Queensland, Australia*
*International Computer Science Institute, Berkeley, CA, USA*

**Asef Nazari** ASEF.NAZARI@DEAKIN.EDU.AU
*School of Information Technology, Deakin University, Australia*

**Michael W. Mahoney** MMAHONEY@STAT.BERKELEY.EDU
*Department of Statistics, University of California at Berkeley, USA*
*International Computer Science Institute, Berkeley, CA, USA*

**Editor:** Garvesh Raskutti

## Abstract

We apply methods from randomized numerical linear algebra (RandNLA) to develop improved algorithms for the analysis of large-scale time series data. We first develop a new fast algorithm to estimate the leverage scores of an autoregressive (AR) model in big data regimes. We show that the accuracy of approximations lies within $(1 + \mathcal{O}(\varepsilon))$ of the true leverage scores with high probability. These theoretical results are subsequently exploited to develop an efficient algorithm, called LSAR, for fitting an appropriate AR model to big time series data. Our proposed algorithm is guaranteed, with high probability, to find the maximum likelihood estimates of the parameters of the underlying true AR model and has a worst case running time that significantly improves those of the state-of-the-art alternatives in big data regimes. Empirical results on large-scale synthetic as well as real data highly support the theoretical results and reveal the efficacy of this new approach.

**Keywords:** autoregressive model, maximum likelihood estimation, big data regime, randomized numerical linear algebra, sampling

## 1. Introduction

A *time series* is a collection of random variables indexed according to the order in which they are observed in time. The main objective of *time series analysis* is to develop a statistical model to forecast the future behavior of the system. At a high level, the main approaches for this include the ones based on considering the data in its original *time domain* and those arising from analyzing the data in the corresponding *frequency domain* (Shumway and Stoffer 2017, Chapter 1). More specifically, the former approach focuses on modeling some future value of a time series as a parametric function of the current and past values by studying the correlation between adjacent points in time. The latter framework, however, assumes the primary characteristics of interest in time series analysis relate to periodic or systematic sinusoidal variations. Although the two approaches may produce similar outcomes for many cases, the comparative performance is better done in the "time domain" (Shumway and Stoffer, 2017, Chapter 1) which is the main focus of this paper.

Box and Jenkins (1976) introduced their celebrated *autoregressive moving average* (ARMA) model for analyzing stationary time series. Although it has been more than 40 years since this model was developed, due to its simplicity and vast practicality, it continues to be widely used in theory and practice. A special case of an ARMA model is an *autoregressive* (AR) model, which merely includes the autoregressive component. Despite their simplicity, AR models have a wide range of applications spanning from genetics and medical sciences to finance and engineering (Hamilton, 1989; Anderson et al., 1998; Chakravarthy et al., 2004; Shen and Lu, 2018; Abolghasemi et al., 2020; Eshragh et al., 2021; Messner and Pinson, 2019).

The main hyper-parameter of an AR model is its *order*, which directly relates to the dimension of the underlying predictor variable. In other words, the order of an AR model amounts to the number of lagged values that are included in the model. In problems involving big time series data, selecting an appropriate order for an AR model amounts to computing the solutions of many potentially large scale *ordinary least squares* (OLS) problems, which can be the main bottleneck of computations (cf. Section 2.1). Here is where randomized sub-sampling algorithms can be used to greatly speed-up such model selection procedures.

For computations involving large matrices in general, and large-scale OLS problems in particular, randomized numerical linear algebra (RandNLA) has successfully employed various random sub-sampling and sketching strategies. There, the underlying data matrix is randomly, yet appropriately, "compressed" into a smaller one, while approximately retaining many of its original properties. As a result, much of the expensive computations can be performed on the smaller matrix; Mahoney (2011) and Woodruff (2014) provided an extensive overview of RandNLA subroutines and their many applications. Moreover, implementations of algorithms based on those ideas have been shown to beat state-of-the-art numerical routines (Avron et al. 2010; Meng et al. 2014; Yang et al. 2016).

Despite their simplicity and efficient constructions, matrix approximations using *uniform* sampling strategies are highly ineffective in the presence of non-uniformity in the data (e.g., outliers). In such situations, *non-uniform* (but still i.i.d.) sampling schemes in general, and leverage score sampling in particular (Drineas et al. 2012), are instrumental not only in obtaining the strongest worst case theoretical guarantees, but also in devising high-quality numerical implementations. In times series data, one might expect that sampling methods based on leverage scores can be highly effective (cf. Figure 8). However, the main challenge lies in computing the leverage scores, which naïvely can be as costly as the solution of the original OLS problems. In this light, exploiting the structure of the time series model for estimating the leverage scores can be the determining factor in obtaining efficient algorithms for time series analysis. We carry out that here in the context of AR models. In particular, our contributions can be summarized as follows:

(i) We introduce RandNLA techniques to the analysis of big time series data.

(ii) By exploiting the available structure, we propose an algorithm for approximating the leverage scores of the underlying data matrix that is shown to be faster than the state-of-the-art alternatives.

(iii) We theoretically obtain a high-probability relative error bound on the leverage score approximations.

(iv) Using these approximations, we then develop a highly-efficient algorithm, called `LSAR`, for fitting `AR` models with provable guarantees.

(v) We empirically demonstrate the effectiveness of the `LSAR` algorithm on several large-scale synthetic as well as real big time series data.

The structure of this paper is as follows: Section 2 introduces `AR` models and RandNLA techniques in approximately solving large-scale OLS problems. Section 3 deals with the theoretical results on developing an efficient leverage score sampling algorithm to fit and estimate the parameters of an `AR` model. All proofs are presented in Appendix A. Section 4 illustrates the efficacy of the new approach by implementing it on several large-scale synthetic as well as real big time series data. Section 5 concludes the paper and addresses future work.

### Notation

Throughout the paper, vectors and matrices are denoted by bold lower-case and bold upper-case letters, respectively (e.g., $\boldsymbol{v}$ and $\boldsymbol{V}$). All vectors are assume to be column vectors. We use regular lower-case to denote scalar constants (e.g., $d$). Random variables are denoted by regular upper-case letters (e.g., $Y$). For a real vector, $\boldsymbol{v}$, its transpose is denoted by $\boldsymbol{v}^{\intercal}$. For two vectors $\boldsymbol{v}, \boldsymbol{w}$, their inner-product is denoted as $\langle \boldsymbol{v}, \boldsymbol{w} \rangle = \boldsymbol{v}^{\intercal}\boldsymbol{w}$. For a vector $\boldsymbol{v}$ and a matrix $\boldsymbol{V}$, $\|\boldsymbol{v}\|$ and $\|\boldsymbol{V}\|$ denote vector $\ell_2$ norm and matrix spectral norm, respectively. The condition number of a matrix $\boldsymbol{A}$, which is the ratio of its largest and smallest singular values, is denoted by $\kappa(\boldsymbol{A})$. Range of a matrix $\boldsymbol{A} \in \mathbb{R}^{n \times d}$, denoted by Range($\boldsymbol{A}$), is a subspace of $\mathbb{R}^n$, consisting all the vectors $\left\{ \boldsymbol{A}\boldsymbol{x} \mid \boldsymbol{x} \in \mathbb{R}^d \right\}$. Adopting `Matlab` notation, we use $\boldsymbol{A}(i, :)$ to refer to the $i^{\text{th}}$ row of the matrix $\boldsymbol{A}$ and consider it as a column vector. Finally, $\boldsymbol{e}_i$ denotes a vector whose $i^{\text{th}}$ component is one, and zero elsewhere.

## 2. Background

In this section, we present a brief overview of the two main ingredients of the results of this paper, namely autoregressive models (Section 2.1) and leverage score sampling for OLS problems (Section 2.2).

### 2.1 Autoregressive Models

A time series $\{Y_t;\, t = 0, \pm 1, \pm 2, \ldots\}$ is called (weakly) stationary, if the mean $\mathbb{E}[Y_t]$ is independent of time $t$, and the auto-covariance $Cov(Y_t, Y_{t+h})$ depends only on the lag $h$ for any integer values $t$ and $h$. A stationary time series $\{Y_t;\, t = 0, \pm 1, \pm 2, \ldots\}$[1] with the

[1] Throughout this paper, we assume that $Y_t$'s are continuous random variables.

constant mean $\mathbb{E}[Y_t] = 0$ is an AR model with the order $p$, denoted by AR($p$), if we have

$$Y_t = \phi_1 Y_{t-1} + \cdots + \phi_p Y_{t-p} + W_t,$$

where $\phi_p \neq 0$ and the time series $\{W_t;\, t = 0, \pm 1, \pm 2, \ldots\}$ is a Gaussian white noise with the mean $\mathbb{E}[W_t] = 0$ and variance $Var(W_t) = \sigma_W^2$. Recall that a Gaussian white noise is a stationary time series in which each individual random variable $W_t$ has a normal distribution and any pair of random variables $W_{t_1}$ and $W_{t_2}$ for distinct values of $t_1, t_2 \in \mathbb{Z}$ are uncorrelated.

**Remark 1** *For the sake of simplicity, we assume that* $\mathbb{E}[Y_t] = 0$. *Otherwise, if* $\mathbb{E}[Y_t] = \mu \neq 0$, *then one can replace* $Y_t$ *with* $Y_t - \mu$ *to obtain*

$$Y_t - \mu = \phi_1(Y_{t-1} - \mu) + \cdots + \phi_p(Y_{t-p} - \mu) + W_t,$$

*which is simplified to*

$$Y_t = \mu(1 - \phi_1 \cdots - \phi_p) + \phi_1 Y_{t-1} + \cdots + \phi_p Y_{t-p} + W_t.$$

It is readily seen that each AR($p$) model has $p+2$ unknown parameters consisting of the order $p$, the coefficients $\phi_i$ and the variance of white noises $\sigma_W^2$. Here, we briefly explain the common methods in the literature to estimate each of these unknown parameters.

**Estimating the order $p$.** A common method to estimate the order of an AR($p$) model is to use the *partial autocorrelation function* (PACF) (Shumway and Stoffer 2017, Chapter 3). The PACF of a stationary time series $\{Y_t;\, t = 0, \pm 1, \pm 2, \ldots\}$ at lag $h$ is defined by

$$\mathtt{PACF}_h := \begin{cases} \rho(Y_t, Y_{t+1}) & \text{for } h = 1, \\ \rho(Y_{t+h} - \widehat{Y}_{t+h,-h}, Y_t - \widehat{Y}_{t,h}) & \text{for } h \geq 2, \end{cases} \tag{1}$$

where $\rho$ denotes the correlation function, and where $\widehat{Y}_{t,h}$ and $\widehat{Y}_{t+h,-h}$ denote the linear regression, in the population sense, of $Y_t$ and $Y_{t+h}$ on $\{Y_{t+1}, \ldots, Y_{t+h-1}\}$, respectively. It can be shown that for a causal AR($p$) model, while the theoretical PACF (1) at lags $h = 1, \ldots, p-1$ may be non-zero and at lag $h = p$ may be strictly non-zero, at lag $h = p+1$ it drops to zero and then remains at zero henceforth (Shumway and Stoffer 2017, Chapter 3). Recall that an AR(p) model is said to be *causal* if the time series $\{Y_t;\, t = 0, \pm 1, \pm 2, \ldots\}$ can be written as $Y_t = W_t + \sum_{i=1}^{\infty} \psi_i W_{t-i}$ with constant coefficients $\psi_i$ such that $\sum_{i=1}^{\infty} |\psi_i| < \infty$. Furthermore, if a sample of size $n$ is obtained from a causal AR($p$) model, then under some mild conditions, an estimated sample PACF at lags $h > p$, scaled by $\sqrt{n}$, has a standard normal distribution, in limit as $n$ tends to infinity (Blackwell and Davis 2009, Chapter 8).

Thus, in practice, the sample PACF versus lag $h$ along with a 95% zero-confidence boundary, that is two horizontal lines at $\pm 1.96/\sqrt{n}$, are plotted. Then, the largest lag $h$ in which the sample PACF lies out of the zero-confidence boundary for PACF is used as an estimation of the order $p$. For instance, Figures 4a, 4d and 4g display the sample PACF plots for the synthetic time series data generated from models AR(20), AR(100), and AR(200), respectively. Each figure illustrates that the largest PACF lying out of the red dashed 95% zero-confidence boundary, locates at a lag which is equal to the order of the AR model.

**Maximum likelihood estimation of the coefficients $\phi_i$ and variance $\sigma^2_W$.** Let $y_1, \ldots, y_n$ be a time series realization of an $\texttt{AR}(p)$ model where $p$ is known and $n \gg p$. Unlike a linear regression model, the log-likelihood function

$$\log(f_{Y_1,\ldots,Y_n}(y_1, \ldots, y_n;\, \phi_1, \ldots, \phi_p, \sigma^2_W)),$$

where $f$ is the joint probability distribution function of the random variables $Y_1, \ldots, Y_n$, is a complicated non-linear function of the unknown parameters. Hence, finding an analytical form of the maximum likelihood estimates (MLEs) is intractable. Consequently, one typically uses some numerical optimization methods to find an MLE of the parameters of an $\texttt{AR(p)}$ model approximately. However, it can be shown that the conditional log-likelihood function is analogous to the log-likelihood function of a linear regression model given below (Hamilton 1994, Chapter 5):

$$\begin{aligned}&\log(f_{Y_{p+1},\ldots,Y_n|Y_1,\ldots,Y_p}(y_{p+1}, \ldots, y_n|y_1, \ldots, y_p;\, \phi_1, \ldots, \phi_p, \sigma^2_W))\\ &\quad = -\frac{n-p}{2}\log(2\pi) - \frac{n-p}{2}\log(\sigma^2_W) - \sum_{t=p+1}^{n} \frac{(y_t - \phi_1 y_{t-1} - \cdots - \phi_p y_{t-p})^2}{2\sigma^2_W}.\end{aligned}$$

Thus, the conditional MLE (CMLE) of the coefficients $\phi_i$ as well as the variance $\sigma^2_W$ can be estimated from an OLS regression of $y_t$ on $p$ of its own lagged values. More precisely,

$$\boldsymbol{\phi}_{n,p} := (\boldsymbol{X}^\intercal_{n,p}\boldsymbol{X}_{n,p})^{-1}\boldsymbol{X}^\intercal_{n,p}\boldsymbol{y}_{n,p}, \tag{2}$$

where $\boldsymbol{\phi}_{n,p}$ is the CMLE of the coefficient vector $[\phi_1, \ldots, \phi_p]^\intercal$, the data matrix

$$\boldsymbol{X}_{n,p} := \begin{pmatrix} y_p & y_{p-1} & \cdots & y_1 \\ y_{p+1} & y_p & \cdots & y_2 \\ \vdots & \vdots & \ddots & \vdots \\ y_{n-1} & y_{n-2} & \cdots & y_{n-p} \end{pmatrix}, \tag{3}$$

and $\boldsymbol{y}_{n,p} := \begin{bmatrix} y_{p+1} & y_{p+2} & \ldots & y_n \end{bmatrix}^\intercal$.

**Remark 2** *The data matrix $\boldsymbol{X}_{n,p}$ in* (3) *possesses Toeplitz structure that we take advantage of for our derivations in this paper, in particular developing the recursion for the leverage scores given in Theorem 7. Also, it is highlighted that as the estimated parameter vector* (2) *is operating under "conditional" MLE, the data matrix $\boldsymbol{X}_{n,p}$ is a fixed design matrix.*

Moreover, the CMLE of $\sigma^2_W$, the so-called MSE, is given by

$$\widehat{\sigma}^2_W = \frac{\|\boldsymbol{r}_{n,p}\|^2}{n-p},$$

where

$$\boldsymbol{r}_{n,p} := \boldsymbol{y}_{n,p} - \boldsymbol{X}_{n,p}\boldsymbol{\phi}_{n,p} \;=\; \begin{bmatrix} \boldsymbol{r}_{n,p}(1) & \ldots & \boldsymbol{r}_{n,p}(n-p) \end{bmatrix}^\intercal \tag{4}$$

and

$$\boldsymbol{r}_{n,p}(i) = y_{p+i} - \boldsymbol{X}_{n,p}^{\intercal}(i,:)\boldsymbol{\phi}_{n,p} \quad \text{for } i = 1, \ldots, n-p.$$

Recall that $\boldsymbol{X}_{n,p}(i,:)$ is the $i^{\text{th}}$ row of matrix $\boldsymbol{X}_{n,p}$, that is,

$$\boldsymbol{X}_{n,p}(i,:) := \begin{bmatrix} y_{i+p-1} & y_{i+p-2} & \ldots & y_i \end{bmatrix}^{\intercal}.$$

One may criticize the CMLE as it requires one to exclude the first $p$ observations to construct the conditional log-likelihood function. Although this is a valid statement, due to the assumption $n \gg p$, dropping the first $p$ observation from the whole time series realization could be negligible.

