Multivariate Singular Spectrum Analysis: A Principled, Practical, and Performant Solution for Time Series Imputation and Forecasting

by
Abdullah Alomar
B.S., King Saud University (2016)
Submitted to the Center for Computational Science and Engineering
and
Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degrees of
Master of Science in Computational Science and Engineering
and
Master of Science in Electrical Engineering and Computer Science
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
September 2021

Author .....
Center for Computational Science and Engineering
and
Department of Electrical Engineering and Computer Science
May 21, 2021

Certified by .....
Devavrat Shah
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Certified by .....
Youssef Marzouk
Professor, Department of Aeronautics and Astronautics
Thesis Supervisor

Accepted by .....
Nicolas Hadjiconstantinou
Professor, Department of Mechanical Engineering
Co-Director, Center for Computational Science and Engineering

Accepted by .....
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Multivariate Singular Spectrum Analysis: A Principled, Practical, and Performant Solution for Time Series Imputation and Forecasting

by

Abdullah Alomar

Submitted to the Center for Computational Science and Engineering
and
Department of Electrical Engineering and Computer Science
on May 21, 2021, in partial fulfillment of the
requirements for the degrees of
Master of Science in Computational Science and Engineering
and
Master of Science in Electrical Engineering and Computer Science

Abstract

The analysis of multivariate time series data is of great interest across many domains, including cyberphysical systems, finance, retail, and healthcare to name a few. A common goal across all of these domains is accurate imputation and forecasting of multivariate time series in the presence of noisy or missing data. Given the growing need to embed predictive functionality in high-performance systems, especially in applications with time series data (e.g., financial systems, control systems), it is increasingly vital that we build *principled* prediction algorithms that are statistically and computationally *performant*, and more broadly *accessible*. To that end, we introduce a novel variant of multivariate singular spectrum analysis (mSSA) that allows for accurate imputation and forecasting of latent time-varying mean and variance of multivariate time series. We further justify this algorithm by introducing a natural Spatio-temporal factor model, under which the algorithm is theoretically analyzed; Specifically, we establish the in-sample prediction error of our mSSA variant for both imputation and forecasting.

Further, we propose an incremental variant of the algorithm, upon which, a real-time prediction system for time series data, tspDB, is instantiated and evaluated. tspDB aims to increase accessibility to predictive functionalities for time series data through the direct integration with existing relational time series Databases. Finally, through rigorous experiments, we show that tspDB provides state-of-the-art statistical accuracy while maintaining a superior computational performance with an incremental model update, low model training time, and low latency for prediction queries.

Thesis Supervisor: Devavrat Shah
Title: Professor of Electrical Engineering and Computer Science

Multivariate Singular Spectrum Analysis: A Principled, Practical, and Performant Solution for Time Series Imputation and Forecasting

by

Abdullah Alomar

Submitted to the Center for Computational Science and Engineering
and
Department of Electrical Engineering and Computer Science
on May 18, 2021, in partial fulfillment of the
requirements for the degrees of
Master of Science in Computational Science and Engineering
and
Master of Science in Electrical Engineering and Computer Science

Abstract

The analysis of multivariate time series data is of great interest across many domains, including cyberphysical systems, finance, retail, and healthcare to name a few. A common goal across all of these domains is accurate imputation and forecasting of multivariate time series in the presence of noisy or missing data. Given the growing need to embed predictive functionality in high-performance systems, especially in applications with time series data (e.g., financial systems, control systems), it is increasingly vital that we build *principled* prediction algorithms that are statistically and computationally *performant*, and more broadly *accessible*. To that end, we introduce a novel variant of multivariate singular spectrum analysis (mSSA) that allows for accurate imputation and forecasting of latent time-varying mean and variance of multivariate time series. We further justify this algorithm by introducing a natural Spatio-temporal factor model, under which the algorithm is theoretically analyzed; Specifically, we establish the in-sample prediction error of our mSSA variant for both imputation and forecasting. Further, we propose an incremental variant of the algorithm, upon which, a real-time prediction system for time series data, tspDB, is instantiated and evaluated. tspDB aims to increase accessibility to predictive functionalities for time series data through the direct integration with existing relational time series Databases. Finally, through rigorous experiments, we show that tspDB provides state-of-the-art statistical accuracy while maintaining a superior computational performance with an incremental model update, low model training time, and low latency for prediction queries.

Thesis Supervisor: Devavrat Shah

Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

I have been very fortunate to have many people who genuinely cared for my development and success. I would like to begin by thanking my advisor, Devavrat Shah, whose patience, belief, and care made this journey both pleasant and successful. When I first came here to MIT, I knew that I had a challenging road ahead of me, but his patience and guidance helped me navigate through it. Devavrat has taken a big gamble on an unproven researcher, and has trusted me with research projects that he cares deeply about, and for this, I will always be grateful. He has also taught me how to dissect complex problems and prioritize simple and elegant solutions over unnecessarily complex ones.

Another benefit of joining Devavrat's lab is the opportunity to work with excellent collaborators; without them, this work would not have been possible. Anish Agarwal has been a great collaborator and mentor: guiding me throughout my research, pushing me to my limit despite my continuous resistance, and genuinely caring about my development as a researcher. I would also like to thank Jehangir Amjad, Dennis Shen, Zhi Xu, Varkey Alumootil, and Cindy Yang for the stimulating discussion, contributions, and help in our joint projects.

This journey would not have been possible without the continuous support of my family. I find myself unable to describe how grateful I am to my parent's love, support, and sacrifice. One of my greatest blessings in life is to have seven supportive and loving siblings, without whom I would not be the person I'm today. From a very young age