**Remark 3** *It can be shown (Shumway and Stoffer 2017, Chapter 3) that if*

$$\widehat{Y}_{t+h,-h} = \alpha_1 Y_{t+h-1} + \cdots + \alpha_{h-1} Y_{t+1},$$

*then*

$$\widehat{Y}_{t,h} = \alpha_1 Y_{t+1} + \cdots + \alpha_{h-1} Y_{t+h-1}.$$

*This implies that finding PACF at each lag requires the solution to only one corresponding OLS problem. Furthermore, one can see that an empirical estimation of the coefficients $\alpha_i$ is the same as finding a CMLE of the coefficients of an* $\texttt{AR}(h-1)$ *model fitted to the data. Thus, empirically estimating the order $p$ using a given time series data involves* repeated *solutions of OLS problems, which can be computationally prohibitive in large-scale settings. Indeed, for $n$ realizations $y_1, \ldots, y_n$, PACF at lag $h$ can be calculated in $\mathcal{O}(nh)$ using Toeplitz properties of the underlying matrix, and as a result selecting an appropriate order parameter $p$ amounts to $\mathcal{O}\left(\sum_{h=1}^{p} nh\right) = \mathcal{O}\left(np^2\right)$ time complexity.*

**Remark 4** *It should be noted that there is another method to estimate the parameters of an* $\texttt{AR}(p)$ *model by solving the Yule-Walker equations with the Durbin-Levinson algorithm (Blackwell and Davis 2009, Chapter 8). Although, those estimates have asymptotic properties similar to CMLEs, solving the corresponding OLS problem is computationally faster than the Durbin-Levinson algorithm and also the CMLEs are statistically more efficient.*

### 2.2 Leverage Scores and RandNLA

Linear algebra, which is the mathematics of linear mappings between vector spaces, has long had a large footprint in statistical data analysis. For example, canonical linear algebra problems such as principal component analysis and OLS are arguably among the first and most widely used techniques by statisticians. In the presence of large amounts data, however, such linear algebra routines, despite their simplicity of formulation, can pose significant computational challenges. For example, consider an over-determined OLS problem

$$\min_{\boldsymbol{x}} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2, \tag{5}$$

involving $m \times d$ matrix $\boldsymbol{A}$, where $m > d$. Note that, instead of $n-p$ and $p$ for the dimensions of the matrix (3), we adopt the notation $m$ and $d$ for the number of rows and columns, respectively. This is due to the fact that our discussion in this section involves arbitrary matrices and not those specifically derived from AR models. Solving (5) amounts to $\mathcal{O}\left(md^2 + d^3/3\right)$ flops by forming the normal equations, $\mathcal{O}\left(md^2 - d^3\right)$ flops via QR factorization with Householder reflections, and $\mathcal{O}\left(md^2 + d^3\right)$ flops using singular value decomposition (SVD) (Golub and Van Loan 1983). Iterative solvers such as LSQR (Paige and Saunders 1982), LSMR (Fong and Saunders 2011), and LSLQ (Estrin et al. 2019), involve matrix-vector products at each iterations, which amount to $\mathcal{O}\left(mdc\right)$ flops after $c$ iterations. In other words, in "big-data" regimes where $md^2 \gg 1$, naïvely performing these algorithms can be costly.

RandNLA subroutines involve the construction of an appropriate sampling/sketching matrix, $\boldsymbol{S} \in \mathbb{R}^{s \times m}$ for $d \leq s \ll m$, and compressing the data matrix into a smaller version $\boldsymbol{SA} \in \mathbb{R}^{s \times d}$. In the context of (5), using the smaller matrix, the above-mentioned classical OLS algorithms can be readily applied to the smaller scale problem

$$\min_{\boldsymbol{x}} \left\|\boldsymbol{SAx} - \boldsymbol{Sb}\right\|^2, \tag{6}$$

at much lower costs. In these algorithms, sampling/sketching is used to obtain a data-oblivious or data-aware subspace embedding, which ensures that for any $0 < \varepsilon, \delta < 1$ and for large enough $s$, we get

$$\Pr\left(\left\|\boldsymbol{Ax}^\star - \boldsymbol{b}\right\|^2 \leq \left\|\boldsymbol{Ax}_s^\star - \boldsymbol{b}\right\|^2 \leq \left(1 + \mathcal{O}\left(\varepsilon\right)\right)\left\|\boldsymbol{Ax}^\star - \boldsymbol{b}\right\|^2\right) \geq 1 - \delta, \tag{7}$$

where $\boldsymbol{x}^\star$ and $\boldsymbol{x}_s^\star$ are the solutions to (5) and (6), respectively. In other words, the solution to the reduced problem (6) is a $1 + \mathcal{O}\left(\varepsilon\right)$ approximation of the solution to the original problem (5).

Arguably, the simplest data-oblivious way to construct the matrix $\boldsymbol{S}$ is using uniform sampling, where each row of $\boldsymbol{S}$ is chosen uniformly at random (with or without replacement) from the rows of the $m \times m$ identity matrix. Despite the fact that the construction and application of such a matrix can be done in constant $\mathcal{O}\left(1\right)$ time, in the presence of non-uniformity among the rows of $\boldsymbol{A}$, such uniform sampling strategies perform very poorly. In such cases, it can be shown that one indeed requires $s \in \mathcal{O}\left(m\right)$ samples to obtain the above sub-space embedding property.

To alleviate this significant shortcoming, data-oblivious sketching schemes involve randomly transforming the data so as to smooth out the non-uniformities, which in turn allows for subsequent uniform sampling in the randomly rotated space (Drineas et al. 2011). Here, the random projection acts as a preconditioner (for the class of random sampling algorithms), which makes the preconditioned data better behaved (in the sense that simple uniform sampling methods can be used successfully) (Mahoney 2011, 2016). With such sketching schemes, depending on the random projection matrix, different sample sizes are required, for instance, $\mathcal{O}\left(d\log(1/\delta)/\varepsilon^2\right)$ samples for Gaussian projection, $\mathcal{O}\left(d\log(d/\delta)/\varepsilon^2\right)$ samples for fast Hadamard-based transforms, and $\mathcal{O}\left(d^2\text{poly}(\log(d/\delta))/\varepsilon^2\right)$ samples using sparse embedding matrices. Woodruff (2014) provided a comprehensive overview of such methods and their extensions.

Alternative to data-oblivious random embedding methods are data-aware sampling techniques, which by taking into account the information contained in the data, sample the rows of the matrix proportional to non-uniform distributions. Among many such strategies, those schemes based on *statistical leverage scores* (Drineas et al. 2012) have not only shown to improve worst case theoretical guarantees of matrix algorithms, but also they are amenable to high-quality numerical implementations (Mahoney 2011). Roughly speaking, the "best" random sampling algorithms base their importance sampling distribution on these scores and the "best" random projection algorithms transform the data to be represented in a rotated basis where these scores are approximately uniform.

The concept of statistical leverage score has long been used in statistical regression diagnostics to identify outliers (Rousseeuw and Hubert 2011). Given a data matrix $\boldsymbol{A} \in \mathbb{R}^{m\times d}$ with $m \geq d$, consider any orthogonal matrix $\boldsymbol{Q}$ such that $\text{Range}(\boldsymbol{Q}) = \text{Range}(\boldsymbol{A})$. The $i^{\text{th}}$ leverage score corresponding to $i^{\text{th}}$ row of $\boldsymbol{A}$ is defined as

$$\ell(i) := \|\boldsymbol{Q}(i,:)\|^2 .$$

It can be easily shown that this is well-defined in that the leverage score does not depend on the particular choice of the basis matrix $\boldsymbol{Q}$. Furthermore, the $i^{\text{th}}$ leverage score boils down to the $i^{\text{th}}$ diagonal entry of the *hat* matrix, that is,

$$\ell(i) = \boldsymbol{e}_i^{\intercal}\boldsymbol{H}\boldsymbol{e}_i \quad \text{for } i = 1, \ldots, m, \tag{8a}$$

where

$$\boldsymbol{H} := \boldsymbol{A}\left(\boldsymbol{A}^{\intercal}\boldsymbol{A}\right)^{-1}\boldsymbol{A}^{\intercal}. \tag{8b}$$

It is also easy to see that

$$\ell(i) \geq 0 \;\; \forall\, i, \qquad \text{and} \qquad \sum_{i=1}^{m} \ell(i) = d.$$

Thus,

$$\pi(i) := \frac{\ell(i)}{d}, \quad \text{for } i = 1, \ldots, m, \tag{8c}$$

defines a non-uniform probability distribution over the rows of $\boldsymbol{A}$.

**Leverage score sampling matrix $\boldsymbol{S}$.** Sampling according to the leverage scores amounts to randomly picking and re-scaling rows of $\boldsymbol{A}$ proportional to their leverage scores and appropriately re-scaling the sampled rows so as to maintain an unbiased estimator of $\boldsymbol{A}^{\intercal}\boldsymbol{A}$, that is,

$$\mathbb{E}[\|\boldsymbol{S}\boldsymbol{A}\boldsymbol{x}\|^2] = \|\boldsymbol{A}\boldsymbol{x}\|^2 , \; \forall \boldsymbol{x}.$$

More precisely, each row of the $s \times m$ sampling matrix $\boldsymbol{S}$ is chosen randomly from the rows of the $m \times m$ identity matrix according to the probability distribution (8c), with replacement. Furthermore, if the $i^{th}$ row is selected, it is re-scaled with the multiplicative factor

$$\frac{1}{\sqrt{s\pi_i}}, \tag{9}$$

implying that $1/\sqrt{s\pi_i}\boldsymbol{e}_i^\intercal$ is appended to $\boldsymbol{S}$.

Clearly, obtaining any orthogonal matrix $\boldsymbol{Q}$ as above by using SVD or QR factorization is almost as costly as solving the original OLS problem (i.e., $\mathcal{O}\left(md^2\right)$ flops), which defeats the purpose of sampling altogether. In this light, Drineas et al. (2012) proposed randomized approximation algorithms, which efficiently estimate the leverage scores in $\mathcal{O}\left(md\log m + d^3\right)$ flops. For sparse matrices, this was further improved by Clarkson and Woodruff (2017), Meng and Mahoney (2013), and Nelson and Nguyen (2013) to $\mathcal{O}\left(nnz(\boldsymbol{A})\log m + d^3\right)$. In particular, it has been shown that with the leverage score estimates $\hat{\ell}(i)$ such that

$$\hat{\ell}(i) \geq \beta\ell(i), \quad \text{for } i = 1, 2, \ldots m, \tag{10}$$

for some *misestimation factor* $0 < \beta \leq 1$, one can obtain (7) with

$$s \in \mathcal{O}\left(d\log(d/\delta)/(\beta\varepsilon^2)\right), \tag{11}$$

samples (Woodruff 2014). As it can be seen from (11), the required sample size $s$ is adversely affected by the leverage score misestimation factor $\beta$.

Recently, randomized sublinear time algorithms for estimating the parameters of an `AR` model for a given order $d$ have been developed by Shi and Woodruff (2019). There, by using the notion of generalized leverage scroes, the authors propose a method for approximating CMLE of the parameters in $\mathcal{O}(m\log^2 m + (d^2\log^2 m)/\varepsilon^2 + (d^3\log m)/\varepsilon^2)$ time, with high probability. The analysis in Shi and Woodruff (2019) makes use of Toeplitz structure of data matrices arising from `AR` models. Also related to our settings here are Van Barel et al. (2003) and Xi et al. (2014), which developed, respectively, an exact and a (numerically stable) randomized approximation algorithm to solve Toeplitz least square problems, both with the time complexity of $\mathcal{O}\left((m+d)\log^2(m+d)\right)$. An alternative sub-sampling algorithm to algorithmic leveraging for OLS problems has been considered by Wang et al. (2018). There, the sub-sampling is approached from the perspective of optimal design using D-optimality criterion, aiming to maximize the determinant of the Fisher information in the sub-sample. We also note that algorithms various statistical aspects of leverage scores have been extensively studied by Raskutti and Mahoney (2016) and Ma et al. (2015). Finally, a more general notion of leverage scores in the context of recovery of continuous time signals from discrete measurements has recently been introduced by Avron et al. (2019).

### 2.3 Theoretical Contributions

Here, by taking the advantage of the structure of `AR` models, we derive an algorithm, called `LSAR`, which given the (approximate) leverage scores of the data matrix for an `AR`$(p-1)$ model (cf. Equation 3), efficiently provides an estimate for the leverage scores related to an `AR`$(p)$ model. In the process, we derive explicit bounds on the misestimation factor $\beta$ in (10). An informal statement of our main results (Theorems 13, 16, 17 and 21) are as follows.

**Claim (Informal).** For any $\varepsilon > 0$ small enough, we prove (with a constant probability of success):

- Theorem 13: If only some suitable approximations of the leverage scores of an `AR`$(p-1)$ model are known, we can estimate those of an `AR`$(p)$ model with a misestimation factor

$\beta \in 1-\mathcal{O}(p\sqrt{\varepsilon})$ in $\mathcal{O}(n+p^3 \log p)$ time complexity. This should be compared with naïve QR-based methods with $\mathcal{O}\left(np^2\right)$ and the universal approximation schemes developed by Drineas et al. (2012) with $\mathcal{O}\left(np \log n + p^3\right)$.

- Theorems 16 and 17: Furthermore, an appropriate AR($p$) model can be fitted, with high-probability, in overall time complexity of $\mathcal{O}(np + (p^4 \log p)/\varepsilon^2)$ as compared with $\mathcal{O}(np^2)$ using exact methods (cf. Theorem 3), $\mathcal{O}((n+p)p \log^2(n+p))$ by leveraging structured matrices as in Van Barel et al. (2003), and $\mathcal{O}(np \log^2 n + (p^3 \log^2 n)/\varepsilon^2 + (p^4 \log n)/\varepsilon^2)$ from sublinear time algorithms developed by Shi and Woodruff (2019).

**Remark 5** *In big data regimes where typically $n \gg p$ the above result implies an improvement over the existing methods for fitting an appropriate AR model. However, we believe that the dependence of the misestimation factor $\beta \in 1 - \mathcal{O}(p\sqrt{\varepsilon})$ on $p$ is superfluously a by-product of our analysis, as in our numerical experiments, we show that a sensible factor may be in the order of $\beta \in 1 - \mathcal{O}(\log p\sqrt{\varepsilon})$.*

## 3. Theoretical Results

In this section, we use the specific structure of the data matrix induced by an AR model to develop a fast algorithm to approximate the leverage scores corresponding to the rows of the data matrix (3). Furthermore, we theoretically show that our approximations possess relative error (cf. Equation 18) bounds with high probability. Motivated from the leverage score based sampling strategy in Section 2.2, we then construct a highly efficient algorithm, namely LSAR, to fit an appropriate AR($p$) model on big time series data. It should be noted that all proofs of this section are presented in Appendix A.

### 3.1 Leverage Score Approximation for AR Models

We first introduce Theorem 6 which relates and unifies notation of Sections 2.1 and 2.2 together.

**Definition 6** *In what follows, we define $\ell_{n,p}$, $\boldsymbol{H}_{n,p}$, and $\pi_{n,p}$ as*

$$\begin{aligned} \ell_{n,p}(i) &:= \boldsymbol{e}_i^\intercal \boldsymbol{H}_{n,p} \boldsymbol{e}_i, \quad \text{for } i = 1, \ldots, n-p, \\ \boldsymbol{H}_{n,p} &:= \boldsymbol{X}_{n,p} \left(\boldsymbol{X}_{n,p}^\intercal \boldsymbol{X}_{n,p}\right)^{-1} \boldsymbol{X}_{n,p}^\intercal, \\ \pi_{n,p}(i) &:= \frac{\ell_{n,p}(i)}{p}, \quad \text{for } i = 1, \ldots, n-p. \end{aligned}$$

*That is, they refer, respectively, to* (8a),(8b), *and* (8c), *using $\boldsymbol{A} = \boldsymbol{X}_{n,p}$ as defined in* (3).

We show that the leverage scores associated with an AR($p$) model can be recursively described using those arising from an AR($p-1$) model. This recursive pattern is a direct result of the special structure of the data matrix (3), which amounts to a rectangular Hankel matrix (Golub and Van Loan 1983).

**Theorem 7 (Exact Leverage Score Computations)** *The leverage scores of an* $\texttt{AR(1)}$ *model are given by*

$$\ell_{n,1}(i) = \frac{y_i^2}{\sum_{t=1}^{n-1} y_t^2}, \quad \textit{for } i = 1, \ldots, n-1. \tag{12a}$$

*For an* $\texttt{AR(p)}$ *model with* $p \geq 2$*, the leverage scores are obtained by the following recursion*

$$\ell_{n,p}(i) = \ell_{n-1,p-1}(i) + \frac{\left(\boldsymbol{r}_{n-1,p-1}(i)\right)^2}{\left\|\boldsymbol{r}_{n-1,p-1}\right\|^2}, \quad \textit{for } i = 1, \ldots, n-p, \tag{12b}$$

*where the residual vector* $\boldsymbol{r}_{n-1,p-1}$ *is defined in* (4)*.*

Theorem 7 shows that the leverage scores of (3) can be exactly calculated through the recursive (12b) on the parameter $p$ with the initial condition (12a). This recursion incorporates the leverage cores of the data matrix $\boldsymbol{X}_{n-1,p-1}$ along with the residual terms of fitting an $\texttt{AR(p}-\texttt{1)}$ model to the time series data $y_1, \ldots, y_{n-1}$. Note that both matrices $\boldsymbol{X}_{n-1,p-1}$ and $\boldsymbol{X}_{n,p}$ have equal number of rows, and accordingly equal number of leverage scores. Moreover, since we are dealing with big time series data (i.e., $n \gg p$), excluding one observation in practice is indeed negligible.

Theorem 7, though enticing at first glance, suffers from two major drawbacks in that not only does it require exact leverage scores associated with $\texttt{AR}(p-1)$ models, but it also involves exact residuals from the corresponding OLS estimations. In the presence of big data, computing either of these factors exactly defeats the whole purpose of data sampling altogether. To alleviate these two issues, we first focus on approximations in computing the latter, and then incorporate the estimations of the former. In doing so, we obtain leverage score approximations, which enjoy desirable a priori relative error bounds.

A natural way to approximate the residuals in the preceding $\texttt{AR}(p-1)$ model (i.e., $\boldsymbol{r}_{n-1,p-1}$), is by means of sampling the data matrix $\boldsymbol{X}_{n-1,p-1}$ and solving the corresponding reduced OLS problem. More specifically, we consider the sampled data matrix

$$\tilde{\boldsymbol{X}}_{n,p} := \boldsymbol{S}\boldsymbol{X}_{n,p},$$

where $\boldsymbol{S} \in \mathbb{R}^{s \times (n-p)}$ is the sampling matrix whose $s$ rows are chosen at random with replacement from the rows of the $(n-p) \times (n-p)$ identity matrix according to the distribution $\{\pi_{n,p}(i)\}_{i=1}^{n-p}$ (cf. Theorem 6) and rescaled by the appropriate factor (9). Using $\tilde{\boldsymbol{X}}_{n,p}$, the estimated parameter vector $\tilde{\boldsymbol{\phi}}_{n,p}$ is calculated as

$$\tilde{\boldsymbol{\phi}}_{n,p} := (\tilde{\boldsymbol{X}}_{n,p}^{\intercal}\tilde{\boldsymbol{X}}_{n,p})^{-1}\tilde{\boldsymbol{X}}_{n,p}^{\intercal}\tilde{\boldsymbol{y}}_{n,p}, \tag{13a}$$

where $\tilde{\boldsymbol{y}}_{n,p} := \boldsymbol{S}\boldsymbol{y}_{n,p}$. Finally, the residuals of $\tilde{\boldsymbol{\phi}}_{n,p}$, analogous to (4), are given by

$$\tilde{\boldsymbol{r}}_{n,p} := \boldsymbol{y}_{n,p} - \boldsymbol{X}_{n,p}\tilde{\boldsymbol{\phi}}_{n,p}. \tag{13b}$$

**Remark 8** *We note that the residual vector $\tilde{\boldsymbol{r}}_{n,p}$ is computed using the sampled data matrix $\tilde{\boldsymbol{X}}_{n,p}$, which is itself formed according to the leverage scores. In other words, the availability $\tilde{\boldsymbol{r}}_{n,p}$ is equivalent to that of $\{\pi_{n,p}(i)\}_{i=1}^{n-p}$.*

The following theorem, derived from the structural result (Drineas et al. 2011), gives estimates on the approximations (13a) and (13b).

**Theorem 9 (Drineas et al. 2011, Theorem 1)** *Consider an* AR(p) *model and let $0 < \varepsilon, \delta < 1$. For sampling with (approximate) leverage scores using a sample size $s$ as in* (11) *with $d = p$, we have with probability at least $1 - \delta$,*

$$\|\tilde{\boldsymbol{r}}_{n,p}\| \leq (1 + \varepsilon) \|\boldsymbol{r}_{n,p}\|, \tag{14a}$$

$$\left\|\boldsymbol{\phi}_{n,p} - \tilde{\boldsymbol{\phi}}_{n,p}\right\| \leq \sqrt{\varepsilon}\eta_{n,p} \|\boldsymbol{\phi}_{n,p}\|, \tag{14b}$$

*where $\boldsymbol{\phi}_{n,p}, \boldsymbol{r}_{n,p}, \tilde{\boldsymbol{\phi}}_{n,p}$ and $\tilde{\boldsymbol{r}}_{n,p}$ are defined,, respectively, in* (2), (4), (13a) *and* (13b),

$$\eta_{n,p} = \kappa(\boldsymbol{X}_{n,p})\sqrt{\xi^{-2} - 1}, \tag{14c}$$

*$\kappa(\boldsymbol{X}_{n,p})$ is the condition number of matrix $\boldsymbol{X}_{n,p}$, and $\xi \in (0, 1]$ is the fraction of $\boldsymbol{y}_{n,p}$ that lies in* $\text{Range}(\boldsymbol{X}_{n,p})$*, that is, $\xi := \|\boldsymbol{H}_{n,p}\boldsymbol{y}_{n,p}\| / \|\boldsymbol{y}_{n,p}\|$ with $\boldsymbol{H}_{n,p}$ as in Theorem 6.*

Using a combination of exact leverage scores and the estimates (13b) on the OLS residuals associated with the AR($p-1$) model, we define *quasi-approximate leverage scores* for the AR($p$) model.

**Definition 10 (Quasi-approximate Leverage Scores)** *For an* AR(p) *model with $p \geq 2$, the quasi-approximate leverage scores are defined by the following equation*

$$\tilde{\ell}_{n,p}(i) := \ell_{n-1,p-1}(i) + \frac{(\tilde{\boldsymbol{r}}_{n-1,p-1}(i))^2}{\|\tilde{\boldsymbol{r}}_{n-1,p-1}\|^2} \quad \textit{for } i = 1, \ldots, n-p, \tag{15}$$

*where $\ell_{n,p}(i)$ and $\tilde{\boldsymbol{r}}_{n,p}$ are as in Theorem 6 and* (13b).

Clearly, the practical advantage of $\tilde{\ell}_{n,p}$ is entirely contingent upon the availability of the exact leverage scores for $p-1$, that is, $\ell_{n-1,p-1}$ (cf. Theorem 10). For $p = 2$, this is indeed possible. More specifically, from (12a), the exact leverage scores of an AR(1) model can be trivially calculated, which in turn give the quasi-approximate leverage scores $\{\tilde{\ell}_{n-2,2}(i)\}_{i=1}^{n-2}$ using (15). However, for $p = 3$ (and subsequent values), the relation (15) does not apply as not only are $\{\ell_{n-1,p-1}(i)\}_{i=1}^{n-p}$ no longer readily available, but also for the same token without having $\{\pi_{n-1,p-1}(i)\}_{i=1}^{n-p}$, the residual vector $\tilde{\boldsymbol{r}}_{n-1,p-1}$ may not be computed directly (cf. Theorem 8). Nonetheless, replacing the exact leverage scores with quasi-approximate ones in (15) for $p = 2$ allows for a new approximation for $p = 3$. Such new leverage score estimates can be in turn incorporated in approximation of subsequent leverage scores for $p \geq 4$. This idea leads to our final and practical definition of *fully-approximate leverage scores.*

**Definition 11 (Fully-approximate Leverage Scores)** *For an* AR(p) *model with* $p \geq 1$, *the fully-approximate leverage scores are defined by the following equation*

$$\hat{\ell}_{n,p}(i) := \begin{cases} \ell_{n,1}(i), & \text{for } p = 1 \\ \tilde{\ell}_{n,2}(i), & \text{for } p = 2 \\ \hat{\ell}_{n-1,p-1}(i) + \dfrac{\left(\hat{\boldsymbol{r}}_{n-1,p-1}(i)\right)^2}{\|\hat{\boldsymbol{r}}_{n-1,p-1}\|^2}, & \text{for } p \geq 3 \end{cases}, \tag{16a}$$

*where*

$$\hat{\boldsymbol{r}}_{n-1,p-1} := \boldsymbol{y}_{n-1,p-1} - \boldsymbol{X}_{n-1,p-1}\hat{\boldsymbol{\phi}}_{n-1,p-1}, \tag{16b}$$

$$\hat{\boldsymbol{\phi}}_{n-1,p-1} := (\hat{\boldsymbol{X}}^{\intercal}_{n-1,p-1}\hat{\boldsymbol{X}}_{n-1,p-1})^{-1}\hat{\boldsymbol{X}}^{\intercal}_{n-1,p-1}\hat{\boldsymbol{y}}_{n-1,p-1} \tag{16c}$$

*and* $\hat{\boldsymbol{X}}_{n-1,p-1}$ *and* $\hat{\boldsymbol{y}}_{n-1,p-1}$ *are the reduced data matrix and response vector, sampled respectively, according to the distribution*

$$\hat{\pi}_{n-1,p-1}(i) = \frac{\hat{\ell}_{n-1,p-1}(i)}{p-1} \quad \text{for } i = 1, \ldots, n-p. \tag{16d}$$

**Remark 12** *It should be noted that* (15) *estimates the leverage scores of an* AR($p$) *model, given the corresponding* exact *values of an* AR($p-1$) *model. This is in sharp contrast to* (16a)*, which recursively provides similar estimates without requiring any information on the exact values.*

Unlike the quasi-approximate leverage scores, the fully-approximate ones in Theorem 11 can be easily calculated for any given the parameter value $p \geq 1$. Finally, Theorem 13 provides a priori relative-error estimate on individual fully-approximate leverage scores.

**Theorem 13 (Relative Errors for Fully-approximate Leverage Scores)** *For the fully-approximate leverage scores, we have with probability at least* $1 - \delta$,

$$\frac{|\ell_{n,p}(i) - \hat{\ell}_{n,p}(i)|}{\ell_{n,p}(i)} \leq \left(1 + 3\eta_{n-1,p-1}\kappa^2(\boldsymbol{X}_{n,p})\right)(p-1)\sqrt{\varepsilon}, \quad \text{for } i = 1, \ldots, n-p,$$

*recalling that* $\delta$, $\eta_{n,p}$, $\kappa(\boldsymbol{X}_{n,p})$, *and* $\varepsilon$ *are as in Theorem 9.*

Although qualitatively descriptive, the bound in Theorem 13 is admittedly pessimistic and involves an overestimation factor that scales quadratically with the condition number of the data matrix, $\kappa$, and linearly with the order of the AR model, $p$. We conjecture that the linear dependence on $p$ can be instead replaced with $\log(p)$, which is supported by the experiment depicted in Figure 2. We leave the investigation of ways to improve the upper-bound of Theorem 13 to future work.

Theorem 13 prescribes the misestimation factor $\beta$ (cf. Equation 10) for the fully-approximate leverage sores of an AR($p$) model, stated in Theorem 14.

**Corollary 14** *The misestimation factor* $\beta$ *for the fully-approximate leverage scores of an* AR($p$) *model is* $1 - \mathcal{O}(p\sqrt{\varepsilon})$.

### 3.2 LSAR Algorithm for Fitting AR Models

Based on these theoretical results, we introduce the LSAR algorithm, depicted in Algorithm 1, which is the first leverage score sampling algorithm to approximately fit an appropriate AR model to a given big time series data. The theoretical properties of the LSAR algorithm are given in Theorems 16 and 17.

**Algorithm 1** LSAR: Leverage Score Sampling Algorithm for Approximate AR Fitting

```
Input:
    - Time series data $\{y_1, \dots, y_n\}$ ;
    - A relatively large value $\bar{p} \ll n$ ;
    - Constant parameters $0 < \varepsilon < 1$ and $0 < \delta_0 < 1$;
Step 0. Set $p = 0$ and $m = n - \bar{p}$ ;
while $p < \bar{p}$ do
    Step 1. $p \leftarrow p + 1$ and $m \leftarrow m + 1$ ;
    Step 2. Estimate PACF at lag $p$, i.e., $\hat{\tau}_p$ ;
    Step 3. Compute the approximate leverage scores $\hat{\ell}_{m,p}(i)$ for $i = 1, \dots, m - p$ as in (16a) ;
    Step 4. Compute the sampling distribution $\hat{\pi}_{m,p}(i)$ for $i = 1, \dots, m - p$ as in (16d) ;
    Step 5. Set $s$ as in (11) by replacing $d$ with $p$, $\delta = \delta_0/p$, and $\beta$ with the bound given in Theorem 14 ;
    Step 6. Form the $s \times m$ sampling matrix $\boldsymbol{S}$ by randomly choosing $s$ rows of the corresponding identity matrix according to the probability distribution found in Step 4, with replacement, and rescaling them with the factor (9) ;
    Step 7. Construct the sampled data matrix $\hat{\boldsymbol{X}}_{m,p} = \boldsymbol{S}\boldsymbol{X}_{m,p}$ and response vector $\hat{\boldsymbol{y}}_{m,p} = \boldsymbol{S}\boldsymbol{y}_{m,p}$ ;
    Step 8. Solve the associated reduced OLS problem to estimate the parameters $\hat{\boldsymbol{\phi}}_{m,p}$ and residuals $\hat{\boldsymbol{r}}_{m,p}$ as in (16b) and (16c), respectively ;
end while
Step 9. Estimate $p^*$ as the largest $p$ such that $|\hat{\tau}_p| \geq 1.96/\sqrt{s}$ ;
Output: Estimated order $p^*$ and parameters $\hat{\boldsymbol{\phi}}_{n-\bar{p}+p^*,p^*}$.
```

**Remark 15** *For the overall failure probability, recall that in order to get an accumulative success probability of* $1 - \delta_0$ *for* $\bar{p}$ *iterations, the per-iteration failure probability is set as* $\delta = 1 - \sqrt[\bar{p}]{1 - \delta_0} \in \Omega(\delta_0/\bar{p})$. *However, since this dependence manifest itself only logarithmically, it is of negligible consequence in overall complexity.*

The quality of the fitted model by the LSAR algorithm depends on two crucial ingredients, the order of the underlying AR model as well the accuracy of the estimated parameters. The latter is guaranteed by Theorem 9. For the former, Theorem 16 shows that for small enough $\varepsilon$, the LSAR algorithm can estimate the same model order as that using the full data matrix.

Let $\tau_p$ and $\hat{\tau}_p$ be the PACF values estimated using the CMLE of parameter vectors based on the full and sampled data matrices, $\boldsymbol{\phi}_{n,p-1}$ and $\hat{\boldsymbol{\phi}}_{n,p-1}$, respectively.

**Theorem 16 (LSAR Model-order Estimation)** *Consider a causal* AR(p$^*$) *model and let* $0 < \varepsilon, \delta < 1$. *For sampling with fully-approximate leverage scores using a sample size* $s$ *as in* (11) *with* $d = p^*$ *and* $\beta$ *as in Theorem 14 with* $p = p^*$, *we have with probability at least* $1 - \delta$,

$$|\hat{\tau}_p| \geq |\tau_p| - c_1\sqrt{\varepsilon}, \qquad \text{for } p = p^*, \tag{17a}$$

$$|\hat{\tau}_p| \leq |\tau_p| + c_2\sqrt{(p-1)\varepsilon}, \quad \text{for } p > p^*, \tag{17b}$$

*where* $c_1$ *and* $c_2$ *are bounded positive constants depending on a given realization of the model.*

Theorem 16 implies that, when $|\tau_{p^*}| \geq 1.96/\sqrt{n}$ and $|\tau_p| \leq 1.96/\sqrt{n}$ for $p > p^*$, with high probability, we are guaranteed to have $|\hat{\tau}_{p^*}| \geq 1.96/\sqrt{n} - \mathcal{O}(\sqrt{\epsilon})$ and $|\hat{\tau}_p| \leq 1.96/\sqrt{n} + \mathcal{O}(\sqrt{\epsilon})$ for $p > p^*$, respectively. In practice, we can consider a larger bandwidth of size $2 \times 1.96/\sqrt{s}$; see the experiments of Section 4.

Theorem 17 gives the overall running time of the LSAR algorithm.

**Theorem 17 (LSAR Computational Complexity)** *The worst case time complexity of the LSAR algorithm for an input* AR(p$^*$) *time series data is* $\mathcal{O}\left(np^* + {p^*}^4 \log p^*/\varepsilon^2\right)$, *with probability at least* $1 - \delta_0$ $(0 < \delta_0 < 1)$ *and the* $p^{th}$ *iteration of the algorithm has* $\delta = \delta_0/p$, *which appears in the log for each sample size.*

**Remark 18** *We believe that the restriction on* $\varepsilon$ *given by Theorem 17 is highly pessimistic and merely a by-product of our proof techniques here. As evidenced by numerical experiments, e.g., Figure 2, we conjecture that a more sensible bound is* $0 < \varepsilon \leq (\log p^*)^{-2}$; *see also the discussion in the last paragraph of Section 2 and Theorem 5. In fact, even the tight bounds on the sample size for RandNLA routines rarely manifest themselves in practice (Roosta-Khorasani et al. 2015; Mahoney 2011, 2016). Guided by these observations, in our numerical experiments of Section 4, we set our sample sizes at factions of the total data, e.g.,* $s = 0.001n$, *even for small values of* $p^*$.

## 4. Empirical Results

In this section, we present the performance of the LSAR algorithm on several synthetic as well as real big time series data. The numerical experiments are run in MATLAB R2018b on a 64-bit windows server with dual processor each at 2.20GHz with 128 GB installed RAM.

The numerical results reveal the efficiency of the LSAR algorithm, as compared with the classical alternative using the entire data. More precisely, it is illustrated that by sampling only 0.1% of the data, not only are the approximation errors kept significantly small, but also the underlying computational times are considerably less than the corresponding exact algorithms.

We present our numerical analysis in three subsequent sections. In Section 4.1, we report the computational times as well as the quality of leverage score approximations (16a) on three synthetically generated data by running Steps 0-8 of the LSAR algorithm. Analogously, Section 4.2 shows similar results for estimating PACF (i.e., the output of Step 2 in the LSAR algorithm). Finally, Section 4.3 displays the performance of the LSAR algorithm on a real big time series data. It should be noted that all computational times reported in this section are in “seconds”.

### 4.1 Synthetic Data: Verification of Theory

We generate synthetic large-scale time series data with two million realizations from the models `AR(20)`, `AR(100)`, and `AR(200)`. For each data set, the leverage scores over a range of lag values (i.e., the variable $h$ in the `LSAR` algorithm) are calculated once by using the exact formula as given in Theorem 6, and another time by estimating the fully-approximate leverage scores as defined in (16a). The latter is computed by running Steps 0-8 of the `LSAR` algorithm with $s = 0.001n = 2000$.

Figure 1 displays and compares the quality and run time between the fast sampled randomized Hadamard transform (SRHT) approximation technique developed by Drineas et al. (2012) and (16). At each lag $p$, the maximum pointwise relative error (`MPRE`, for short) is defined by

$$\max_{1 \leq i \leq n-p} \left\{ \frac{|\hat{\ell}_{n,p}(i) - \ell_{n,p}(i)|}{\ell_{n,p}(i)} \right\}. \tag{18}$$

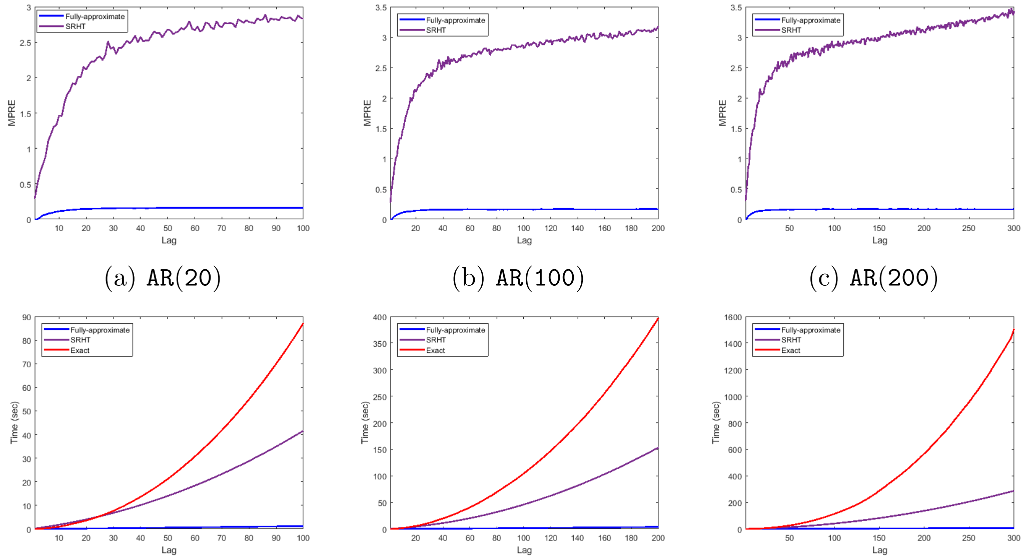
As displayed in Figures 1a to 1c, while the `MPRE` curves have sharp increase at the beginning and then quickly converge to an upper limit around 0.1670 for fully-approximate leverage scores, the output of SRHT seems to converge around 3. This demonstrates the high-quality of the fully-approximate leverage scores using only 0.1% of the rows of the data matrix. More interestingly, Figures 1d to 1f demonstrate the computational efficiency of the fully-approximate leverage scores. In light of the inferior performance of SRHT, both in terms of the quality of approximation and also run time, in the subsequent experiments, we will no longer consider SRHT approximation alternative.

Figures 1a to 1c suggest that the upper bound provided in Theorem 13 might be improved by replacing $p-1$ with an appropriate scaled function of $\log(p)$. This observation is numerically investigated in Figure 2. In this figure (which in logarithmic scale), the `MPRE` (18) (in blue) is compared with the right hand side (RHS) of Theorem 13 (in red) as well as the RHS of Theorem 13 with $p-1$ replaced with a scaled $\log(p)$ (in green). These results are in strong agreement with Theorems 5 and 18. Indeed, improving the dependence of the RHS of Theorem 13 on $p$ is an interesting problem, which we intend to address in future works.

Figure 3 exhibits the impact of the data size $n$ and the sample size $s$ on `MPRE` for the `AR(100)` synthetic data. More precisely, this figure demonstrates `MPRE` for values of $n \in \{500K, 1M, 2M\}$ (where, $K$ and $M$ stand for “thousand” and “million”, respectively) and $s \in \{0.001n, 0.01n, 0.1n\}$. Clearly, for each fixed value of $n$, by increasing $s$, `MPRE` decreases. Furthermore, for each fixed ratio of $s/n$, by increasing $n$, s increases and accordingly `MPRE` decreases. It is clear that more data amounts to smaller approximation errors.

### 4.2 PACF: Computational Time and Estimation Accuracy

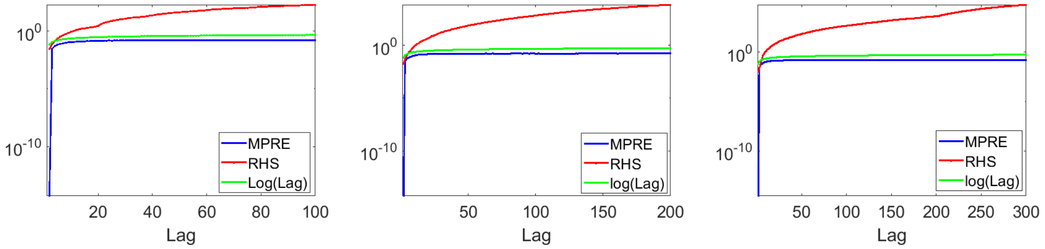
In this section, using the same synthetic data as in Section 4.1, we estimate PACF and fit an `AR` model. More precisely, for each data set, PACF is estimated for a range of lag



(a) AR(20) (b) AR(100) (c) AR(200)

(d) AR(20) (e) AR(100) (f) AR(200)

Figure 1: Figures (a), (b) and (c) correspond to `AR(20)`, `AR(100)`, and `AR(200)` using synthetic data, respectively, and display the `MPRE` (18) versus the lag values $h$ for fully-approximate and the SRHT method. Similarly, Figures (d), (e), and (f) represent the computational time spent, in seconds, to compute the fully-approximate leverage scores (in blue), the SRHT approximation (in magenta), and the exact leverage scores (in red) on `AR(20)`, `AR(100)`, and `AR(200)` using synthetic data, respectively.

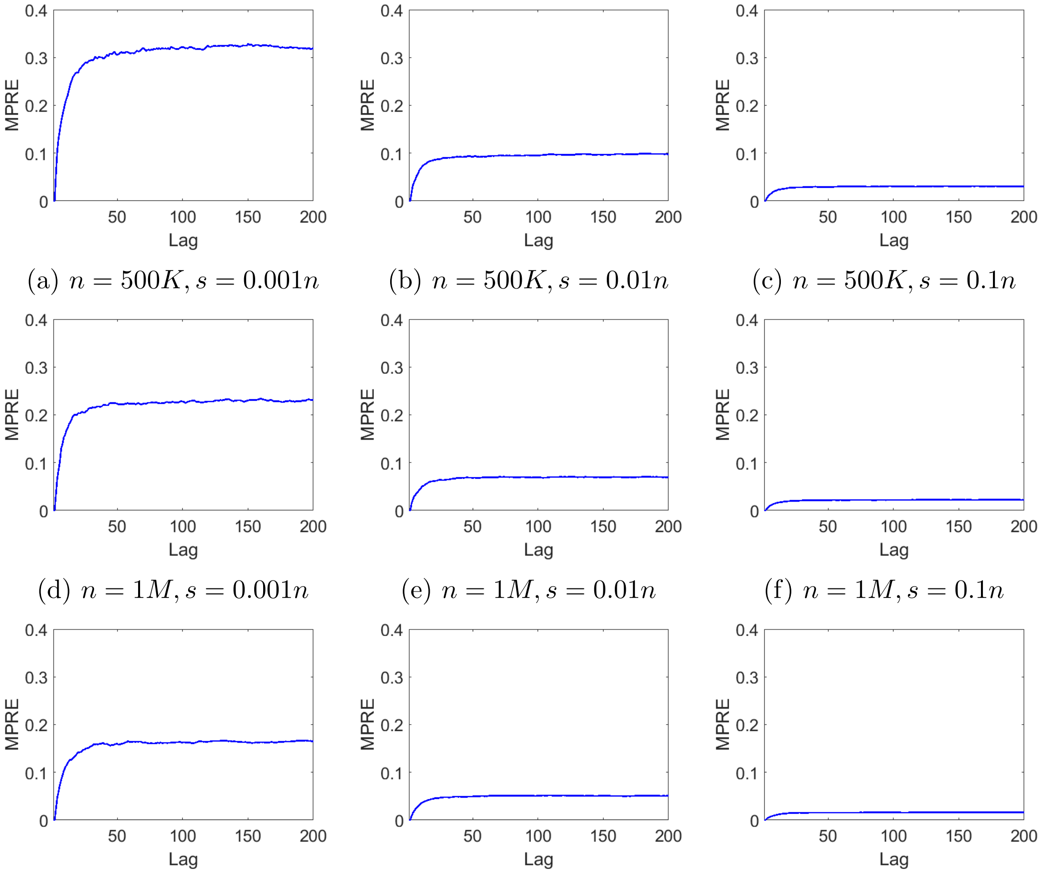


(a) AR(20) (b) AR(100) (c) AR(200)

Figure 2: Figures (a), (b) and (c) correspond to `AR(20)`, `AR(100)`, and `AR(200)` with synthetic data, respectively. Here, we display the `MPRE` (18) (in blue), the RHS of Theorem 13 (in red) and RHS of Theorem 13 with $p - 1$ replaced with a scaled $\log(p)$ (in green).

values $h$, once by solving the corresponding OLS problem with the full-data matrix (called, "exact"), and another time by running the `LSAR` algorithm (Algorithm 1).

The numerical results of these experiments for the three synthetic data sets are displayed in Figure 4. As explained in Section 2.1, the most important application of a PACF plot



(a) $n = 500K, s = 0.001n$ (b) $n = 500K, s = 0.01n$ (c) $n = 500K, s = 0.1n$

(d) $n = 1M, s = 0.001n$ (e) $n = 1M, s = 0.01n$ (f) $n = 1M, s = 0.1n$

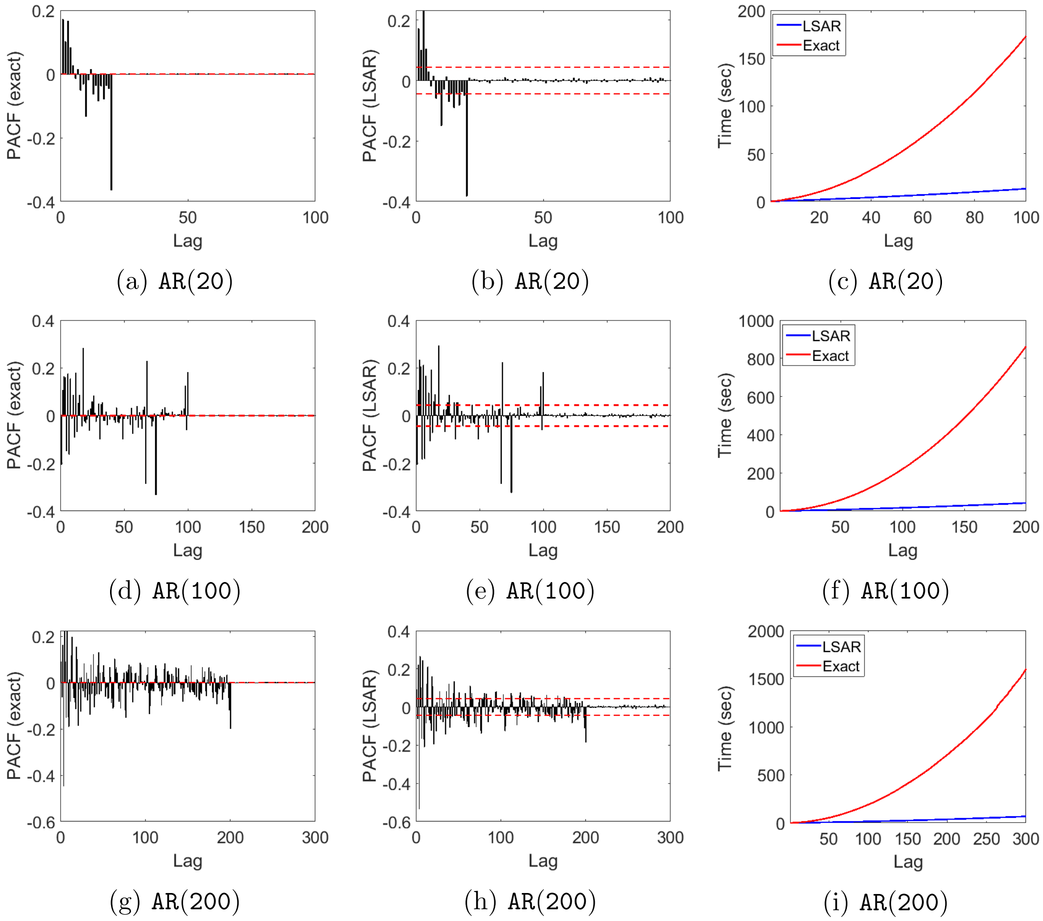
(g) $n = 2M, s = 0.001n$ (h) $n = 2M, s = 0.01n$ (i) $n = 2M, s = 0.1n$

Figure 3: The impact of the data size $n \in \{500K, 1M, 2M\}$ and the sample size $s \in \{0.001n, 0.01n, 0.1n\}$ on MPRE for the AR(100) synthetic data.

is estimating the order $p$ by choosing the largest lag $h$ such that its corresponding PACF bar lies out of the 95% zero-confidence boundary. It is readily seen that Figures 4b, 4e and 4h not only provide the correct estimate of the order $p$ for the generated synthetic data, but also are very close to the exact PACF plots in Figures 4a, 4d and 4g. This is achieved all the while by merely sampling only 0.1% of the rows of the data matrix (i.e., $s = 0.001, n = 2000$). Subsequently, from Figures 4c, 4f and 4i, one can observe a significant difference in the time required for computing PACF exactly as compared with obtaining a high-quality approximation using the LSAR algorithm.

**Remark 19** *Following Theorem 3, finding PACF at each lag requires the solution to the corresponding OLS problem. Hence, to avoid duplication, the computational times of Steps 4-8 of the* LSAR *algorithm are excluded in Figure 1. Indeed, those computational times are considered in Figure 4.*

To show the accuracy of maximum likelihood estimates generated by the LSAR algorithm, the estimates derived by the two scenarios of "full-data matrix" and "reduced data matrix"



(a) `AR(20)` (b) `AR(20)` (c) `AR(20)`

(d) `AR(100)` (e) `AR(100)` (f) `AR(100)`

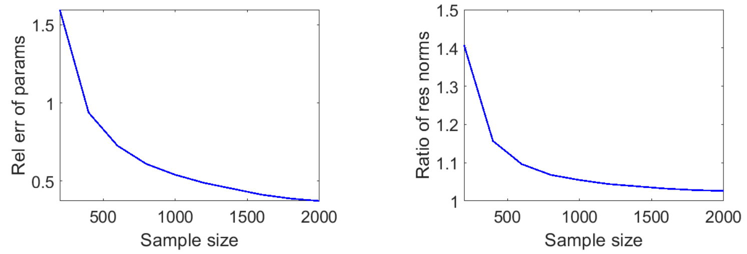
(g) `AR(200)` (h) `AR(200)` (i) `AR(200)`

Figure 4: Figures (a), (b) and (c) corresponding to the `AR(20)` synthetic data, display the exact PACF plot, the PACF plot generated by the `LSAR` algorithm, and the comparison between the computational time of (a) (in red) and (b) (in blue), respectively. Figures (d), (e) and (f) are similar for the `AR(100)` synthetic data; and Figures (g), (h) and (i) are similar for the `AR(200)` synthetic data.

are relatively compared. For this purpose, following notation defined in Sections 2 and 3, let $\boldsymbol{\phi}_{n,p}$ and $\hat{\boldsymbol{\phi}}_{n,p}^{s}$ denote the maximum likelihood estimates of parameters based on the full-data matrix (cf. Equation 2) and reduced sampled data matrix with the sample size of $s$ (cf. Equation 16c), respectively. Accordingly, we define the relative error of parameter estimates by

$$\frac{||\hat{\boldsymbol{\phi}}_{n,p}^{s} - \boldsymbol{\phi}_{n,p}||}{||\boldsymbol{\phi}_{n,p}||}. \tag{19a}$$

Analogously, let $\boldsymbol{r}_{n,p}$ and $\hat{\boldsymbol{r}}_{n,p}^{s}$ be the residuals of estimates based on the full-data matrix (cf. Equation 4) and reduced sampled data matrix with the sample size of $s$ (cf. Equation



(a) (19a) for the `AR(100)` data (b) (19b) for the `AR(100)` data

Figure 5: Figures (a) and (b) display the relative error of parameter estimates (19a) and the ratio of residual norms (19b) for the `AR(100)` synthetic data, respectively, both as a function of sample size $s$.

16b), respectively. The ratio of two residual norms is given by

$$\frac{\left\|\hat{\boldsymbol{r}}_{n,p}^{s}\right\|}{\left\|\boldsymbol{r}_{n,p}\right\|}. \tag{19b}$$

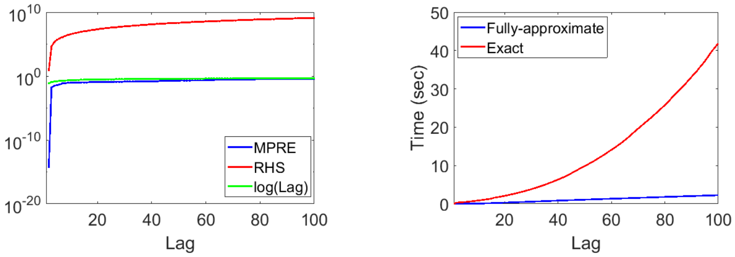
The two ratios (19a) and (19b) are calculated for a range of values of $s \in \{200, 300, \ldots, 1000\}$ by computing the maximum likelihood estimates of the `AR(100)` synthetic data once with the full-data matrix and another time by running the `LSAR` algorithm. Also, the estimates are smoothed out by replicating the `LSAR` algorithm $1,000$ times and taking the average of all estimates. The outcome is displayed in Figure 5. Figure 5a displays the relative errors of parameter estimates (19a) versus the sample size $s$ and Figure 5b shows the ratio of residual norms (19b) versus the sample size $s$.

### 4.3 Real-world Big Time Series: Gas Sensors Data

Huerta et al. (2016) studied the accuracy of electronic nose measurements. They constructed a nose consisting of eight different metal-oxide sensors in addition to humidity and temperature sensors with a wireless communication channel to collect data. The nose monitored airflow for two years in a designated location, and data continuously collected with a rate of two observations per second. In this configuration, a standard energy band model for an $n-$type metal-oxide sensor was used to estimate the changes in air temperature and humidity. Based on their observations, humidity changes and correlated changes of humidity and temperature were the most significant statistical factors in variations of sensor conductivity. The model successfully used for gas discrimination with an R-squared close to 1.

The data is available in the UCI machine learning repository[2]. In our experiment, we use the output of sensor number 8 (column labeled R8 in the data set) as real time series data with $n = 919,438$ observations. The original time series data is heteroscedastic. However, by taking the logarithm of the data and differencing in one lag, it becomes stationary and an `AR(16)` model seems to be a good fit to the transformed data. We run the `LSAR` algorithm

[2] `https://archive.ics.uci.edu/ml/datasets/Gas+sensors+for+home+activity+monitoring`, Accessed on 14 January 2022.



(a) MPRE (b) Computational time

Figure 6: Figure (a) displays the MPRE (18) (in blue), the RHS of Theorem 13 (in red) and the RHS of Theorem 13 with $p-1$ replaced with a scaled $\log(p)$ (in green) for the real gas sensors data, Figure (b) shows the computational time spent, in seconds, to compute the fully-approximate (in blue) and exact (in red) leverage scores for the real gas sensors data.

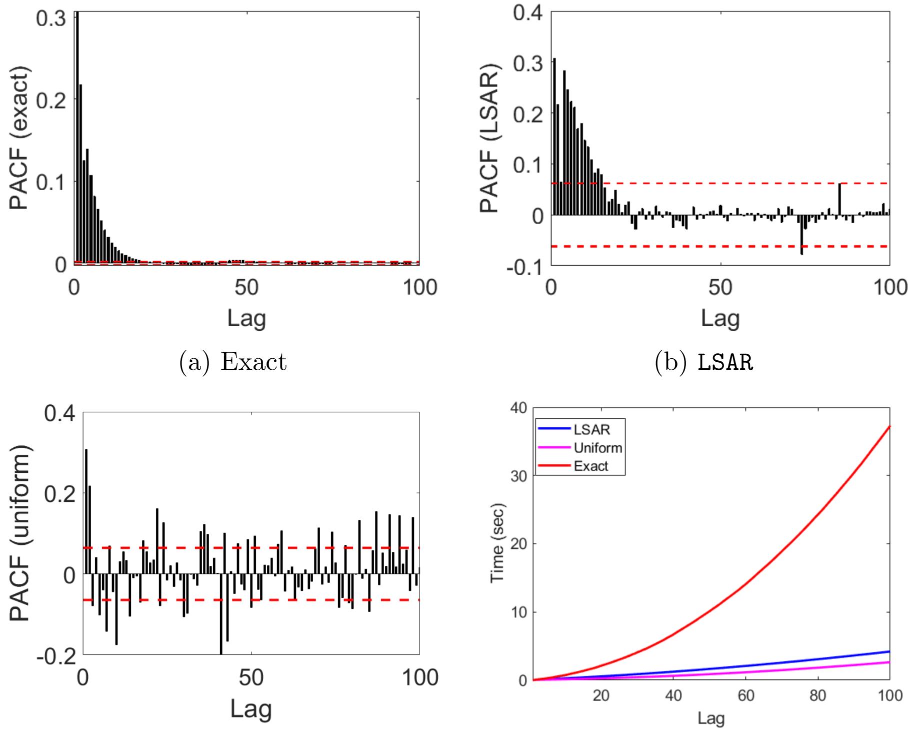
with the initial input parameter $\bar{p} = 100$. Recalling that $\bar{p}$ ($p < \bar{p} \ll n$) is initially set large enough to estimate the order $p$ of the underlying AR model. Also, in all iterations of the LSAR algorithm, we set the sample size $s = 0.001n$. For sake of fairness and completeness, all figures generated for synthetic data in Sections 4.1 and 4.2, are regenerated for the gas sensors data.

Figure 6a shows (in logarithmic scale) the maximum pointwise relative error (18) (in blue) along with the RHS of Theorem 13 (in red) as well as the RHS of Theorem 13 with $p-1$ replaced with a scaled $\log(p)$ (in green). The behavior of these three graphs are very similar to those ones on synthetic data discussed in Section 4.1. Furthermore, Figure 6b reveals analogous computational efficiency in finding the fully-approximate leverage scores comparing with the exact values for the gas sensors data.

**Leverage score sampling versus uniform sampling.** In order to show the efficacy of the LSAR algorithm, we compare the performance of leverage score sampling with naïve uniform sampling in estimating the order as well as parameters of an AR model for the gas sensor data. For the uniform sampling, we modify the LSAR algorithm slightly by removing Step 3 and replacing the uniform distribution $\hat{\pi}_{m,p}(i) = 1/(m-p)$ for $i = 1, \ldots, m-p$ in Step 4.

Figures 7a to 7d demonstrate the PACF plot calculated exactly, the PACF plot approximated with the LSAR algorithm, the PACF plot approximated based on a uniform sampling, and a comparison between the computational times of these three PACF plots, respectively. Similar to Section 4.2, Figure 7d reveals that while Figure 7b can be generated much faster than Figure 7a, they both suggest the same AR model for the gas sensors data. In addition, Figures 7c and 7d divulge that although the uniform sampling is slightly faster than the LSAR algorithm, the PACF plot generated by the former is very poor and far away from the exact plot given in Figure 7a. While both Figures 7a and 7b estimate an AR(16) model for the data, Figure 7c fails to make an appropriate estimate of the order.

Finally, we compare the performance of leverage score sampling with naïve uniform sampling in estimating the parameters of an AR(16) model for the gas sensor data. Figure 8 compares the performance of these two sampling strategies for sample sizes chosen from



(a) Exact

(b) LSAR

(c) Uniform

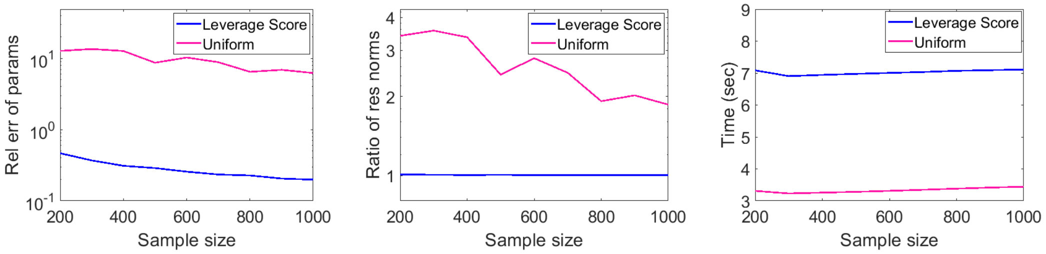
(d) Computational time

Figure 7: Figures (a), (b), (c) and (d) display the exact PACF plot, the PACF plot generated by the LSAR algorithm, the PACF plot approximated based on a uniform sampling, and the comparison between the computational time of (a) (in red), (b) (in blue), and (c) (in pink), respectively.

$s \in \{200, 300, \ldots, 1000\}$. For each sampling scheme and a fixed sample size, the maximum likelihood estimates are smoothed out by replicating the LSAR algorithm $1,000$ times and taking the average of all estimates. Note that in all three Figures 8a to 8c, the blue and red plots correspond with the leverage score and uniform sampling scheme, respectively.

Figure 8a displays the relative errors of parameter estimates (19a) and Figure 8b shows the ratio of residual norms (19b), under the two sampling schemes. Both figures strongly suggest that the leverage score sampling scheme outperforms the uniform sampling scheme. Furthermore, while the output of the former shows stability and almost monotonic convergence, the latter exhibits oscillations and does not show any indication of convergence for such small sample sizes. This observation is consistent with the literature discussed in Section 2.2. Despite the fact uniform sampling can be performed almost for free, Figure 8c shows no significant difference between the computational time of both sampling scheme.

Finally, in our numerical examples, depending on the order of the AR model, the time difference between the exact method and the LSAR algorithm for model fitting vary between 75 to 1600 seconds. In many practical situations, one might need to fit hundreds of such models and make time-sensitive decisions based on the generated forecasts, before new data is provided. One such example is predicting several stock prices in a financial market for



(a) Relative error (19a) (b) Ratio of residuals (19b) (c) Computational time

Figure 8: Figures (a), (b) and (c) display the relative error of parameter estimates (19a), the ratio of residual norms (19b), and the computational time of two sampling schemes based on the leverage scores (blue) and uniform distribution (pink) for the gas sensors data, respectively.

portfolio optimization, while the prices may be updated every few seconds. Another practical example is predicting the meteorology indices for several different purposes, with updates becoming available every few minutes. In these situations, saving a few seconds/minutes in forecasting can be crucial.

## 5. Conclusion

In this paper, we have developed a new approach to fit an AR model to big time series data. Motivated from the literature of RandNLA in dealing with large matrices, we construct a fast and efficient algorithm, called LSAR, to approximate the leverage scores corresponding to the data matrix of an AR model, to estimate the appropriate underlying order, and to find the conditional maximum likelihood estimates of its parameters. Analytical error bounds are developed for such approximations and the worst case running time of the LSAR algorithm is derived. Empirical results on large-scale synthetic as well as big real time series data highly support the theoretical results and reveal the efficacy of this new approach.

For future work, we are mainly interested in developing this approach for a more general ARMA model. However, unlike AR, the (conditional) log-likelihood function for ARMA is a complicated non-linear function such that (C)MLEs cannot be derived analytically. Thus, it may require to exploit not only RandNLA techniques, but also modern optimization algorithms in big data regime to develop an efficient leverage score sampling scheme for ARMA models.

## Acknowledgments

The authors thank Rob J. Hyndman, Vivak Patel and David P. Woodruff for suggesting some improvements. Ali Eshragh wishes to acknowledge the support from the Australian Research Council Centre of Excellence for Mathematical and Statistical Frontiers (ACEMS) and the International Computer Science Institute at the University of California, Berkeley. Fred Roosta was partially supported by DARPA, ACEMS, and ARC DECRA Award

(DE180100923). Part of this work was done while Fred Roosta and Michael W. Mahoney were visiting the Simons Institute for the Theory of Computing. Michael W. Mahoney would also like to acknowledge ARO, DARPA, NSF, and ONR for providing partial support of this work.

## Appendix A. Technical Lemmas and Proofs

### A.1 Proof of Theorem 7

We first present Theorem 20 which is used in the proof of Theorem 7.

**Lemma 20 (Golub and Van Loan 1983)** *Consider the* $2 \times 2$ *block matrix*

$$M = \begin{pmatrix} c & \boldsymbol{b}^\intercal \\ \boldsymbol{b} & \boldsymbol{A} \end{pmatrix},$$

*where* $\boldsymbol{A}$*,* $\boldsymbol{b}$*, and* $c$ *are an* $m \times m$ *matrix, an* $m \times 1$ *vector and a scalar, respectively. If* $\boldsymbol{A}$ *is invariable, the inverse of matrix* $M$ *exists an can be calculated as follows*

$$M^{-1} = \frac{1}{k} \begin{pmatrix} 1 & -\boldsymbol{b}^\intercal \boldsymbol{A}^{-1} \\ -\boldsymbol{A}^{-1}\boldsymbol{b} & k\boldsymbol{A}^{-1} + \boldsymbol{A}^{-1}\boldsymbol{b}\boldsymbol{b}^\intercal \boldsymbol{A}^{-1} \end{pmatrix},$$

*where* $k = c - \boldsymbol{b}^\intercal \boldsymbol{A}^{-1}\boldsymbol{b}$*.*

Proof of Theorem 7

**Proof** For $p = 1$, computing the leverage score trivially boils down to normalizing the data vector. For $p \geq 2$, the data matrix is given by

$$\boldsymbol{X}_{n,p} = \begin{pmatrix} \boldsymbol{y}_{n-1,p-1} & \boldsymbol{X}_{n-1,p-1} \end{pmatrix}.$$

So, we have

$$\boldsymbol{X}_{n,p}^\intercal \boldsymbol{X}_{n,p} = \begin{pmatrix} \boldsymbol{y}_{n-1,p-1}^\intercal \boldsymbol{y}_{n-1,p-1} & \boldsymbol{y}_{n-1,p-1}^\intercal \boldsymbol{X}_{n-1,p-1} \\ \boldsymbol{X}_{n-1,p-1}^\intercal \boldsymbol{y}_{n-1,p-1} & \boldsymbol{X}_{n-1,p-1}^\intercal \boldsymbol{X}_{n-1,p-1} \end{pmatrix}.$$

For sake of simplicity, let us define

$$\boldsymbol{W}_{n,p} := \boldsymbol{X}_{n,p}^\intercal \boldsymbol{X}_{n,p}.$$

Following Theorem 20, the inverse of matrix $\boldsymbol{W}_{n,p}$ is given by

$$\boldsymbol{W}_{n,p}^{-1} = \frac{1}{u_{n,p}} \begin{pmatrix} 1 & -\boldsymbol{\phi}_{n-1,p-1}^\intercal \\ -\boldsymbol{\phi}_{n-1,p-1} & u_{n,p}\boldsymbol{W}_{n-1,p-1}^{-1} + \boldsymbol{\phi}_{n-1,p-1}\boldsymbol{\phi}_{n-1,p-1}^\intercal \end{pmatrix},$$

where

$$u_{n,p} := \boldsymbol{y}_{n-1,p-1}^\intercal \boldsymbol{y}_{n-1,p-1} - \boldsymbol{y}_{n-1,p-1}^\intercal \boldsymbol{X}_{n-1,p-1} \boldsymbol{W}_{n-1,p-1}^{-1} \boldsymbol{X}_{n-1,p-1}^\intercal \boldsymbol{y}_{n-1,p-1}$$

$$= \boldsymbol{y}^{\intercal}_{n-1,p-1}\boldsymbol{y}_{n-1,p-1} - \boldsymbol{y}^{\intercal}_{n-1,p-1}\boldsymbol{X}_{n-1,p-1}\boldsymbol{\phi}_{n-1,p-1}.$$

It is readily seen that

$$\begin{aligned}
u_{n,p} &:= \boldsymbol{y}^{\intercal}_{n-1,p-1}\boldsymbol{y}_{n-1,p-1} - 2\boldsymbol{y}^{\intercal}_{n-1,p-1}\boldsymbol{X}_{n-1,p-1}\boldsymbol{\phi}_{n-1,p-1} + \boldsymbol{y}^{\intercal}_{n-1,p-1}\boldsymbol{X}_{n-1,p-1}\boldsymbol{\phi}_{n-1,p-1}\\
&= \boldsymbol{y}^{\intercal}_{n-1,p-1}\boldsymbol{y}_{n-1,p-1} - 2\boldsymbol{y}^{\intercal}_{n-1,p-1}\boldsymbol{X}_{n-1,p-1}\boldsymbol{\phi}_{n-1,p-1}\\
&\quad + \boldsymbol{y}^{\intercal}_{n-1,p-1}\boldsymbol{X}_{n-1,p-1}\boldsymbol{W}^{-1}_{n-1,p-1}\boldsymbol{W}_{n-1,p-1}\boldsymbol{\phi}_{n-1,p-1}\\
&= \boldsymbol{y}^{\intercal}_{n-1,p-1}\boldsymbol{y}_{n-1,p-1} - 2\boldsymbol{y}^{\intercal}_{n-1,p-1}\boldsymbol{X}_{n-1,p-1}\boldsymbol{\phi}_{n-1,p-1} + \boldsymbol{\phi}^{\intercal}_{n-1,p-1}\boldsymbol{W}_{n-1,p-1}\boldsymbol{\phi}_{n-1,p-1}\\
&= \boldsymbol{y}^{\intercal}_{n-1,p-1}\boldsymbol{y}_{n-1,p-1} - 2\boldsymbol{y}^{\intercal}_{n-1,p-1}\boldsymbol{X}_{n-1,p-1}\boldsymbol{\phi}_{n-1,p-1}\\
&\quad + \boldsymbol{\phi}^{\intercal}_{n-1,p-1}\boldsymbol{X}^{\intercal}_{n-1,p-1}\boldsymbol{X}_{n-1,p-1}\boldsymbol{\phi}_{n-1,p-1}\\
&= \left\|\boldsymbol{y}_{n-1,p-1} - \boldsymbol{X}_{n-1,p-1}\boldsymbol{\phi}_{n-1,p-1}\right\|^2\\
&= \left\|\boldsymbol{r}_{n-1,p-1}\right\|^2.
\end{aligned}$$

The $i^{th}$ leverage score is given by

$$\begin{aligned}
\ell_{n,p}(i) &= \boldsymbol{X}^{\intercal}_{n,p}(i,:)\boldsymbol{W}^{-1}_{n,p}\boldsymbol{X}_{n,p}(i,:)\\
&= \begin{bmatrix} y_{i+p-1} & \boldsymbol{X}^{\intercal}_{n-1,p-1}(i,:)\end{bmatrix}\boldsymbol{W}^{-1}_{n,p}\begin{bmatrix} y_{i+p-1}\\ \boldsymbol{X}_{n-1,p-1}(i,:)\end{bmatrix}\\
&= \frac{1}{\left\|\boldsymbol{r}_{n-1,p-1}\right\|^2}\Big[y_{i+p-1} - \boldsymbol{X}^{\intercal}_{n-1,p-1}(i,:)\boldsymbol{\phi}_{n-1,p-1}\\
&\quad -y_{i+p-1}\boldsymbol{\phi}^{\intercal}_{n-1,p-1} + \boldsymbol{X}^{\intercal}_{n-1,p-1}(i,:)(\left\|\boldsymbol{r}_{n-1,p-1}\right\|^2\boldsymbol{W}^{-1}_{n-1,p-1} + \boldsymbol{\phi}_{n-1,p-1}\boldsymbol{\phi}^{\intercal}_{n-1,p-1})\Big]\\
&\quad \times \begin{bmatrix} y_{i+p-1}\\ \boldsymbol{X}_{n-1,p-1}(i,:)\end{bmatrix}\\
&= \frac{1}{\left\|\boldsymbol{r}_{n-1,p-1}\right\|^2}\big(y^2_{i+p-1} - \boldsymbol{X}^{\intercal}_{n-1,p-1}(i,:)\boldsymbol{\phi}_{n-1,p-1}y_{i+p-1} - y_{i+p-1}\boldsymbol{\phi}^{\intercal}_{n-1,p-1}\boldsymbol{X}_{n-1,p-1}(i,:)\\
&\quad +\boldsymbol{X}^{\intercal}_{n-1,p-1}(i,:)(\left\|\boldsymbol{r}_{n-1,p-1}\right\|^2\boldsymbol{W}^{-1}_{n-1,p-1} + \boldsymbol{\phi}_{n-1,p-1}\boldsymbol{\phi}^{\intercal}_{n-1,p-1})\boldsymbol{X}_{n-1,p-1}(i,:)\big)\\
&= \boldsymbol{X}^{\intercal}_{n-1,p-1}(i,:)\boldsymbol{W}^{-1}_{n-1,p-1}\boldsymbol{X}_{n-1,p-1}(i,:)\\
&\quad + \frac{1}{\left\|\boldsymbol{r}_{n-1,p-1}\right\|^2}\left(y^2_{i+p-1} - 2y_{i+p-1}\boldsymbol{X}^{\intercal}_{n-1,p-1}(i,:)\boldsymbol{\phi}_{n-1,p-1} + (\boldsymbol{X}^{\intercal}_{n-1,p-1}(i,:)\boldsymbol{\phi}_{n-1,p-1})^2\right)\\
&= \ell_{n-1,p-1}(i) + \frac{1}{\left\|\boldsymbol{r}_{n-1,p-1}\right\|^2}\left\|y_{i+p-1} - \boldsymbol{X}^{\intercal}_{n-1,p-1}(i,:)\boldsymbol{\phi}_{n-1,p-1}\right\|^2\\
&= \ell_{n-1,p-1}(i) + \frac{\boldsymbol{r}^2_{n-1,p-1}(i)}{\left\|\boldsymbol{r}_{n-1,p-1}\right\|^2}.
\end{aligned}$$

■

### A.2 Theorem 21 and Its Proof

**Theorem 21 (Relative Errors for Quasi-approximate Leverage Scores)** *For the quasi-approximate leverage scores, we have with probability at least* $1-\delta$,

$$\frac{|\ell_{n,p}(i)-\tilde{\ell}_{n,p}(i)|}{\ell_{n,p}(i)} \leq \left(1+3\eta_{n-1,p-1}\kappa^2(\boldsymbol{X}_{n,p})\right)\sqrt{\varepsilon}, \quad \textit{for } i=1,\ldots,n-p,$$

*recalling that* $\eta_{n,p}, \kappa(\boldsymbol{X}_{n,p})$*, and* $\varepsilon$ *are as in Theorem 9.*

In order to prove Theorem 21, we first introduce the following lemmas and corollary.

**Lemma 22** *The leverage scores of an* `AR(p)` *model for* $p \geq 1$*, are given by*

$$\ell_{n,p}(i) = \min_{\boldsymbol{z}\in\mathbb{R}^{n-p}} \left\{\|\boldsymbol{z}\|^2 \mid \boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{z} = \boldsymbol{X}_{n,p}(i,:)\right\}, \quad \textit{for } i=1,\ldots,n-p,$$

*where* $\boldsymbol{X}_{n,p}$ *is the data matrix of the* `AR(p)` *model defined in* (3).

**Proof** We prove this lemma by using Lagrangian multipliers. Define the function

$$h(\boldsymbol{z},\boldsymbol{\lambda}) := \frac{1}{2}\boldsymbol{z}^{\intercal}\boldsymbol{z} - \boldsymbol{\lambda}^{\intercal}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{z} - \boldsymbol{X}_{n,p}(i,:)).$$

By taking the first derivative with respect to the vector $\boldsymbol{z}$ and setting equal to zero, we have,

$$\frac{\partial h(\boldsymbol{z},\boldsymbol{\lambda})}{\partial \boldsymbol{z}} = \boldsymbol{z} - \boldsymbol{X}_{n,p}\boldsymbol{\lambda} \;=\; 0 \;\Rightarrow\; \boldsymbol{z}^{\star} \;=\; \boldsymbol{X}_{n,p}\boldsymbol{\lambda}^{\star}.$$

Now, by multiplying both sides by $\boldsymbol{X}_{n,p}^{\intercal}$, we obtain,

$$\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{z}^{\star} = \boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p}\boldsymbol{\lambda}^{\star},$$

simplified to

$$\boldsymbol{X}_{n,p}(i,:) = \boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p}\boldsymbol{\lambda}^{\star}.$$

This implies that,

$$\boldsymbol{\lambda}^{\star} = (\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})^{-1}\boldsymbol{X}_{n,p}(i,:).$$

Thus,

$$\boldsymbol{z}^{\star} = \boldsymbol{X}_{n,p}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})^{-1}\boldsymbol{X}_{n,p}(i,:).$$

The square of the norm of $\boldsymbol{z}^{\star}$ is equal to

$$\begin{aligned}\|\boldsymbol{z}^{\star}\|^2 &= \left(\boldsymbol{X}_{n,p}^{\intercal}(i,:)(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})^{-1}\boldsymbol{X}_{n,p}^{\intercal}\right)\left(\boldsymbol{X}_{n,p}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})^{-1}\boldsymbol{X}_{n,p}(i,:)\right)\\ &= \boldsymbol{X}_{n,p}^{\intercal}(i,:)(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})^{-1}\boldsymbol{X}_{n,p}(i,:)\\ &= \ell_{n,p}(i).\end{aligned}$$

■

**Lemma 23** *For an* $\texttt{AR(p)}$ *model with* $p \geq 1$*, we have*

$$\|\boldsymbol{X}_{n,p}(i,:)\| \leq \|\boldsymbol{X}_{n,p}\| \sqrt{\ell_{n,p}(i)}, \quad \textit{for } i = 1, \ldots, n-p,$$

*where* $\boldsymbol{X}_{n,p}$ *and* $\ell_{n,p}(i)$ *are defined, respectively, in* (3) *and Theorem 6.*

**Proof** From Theorem 22 we have,

$$\begin{aligned}\|\boldsymbol{X}_{n,p}(i,:)\| &= \left\|\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{z}^{\star}\right\| \\ &\leq \left\|\boldsymbol{X}_{n,p}^{\intercal}\right\| \|\boldsymbol{z}^{\star}\| \\ &= \|\boldsymbol{X}_{n,p}\| \sqrt{\ell_{n,p}(i)}.\end{aligned}$$

■

**Lemma 24** *For an* $\texttt{AR(p)}$ *model with* $p \geq 1$*, we have*

$$|\boldsymbol{r}_{n,p}(i) - \tilde{\boldsymbol{r}}_{n,p}(i)| \leq \sqrt{\varepsilon}\eta_{n,p} \|\boldsymbol{\phi}_{n,p}\| \|\boldsymbol{X}_{n,p}\| \sqrt{\ell_{n,p}(i)}, \quad \textit{for } i = 1, \ldots, n-p,$$

*where* $\boldsymbol{r}_{n,p}, \tilde{\boldsymbol{r}}_{n,p}, \eta_{n,p}, \boldsymbol{\phi}_{n,p}, \boldsymbol{X}_{n,p}$*, and* $\ell_{n,p}(i)$ *are defined respectively in* (4)*,* (13b)*,* (14c)*,* (2)*,* (3)*, and Theorem 6 and* $\varepsilon$ *is the error in* (14a)*.*

**Proof** From (14b) and the definition of $l_2$ norm, we have

$$\begin{aligned}\left\langle \frac{\boldsymbol{X}_{n,p}(i,:)}{\|\boldsymbol{X}_{n,p}(i,:)\|}, (\boldsymbol{\phi}_{n,p} - \tilde{\boldsymbol{\phi}}_{n,p}) \right\rangle &\leq \left\|\boldsymbol{\phi}_{n,p} - \tilde{\boldsymbol{\phi}}_{n,p}\right\| \\ &\leq \sqrt{\varepsilon}\eta_{n,p} \|\boldsymbol{\phi}_{n,p}\|.\end{aligned}$$

So, we have

$$\boldsymbol{X}_{n,p}^{\intercal}(i,:)\boldsymbol{\phi}_{n,p} - \boldsymbol{X}_{n,p}^{\intercal}(i,:)\tilde{\boldsymbol{\phi}}_{n,p} \leq \sqrt{\varepsilon}\eta_{n,p} \|\boldsymbol{\phi}_{n,p}\| \|\boldsymbol{X}_{n,p}(i,:)\|.$$

Now by adding and subtracting $y_{i+p}$ on the left hand side, we yield

$$\left(y_{i+p} - \boldsymbol{X}_{n,p}^{\intercal}(i,:)\tilde{\boldsymbol{\phi}}_{n,p}\right) - \left(y_{i+p} - \boldsymbol{X}_{n,p}^{\intercal}(i,:)\boldsymbol{\phi}_{n,p}\right) \leq \sqrt{\varepsilon}\eta_{n,p} \|\boldsymbol{\phi}_{n,p}\| \|\boldsymbol{X}_{n,p}(i,:)\|,$$

implying that,

$$\tilde{\boldsymbol{r}}_{n,p}(i) - \boldsymbol{r}_{n,p}(i) \leq \sqrt{\varepsilon}\eta_{n,p} \|\boldsymbol{\phi}_{n,p}\| \|\boldsymbol{X}_{n,p}(i,:)\|.$$

As analogously we can construct a similar inequality for $\boldsymbol{r}_{n,p}(i) - \tilde{\boldsymbol{r}}_{n,p}(i)$, we have that

$$|\boldsymbol{r}_{n,p}(i) - \tilde{\boldsymbol{r}}_{n,p}(i)| \leq \sqrt{\varepsilon}\eta_{n,p} \|\boldsymbol{\phi}_{n,p}\| \|\boldsymbol{X}_{n,p}(i,:)\|.$$

Now, by using Theorem 23, we obtain

$$|\boldsymbol{r}_{n,p}(i) - \tilde{\boldsymbol{r}}_{n,p}(i)| \leq \sqrt{\varepsilon}\eta_{n,p} \|\boldsymbol{\phi}_{n,p}\| \|\boldsymbol{X}_{n,p}\| \sqrt{\ell_{n,p}(i)}.$$

■

**Lemma 25** *Let $\{y_1, \ldots, y_n\}$ be a time series data. For $i = 1, \ldots, n-p$, we have*

$$|\boldsymbol{r}_{n-1,p-1}(i)| \leq \sqrt{\|\boldsymbol{\phi}_{n-1,p-1}\|^2 + 1}\, \|\boldsymbol{X}_{n,p}\| \sqrt{\ell_{n,p}(i)}, \tag{20a}$$

$$|\tilde{\boldsymbol{r}}_{n-1,p-1}(i)| \leq \sqrt{\left\|\tilde{\boldsymbol{\phi}}_{n-1,p-1}\right\|^2 + 1}\, \|\boldsymbol{X}_{n,p}\| \sqrt{\ell_{n,p}(i)}, \tag{20b}$$

*where $\boldsymbol{r}_{n,p}, \tilde{\boldsymbol{r}}_{n,p}, \boldsymbol{\phi}_{n,p}, \tilde{\boldsymbol{\phi}}_{n,p}, \boldsymbol{X}_{n,p}$, and $\ell_{n,p}(i)$ are defined respectively in* (4), (13b), (2), (13a), (3), *and Theorem 6.*

**Proof** The left hand side of (20a) can be written as below:

$$\begin{aligned}
|\boldsymbol{r}_{n-1,p-1}(i)| &= |y_{i+p-1} - \boldsymbol{X}_{n-1,p-1}^{\intercal}(i,:)\boldsymbol{\phi}_{n-1,p-1}| \\
&= |\begin{bmatrix} y_{i+p-1} & \boldsymbol{X}_{n-1,p-1}^{\intercal}(i,:) \end{bmatrix} \begin{bmatrix} 1 & -\boldsymbol{\phi}_{\mathbf{n-1,p-1}}^{\intercal} \end{bmatrix}^{\intercal} | \\
&= |\boldsymbol{X}_{n,p}^{\intercal}(i,:) \begin{bmatrix} 1 & -\boldsymbol{\phi}_{\mathbf{n-1,p-1}}^{\intercal} \end{bmatrix}^{\intercal} | \\
&= \sqrt{\|\boldsymbol{\phi}_{n-1,p-1}\|^2 + 1} \left| \boldsymbol{X}_{n,p}^{\intercal}(i,:) \frac{\begin{bmatrix} 1 & -\boldsymbol{\phi}_{\mathbf{n-1,p-1}}^{\intercal} \end{bmatrix}^{\intercal}}{\sqrt{\|\boldsymbol{\phi}_{n-1,p-1}\|^2 + 1}} \right| \\
&\leq \sqrt{\|\boldsymbol{\phi}_{n-1,p-1}\|^2 + 1}\, \|\boldsymbol{X}_{n,p}(i,:)\|\,.
\end{aligned}$$

Now, by using Theorem 23, we obtain,

$$|\boldsymbol{r}_{n-1,p-1}(i)| \leq \sqrt{\|\boldsymbol{\phi}_{n-1,p-1}\|^2 + 1}\, \|\boldsymbol{X}_{n,p}\| \sqrt{\ell_{n,p}(i)}.$$

Inequality (20b) can be proved analogously. ■

**Lemma 26** *Let $\{y_1, \ldots, y_n\}$ be a time series data. We have,*

$$\frac{(\boldsymbol{r}_{n-1,p-1}(i))^2}{\|\boldsymbol{r}_{n-1,p-1}\|^2} \leq \ell_{n,p}(i), \quad \textit{for } i = 1, \ldots, n-p,$$

*where $\boldsymbol{r}_{n,p}$ and $\ell_{n,p}(i)$ are defined respectively in* (4) *and Theorem 6.*

**Proof** Since the leverage score is a non-negative valued function, the proof is directly achieved from Theorem 7. ■

**Lemma 27** *Let $\{y_1, \ldots, y_n\}$ be a time series data. We have*

$$\|\boldsymbol{r}_{n-1,p-1}\| \geq \sqrt{\lambda_{\min}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p}) \left(\|\boldsymbol{\phi}_{n-1,p-1}\|^2 + 1\right)}, \tag{21a}$$

$$\|\tilde{\boldsymbol{r}}_{n-1,p-1}\| \geq \sqrt{\lambda_{\min}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p}) \left(\left\|\tilde{\boldsymbol{\phi}}_{n-1,p-1}\right\|^2 + 1\right)}, \tag{21b}$$

*where $\boldsymbol{r}_{n,p}, \tilde{\boldsymbol{r}}_{n,p}, \boldsymbol{\phi}_{n,p}, \tilde{\boldsymbol{\phi}}_{n,p}$, and $\boldsymbol{X}_{n,p}$ are defined respectively in* (4), (13b), (2), (13a), *and* (3) *and $\lambda_{\min}(.)$ denotes the minimum eigenvalue.*

**Proof** Consider (21a), by definition, we have

$$\begin{aligned}\|\boldsymbol{r}_{n-1,p-1}\| &= \|\boldsymbol{y}_{n-1,p-1} - \boldsymbol{X}_{n-1,p-1}\boldsymbol{\phi}_{n-1,p-1}\| \\ &= \left\|\begin{pmatrix}\boldsymbol{y}_{n-1,p-1} & \boldsymbol{X}_{n-1,p-1}\end{pmatrix}\begin{bmatrix}1 & -\boldsymbol{\phi}_{n-1,p-1}^{\intercal}\end{bmatrix}^{\intercal}\right\| \\ &= \left\|\boldsymbol{X}_{n,p}\begin{bmatrix}1 & -\boldsymbol{\phi}_{n-1,p-1}^{\intercal}\end{bmatrix}^{\intercal}\right\| \\ &= \sqrt{\begin{bmatrix}1 & -\boldsymbol{\phi}_{n-1,p-1}^{\intercal}\end{bmatrix}\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p}\begin{bmatrix}1 & -\boldsymbol{\phi}_{n-1,p-1}^{\intercal}\end{bmatrix}^{\intercal}} \\ &\geq \sqrt{\lambda_{\min}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})}\,\left\|\begin{bmatrix}1 & -\boldsymbol{\phi}_{n-1,p-1}^{\intercal}\end{bmatrix}\right\| \\ &= \sqrt{\lambda_{\min}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})\left(\|\boldsymbol{\phi}_{n-1,p-1}\|^2 + 1\right)}.\end{aligned}$$

Inequality (21b) is proved analogously. ■

**Lemma 28** *For any positive integer numbers $1 < p < n$, we have*

$$\kappa(\boldsymbol{X}_{n-1,p-1}) \leq \kappa(\boldsymbol{X}_{n,p}),$$

*where $\boldsymbol{X}_{n,p}$ is defined in* (3) *an $\kappa(.)$ denotes the condition number.*

**Proof** It is readily seen that the matrix $\boldsymbol{X}_{n,p}$ can be written in the form of

$$\boldsymbol{X}_{n,p} = \begin{pmatrix}\boldsymbol{y}_{n,p} & \boldsymbol{X}_{n-1,p-1}\end{pmatrix}.$$

On the other hand, by definition, we know that

$$\lambda_{\max}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p}) = \sup_{\|\boldsymbol{\nu}\|\leq 1} \boldsymbol{\nu}^{\intercal}\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p}\boldsymbol{\nu}.$$

Let $\boldsymbol{u}$ be a unit vector corresponding to the maximum eigenvalue $\lambda_{\max}(\boldsymbol{X}_{n-1,p-1}^{\intercal}\boldsymbol{X}_{n-1,p-1})$ and construct the vector

$$\bar{\boldsymbol{u}} := \begin{bmatrix}0 & \boldsymbol{u}^{\intercal}\end{bmatrix}^{\intercal}.$$

We have

$$\begin{aligned}\lambda_{\max}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p}) &\geq \bar{\boldsymbol{u}}^{\intercal}\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p}\bar{\boldsymbol{u}} \\ &= \boldsymbol{u}^{\intercal}\boldsymbol{X}_{n-1,p-1}^{\intercal}\boldsymbol{X}_{n-1,p-1}\boldsymbol{u} \\ &= \lambda_{\max}(\boldsymbol{X}_{n-1,p-1}^{\intercal}\boldsymbol{X}_{n-1,p-1}).\end{aligned}$$

Analogously, one can show that $\lambda_{\min}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p}) \leq \lambda_{\min}(\boldsymbol{X}_{n-1,p-1}^{\intercal}\boldsymbol{X}_{n-1,p-1})$. Thus, we have

$$\kappa(\boldsymbol{X}_{n,p}) = \sqrt{\frac{\lambda_{\max}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})}{\lambda_{\min}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})}}$$

$$\begin{aligned}&\geq \sqrt{\frac{\lambda_{\max}(\boldsymbol{X}_{n-1,p-1}^{\intercal}\boldsymbol{X}_{n-1,p-1})}{\lambda_{\min}(\boldsymbol{X}_{n-1,p-1}^{\intercal}\boldsymbol{X}_{n-1,p-1})}}\\ &= \kappa(\boldsymbol{X}_{n-1,p-1}).\end{aligned}$$

■

**Corollary 29** *For any positive integer numbers $1 < p < n$, we have*

$$\|\boldsymbol{X}_{n-1,p-1}\| \leq \|\boldsymbol{X}_{n,p}\|,$$

*where $\boldsymbol{X}_{n,p}$ is defined in* (3)*.*

**Proof** Since $\lambda_{\max}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p}) = \|\boldsymbol{X}_{n,p}\|^2$, this inequality is directly derived from the proof of Theorem 28. ■

Proof of Theorem 21

**Proof** By using Theorems 7 and 9, we have

$$\begin{aligned}|\ell_{n,p}(i) - \tilde{\ell}_{n,p}(i)| &= \left|\ell_{n-1,p-1}(i) + \frac{(\boldsymbol{r}_{n-1,p-1}(i))^2}{\|\boldsymbol{r}_{n-1,p-1}\|^2} - \ell_{n-1,p-1}(i) - \frac{(\tilde{\boldsymbol{r}}_{n-1,p-1}(i))^2}{\|\tilde{\boldsymbol{r}}_{n-1,p-1}\|^2}\right|\\ &= \left|\frac{(\boldsymbol{r}_{n-1,p-1}(i))^2}{\|\boldsymbol{r}_{n-1,p-1}\|^2} - \frac{(\boldsymbol{r}_{n-1,p-1}(i))^2}{\|\tilde{\boldsymbol{r}}_{n-1,p-1}\|^2} + \frac{(\boldsymbol{r}_{n-1,p-1}(i))^2}{\|\tilde{\boldsymbol{r}}_{n-1,p-1}\|^2} - \frac{(\tilde{\boldsymbol{r}}_{n-1,p-1}(i))^2}{\|\tilde{\boldsymbol{r}}_{n-1,p-1}\|^2}\right|\\ &\leq \left|\frac{(\boldsymbol{r}_{n-1,p-1}(i))^2}{\|\boldsymbol{r}_{n-1,p-1}\|^2} - \frac{(\boldsymbol{r}_{n-1,p-1}(i))^2}{\|\tilde{\boldsymbol{r}}_{n-1,p-1}\|^2}\right| + \left|\frac{(\boldsymbol{r}_{n-1,p-1}(i))^2}{\|\tilde{\boldsymbol{r}}_{n-1,p-1}\|^2} - \frac{(\tilde{\boldsymbol{r}}_{n-1,p-1}(i))^2}{\|\tilde{\boldsymbol{r}}_{n-1,p-1}\|^2}\right|\\ &\leq (\boldsymbol{r}_{n-1,p-1}(i))^2\left|\frac{1}{\|\boldsymbol{r}_{n-1,p-1}\|^2} - \frac{1}{\|\tilde{\boldsymbol{r}}_{n-1,p-1}\|^2}\right|\\ &\quad + \frac{1}{\|\tilde{\boldsymbol{r}}_{n-1,p-1}\|^2}\left|(\boldsymbol{r}_{n-1,p-1}(i))^2 - (\tilde{\boldsymbol{r}}_{n-1,p-1}(i))^2\right|\\ &\leq (\boldsymbol{r}_{n-1,p-1}(i))^2\left|\frac{1}{\|\boldsymbol{r}_{n-1,p-1}\|^2} - \frac{1}{(1+\varepsilon)^2\|\boldsymbol{r}_{n-1,p-1}\|^2}\right|\\ &\quad + \frac{1}{\|\boldsymbol{r}_{n-1,p-1}\|^2}\left|(\boldsymbol{r}_{n-1,p-1}(i) - (\tilde{\boldsymbol{r}}_{n-1,p-1})(i))(\boldsymbol{r}_{n-1,p-1}(i) + (\tilde{\boldsymbol{r}}_{n-1,p-1})(i))\right|\\ &\leq \frac{\varepsilon^2+2\varepsilon}{(1+\varepsilon)^2}\frac{(\boldsymbol{r}_{n-1,p-1}(i))^2}{\|\boldsymbol{r}_{n-1,p-1}\|^2}\\ &\quad + \frac{1}{\|\boldsymbol{r}_{n-1,p-1}\|^2}\left|\boldsymbol{r}_{n-1,p-1}(i) - (\tilde{\boldsymbol{r}}_{n-1,p-1})(i)\right|\\ &\qquad \times \left(|\boldsymbol{r}_{n-1,p-1}(i)| + |(\tilde{\boldsymbol{r}}_{n-1,p-1})(i)|\right).\end{aligned}$$

Now, from Theorems 24 to 26, we have

$$\begin{aligned}
|\ell_{n,p}(i) - \tilde{\ell}_{n,p}(i)| &\leq \frac{\varepsilon^2 + 2\varepsilon}{(1+\varepsilon)^2}\ell_{n,p}(i) \\
&\quad + \frac{1}{\|\boldsymbol{r}_{n-1,p-1}\|^2}\left(\sqrt{\varepsilon}\eta_{n-1,p-1}\|\boldsymbol{\phi}_{n-1,p-1}\|\|\boldsymbol{X}_{n-1,p-1}\|\sqrt{\ell_{n-1,p-1}(i)}\right) \\
&\qquad \times\left(\sqrt{\|\boldsymbol{\phi}_{n-1,p-1}\|^2+1}\,\|\boldsymbol{X}_{n,p}\|\sqrt{\ell_{n,p}(i)}\right. \\
&\qquad\quad \left. + \sqrt{\left\|\tilde{\boldsymbol{\phi}}_{n-1,p-1}\right\|^2+1}\,\|\boldsymbol{X}_{n,p}\|\sqrt{\ell_{n,p}(i)}\right) \\
&\leq \left(\frac{\sqrt{\varepsilon}(2+\varepsilon)}{(1+\varepsilon)^2} + \frac{\eta_{n-1,p-1}\|\boldsymbol{\phi}_{n-1,p-1}\|\|\boldsymbol{X}_{n-1,p-1}\|\|\boldsymbol{X}_{n,p}\|}{\|\boldsymbol{r}_{n-1,p-1}\|^2}\right. \\
&\quad \left.\times\left(\sqrt{\|\boldsymbol{\phi}_{n-1,p-1}\|^2+1} + \sqrt{\left\|\tilde{\boldsymbol{\phi}}_{n-1,p-1}\right\|^2+1}\right)\right)\sqrt{\varepsilon}\ell_{n,p}(i) \\
&\leq \left(1 + \frac{\eta_{n-1,p-1}\|\boldsymbol{\phi}_{n-1,p-1}\|\|\boldsymbol{X}_{n-1,p-1}\|\|\boldsymbol{X}_{n,p}\|}{\|\boldsymbol{r}_{n-1,p-1}\|^2}\right. \\
&\quad \left.\times\left(\sqrt{\|\boldsymbol{\phi}_{n-1,p-1}\|^2+1} + \sqrt{\left\|\tilde{\boldsymbol{\phi}}_{n-1,p-1}\right\|^2+1}\right)\right)\sqrt{\varepsilon}\ell_{n,p}(i).
\end{aligned}$$

Motivated from Theorem 27 along with using Theorem 29, we obtain

$$\begin{aligned}
|\ell_{n,p}(i) - \tilde{\ell}_{n,p}(i)| &\leq \left(1 + \frac{\eta_{n-1,p-1}\|\boldsymbol{\phi}_{n-1,p-1}\|\|\boldsymbol{X}_{n-1,p-1}\|\|\boldsymbol{X}_{n,p}\|}{\|\boldsymbol{r}_{n-1,p-1}\|}\right. \\
&\quad \left.\times\left(\frac{\sqrt{\|\boldsymbol{\phi}_{n-1,p-1}\|^2+1}}{\|\boldsymbol{r}_{n-1,p-1}\|} + \frac{\sqrt{\left\|\tilde{\boldsymbol{\phi}}_{n-1,p-1}\right\|^2+1}}{\|\boldsymbol{r}_{n-1,p-1}\|}\right)\right)\sqrt{\varepsilon}\ell_{n,p}(i) \\
&\leq \left(1 + \frac{\eta_{n-1,p-1}\sqrt{\|\boldsymbol{\phi}_{n-1,p-1}\|^2+1}\,\|\boldsymbol{X}_{n,p}\|^2}{\|\boldsymbol{r}_{n-1,p-1}\|}\right. \\
&\quad \left.\times\left(\frac{\sqrt{\|\boldsymbol{\phi}_{n-1,p-1}\|^2+1}}{\|\boldsymbol{r}_{n-1,p-1}\|} + \frac{(1+\varepsilon)\sqrt{\left\|\tilde{\boldsymbol{\phi}}_{n-1,p-1}\right\|^2+1}}{\|\tilde{\boldsymbol{r}}_{n-1,p-1}\|}\right)\right)\sqrt{\varepsilon}\ell_{n,p}(i).
\end{aligned}$$

Now, by using Theorem 27, we obtain

$$\begin{aligned}
|\ell_{n,p}(i) - \tilde{\ell}_{n,p}(i)| &\leq \left(1 + \frac{\eta_{n-1,p-1}\lambda_{\max}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})}{\sqrt{\lambda_{\min}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})}}\right. \\
&\quad \left.\times\left(\frac{1}{\sqrt{\lambda_{\min}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})}} + \frac{1+\varepsilon}{\sqrt{\lambda_{\min}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})}}\right)\right)\sqrt{\varepsilon}\ell_{n,p}(i)
\end{aligned}$$

$$
\begin{aligned}
&\le \left(1 + \frac{3\eta_{n-1,p-1}\lambda_{\max}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})}{\lambda_{\min}(\boldsymbol{X}_{n,p}^{\intercal}\boldsymbol{X}_{n,p})}\right)\sqrt{\varepsilon}\ell_{n,p}(i) \\
&= \left(1 + 3\eta_{n-1,p-1}\kappa^2(\boldsymbol{X}_{n,p})\right)\sqrt{\varepsilon}\ell_{n,p}(i).
\end{aligned}
$$

■

### A.3 Proof of Theorem 13

**Proof** We prove by induction. For $p = 2$, it is derived directly from Theorem 21. Let us assume that the statement of theorem is correct for all values of $p < \bar{p}$, and prove that it is also correct for $p = \bar{p}$.

$$
\begin{aligned}
|\ell_{n,\bar{p}}(i) - \hat{\ell}_{n,\bar{p}}(i)| &= \left|\ell_{n-1,\bar{p}-1}(i) + \frac{(\boldsymbol{r}_{n-1,p-1}(i))^2}{\|\boldsymbol{r}_{n-1,p-1}\|^2} - \hat{\ell}_{n-1,\bar{p}-1}(i) - \frac{(\hat{\boldsymbol{r}}_{n-1,p-1}(i))^2}{\|\hat{\boldsymbol{r}}_{n-1,p-1}\|^2}\right| \\
&\le \left|\ell_{n-1,\bar{p}-1}(i) - \hat{\ell}_{n-1,\bar{p}-1}(i)\right| + \left|\frac{(\boldsymbol{r}_{n-1,p-1}(i))^2}{\|\boldsymbol{r}_{n-1,p-1}\|^2} - \frac{(\hat{\boldsymbol{r}}_{n-1,p-1}(i))^2}{\|\hat{\boldsymbol{r}}_{n-1,p-1}\|^2}\right| \\
&\le \left(1 + 3\eta_{n-2,\bar{p}-2}\kappa^2(X_{n-1,\bar{p}-1})\right)(\bar{p}-2)\sqrt{\varepsilon}\ell_{n-1,\bar{p}-1}(i) \\
&\quad + \left(1 + 3\eta_{n-1,\bar{p}-1}\kappa^2(X_{n,\bar{p}})\right)\sqrt{\varepsilon}\ell_{n,\bar{p}}(i) \\
&\le \left(1 + 3\eta_{n-1,\bar{p}-1}\kappa^2(X_{n,\bar{p}})\right)(\bar{p}-2)\sqrt{\varepsilon}\ell_{n,\bar{p}}(i) \\
&\quad + \left(1 + 3\eta_{n-1,\bar{p}-1}\kappa^2(X_{n,\bar{p}})\right)\sqrt{\varepsilon}\ell_{n,\bar{p}}(i) \\
&= \left(1 + 3\eta_{n-1,\bar{p}-1}\kappa^2(X_{n,\bar{p}})\right)(\bar{p}-1)\sqrt{\varepsilon}\ell_{n,\bar{p}}(i).
\end{aligned}
$$

The second last inequality comes from the induction hypothesis as well as Theorem 21 and the last inequality is from Theorem 28. ■

### A.4 Proof of Theorem 16

**Proof** From Theorem 9, we have (14b), which in turn implies

$$\left|\boldsymbol{\phi}_{n,p^*}(k) - \hat{\boldsymbol{\phi}}_{n,p^*}(k)\right| \le \sqrt{\varepsilon}\eta_{n,p^*}\left\|\boldsymbol{\phi}_{n,p^*}\right\|, \quad \text{for } 1 \le k \le p^*.$$

One can estimate the PACF value at lag $p^*$ using the $p^{*\text{th}}$ component of the CMLE of the parameter vector based on the full data matrix, i.e., $\boldsymbol{\phi}_{n,p^*}(p^*)$, (Shumway and Stoffer, 2017, Chapter 3). Hence, (17a) now readily follows by an application of reverse triangular inequality.

To show (17b), we recall that (Shumway and Stoffer, 2017, Chapter 3)

$$\texttt{PACF}_p = \frac{\texttt{Cov}(p) - \sum_{k=1}^{p-1}\phi_k\texttt{Cov}(p-k)}{\sigma_W^2},$$

where $\texttt{Cov}(p)$ is the autocovariance function at lag $p$ and $\sigma_W^2$ is the variance of white noise series in an $\texttt{AR}(\texttt{p}-\texttt{1})$ model. It follows that $\tau_p$ is given by plugin the CMLE of $\texttt{Cov}(p)$, $\phi_k$

for $k = 1, \ldots, p-1$ and $\sigma_W^2$, that is,

$$\tau_p = \frac{\gamma(p) - \sum_{k=1}^{p-1} \boldsymbol{\phi}_{n,p-1}(k)\gamma(p-k)}{\left\| \boldsymbol{r}_{n,p-1} \right\|^2 / n}.$$

Hence, for $p > p^*$, we have

$$\begin{aligned}
|\hat{\tau}_p| &= \frac{\left| \gamma(p) - \sum_{k=1}^{p-1} \hat{\boldsymbol{\phi}}_{n,p-1}(k)\gamma(p-k) \right|}{\left\| \hat{\boldsymbol{r}}_{n,p-1} \right\|^2 / n} \\
&= \frac{\left| \gamma(p) - \sum_{k=1}^{p-1} \left[ \left( \hat{\boldsymbol{\phi}}_{n,p-1}(k) + \boldsymbol{\phi}_{n,p-1}(k) - \boldsymbol{\phi}_{n,p-1}(k) \right) \gamma(p-k) \right] \right|}{\left\| \hat{\boldsymbol{r}}_{n,p-1} \right\|^2 / n} \\
&\leq |\tau_p| + \frac{\sum_{k=1}^{p-1} \left| \left( \boldsymbol{\phi}_{n,p-1}(k) - \hat{\boldsymbol{\phi}}_{n,p-1}(k) \right) \gamma(p-k) \right|}{\left\| \boldsymbol{r}_{n,p-1} \right\|^2 / n} \\
&\leq |\tau_p| + \frac{\gamma(0) \sum_{k=1}^{p-1} \left| \boldsymbol{\phi}_{n,p-1}(k) - \hat{\boldsymbol{\phi}}_{n,p-1}(k) \right|}{\left\| \boldsymbol{r}_{n,p-1} \right\|^2 / n} \\
&\leq |\tau_p| + \frac{\gamma(0)\sqrt{p-1} \left\| \boldsymbol{\phi}_{n,p-1} - \hat{\boldsymbol{\phi}}_{n,p-1} \right\|}{\left\| \boldsymbol{r}_{n,p-1} \right\|^2 / n} \\
&\leq |\tau_p| + \frac{\eta_{n,p} \left\| \boldsymbol{\phi}_{n,p} \right\| \gamma(0)}{\left\| \boldsymbol{r}_{n,p-1} \right\|^2 / n} \sqrt{(p-1)\varepsilon}.
\end{aligned}$$

Now, the result follows by noting that $\left\| \boldsymbol{r}_{n,p-1} \right\|^2 / n$ is an MLE estimate of $\sigma_W^2$, and from convergence in probability of this estimate, we have that, for large enough $n$, it is bounded with probability at least $1 - \delta$. ■

### A.5 Proof of Theorem 17

**Proof** Consider an input $\texttt{AR}(p^*)$ time series data of size $n$. From Theorem 11, Theorem 13, and Theorem 19, given the fully-approximate leverage scores for the data matrix corresponding to the $\texttt{AR}(p-1)$ models for $p$ varying from 2 to $p^*$, we can estimate those of $\texttt{AR}(p)$ models in $\mathcal{O}(n)$ time. Here, we assume that $\kappa(\boldsymbol{X}_{n,p})$ does not scale with the dimension $p$ (at least unfavorably so), and treat it as a constant. Theorem 13 implies that we must choose $0 < \varepsilon \leq p^{-2}$. Now, solving the compressed OLS problem (e.g., applying QR factorization with Householder reflections) requires $\mathcal{O}\left(sp^2\right) = \mathcal{O}\left(p^3 \log p / \varepsilon^2\right)$. As a result, the overall complexity of performing the $\texttt{LSAR}$ for an input $\texttt{AR}(\texttt{p}^*)$ time series data is $\mathcal{O}\left(\sum_{p=1}^{p^*} (n + p^3 \log p / \varepsilon^2)\right) = \mathcal{O}\left(np^* + {p^*}^4 \log p^* / \varepsilon^2\right)$. ■

## References

M. Abolghasemi, J. Hurley, A. Eshragh, and B. Fahimnia. Demand forecasting in the presence of systematic events: Cases in capturing sales promotions. *International Journal of Production Economics*, 230:107892, 2020.

C.W. Anderson, E.A. Stolz, and S. Shamsunder. Multivariate autoregressive models for classification of spontaneous electroencephalographic signals during mental tasks. *IEEE Transactions on Biomedical Engineering*, 45(3):277–286, 1998.

H. Avron, P. Maymounkov, and S. Toledo. Blendenpik: Supercharging LAPACK's least-squares solver. *SIAM Journal on Scientific Computing*, 32(3):1217–1236, 2010.

H. Avron, M. Kapralov, C. Musco, C. Musco, A. Velingker, and A. Zandieh. A universal sampling method for reconstructing signals with simple Fourier transforms. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 1051–1063. ACM, 2019.

P.J. Blackwell and R.A. Davis. *Time Series: Theory and Methods.* Springer Series in Statistics. Springer, 2009. ISBN 9781441903198.

G.E.P. Box and G.M. Jenkins. *Time Series Analysis, Forecasting and Control.* Holden-Day, San Francisco, 1976.

N. Chakravarthy, A. Spanias, L.D. Iasemidis, and K. Tsakalis. Autoregressive modeling and feature analysis of dna sequences. *EURASIP Journal on Advances in Signal Processing*, 2004:13–28, 2004.

K.L Clarkson and D.P. Woodruff. Low-rank approximation and regression in input sparsity time. *Journal of the ACM (JACM)*, 63(6):54, 2017.

P. Drineas, M.W. Mahoney, S. Muthukrishnan, and T. Sarlós. Faster least squares approximation. *Numerische Mathematik*, 117(2):219–249, 2011.

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.

A. Eshragh, B. Ganim, T. Perkins, and K Bandara. The importance of environmental factors in forecasting australian power demand. *Environmental Modeling & Assessment*, 2021. doi: 10.1007/s10666-021-09806-1.

R. Estrin, D. Orban, and M.A. Saunders. LSLQ: An iterative method for linear least-squares with an error minimization property. *SIAM Journal on Matrix Analysis and Applications*, 40(1):254–275, 2019.

D.C-L. Fong and M. Saunders. LSMR: An iterative algorithm for sparse least-squares problems. *SIAM Journal on Scientific Computing*, 33(5):2950–2971, 2011.

G.H. Golub and C.F. Van Loan. *Matrix Computations.* Johns Hopkins paperback. Johns Hopkins University Press, 1983. ISBN 9780801830112.

J.D. Hamilton. A new approach to the economic analysis of nonstationary time series and the business cycle. *Econometrica*, 57(2):357–384, 1989.

J.D. Hamilton. *Time Series Analysis*. Princeton University Press, New Jersey, 1994.

R.A. Huerta, T.S. Mosqueiro, J. Fonollosa, N.F. Rulkov, and I. Rodríguez-Luján. Online humidity and temperature decorrelation of chemical sensors for continuous monitoring. *Chemometrics and Intelligent Laboratory Systems*, 157(15):169–176, 2016.

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

M.W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends® in Machine Learning*, 2011.

M.W. Mahoney. Lecture notes on randomized linear algebra. *arXiv preprint arXiv:1608.04481*, 2016.

X. Meng and M.W. Mahoney. Low-distortion subspace embeddings in input-sparsity time and applications to robust linear regression. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*, pages 91–100, 2013.

X. Meng, M.A. Saunders, and M.W. Mahoney. LSRN: A parallel iterative solver for strongly over-or underdetermined systems. *SIAM Journal on Scientific Computing*, 36(2):C95–C118, 2014.

J.W. Messner and P. Pinson. Online adaptive lasso estimation in vector autoregressive models for high dimensional wind power forecasting. *International Journal of Forecasting*, 35(4):1485–1498, 2019.

J. Nelson and H.L. Nguyen. OSNAP: Faster numerical linear algebra algorithms via sparser subspace embeddings. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science*, pages 117–126, 2013.

C.C. Paige and M.A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software (TOMS)*, 8(1):43–71, 1982.

G. Raskutti and M.W. Mahoney. A statistical perspective on randomized sketching for ordinary least-squares. *Journal of Machine Learning Research*, 17(1):7508–7538, 2016.

F. Roosta-Khorasani, G.J. Székely, and U.M. Ascher. Assessing stochastic algorithms for large scale nonlinear least squares problems using extremal probabilities of linear combinations of gamma random variables. *SIAM/ASA Journal on Uncertainty Quantification*, 3(1):61–90, 2015.

P.J. Rousseeuw and M. Hubert. Robust statistics for outlier detection. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(1):73–79, 2011.

X. Shen and Q. Lu. Joint analysis of genetic and epigenetic data using a conditional autoregressive model. *BMC Genetics*, 16(Suppl 1):51–54, 2018.

X. Shi and D. P. Woodruff. Sublinear time numerical linear algebra for structured matrices. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):4918–4925, Jul. 2019.

R.H. Shumway and D.S. Stoffer. *Time Series Analysis and Its Applications.* Springer, London, 2017.

Marc Van Barel, Georg Heinig, and Peter Kravanja. A superfast method for solving Toeplitz linear least squares problems. *Linear algebra and its applications*, 366:441–457, 2003.

H.Y. Wang, M. Yang, and J. Stufken. Information-based optimal subdata selection for big data linear regression. *Journal of the American Statistical Association*, 114(525):393–405, 2018.

D.P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 2014.

Yuanzhe Xi, Jianlin Xia, Stephen Cauley, and Venkataramanan Balakrishnan. Superfast and stable structured solvers for Toeplitz least squares via randomized sampling. *SIAM Journal on Matrix Analysis and Applications*, 35(1):44–72, 2014.

J. Yang, X. Meng, and M.W. Mahoney. Implementing randomized matrix algorithms in parallel and distributed environments. In *Proceedings of the IEEE*, pages 58–92, 2016.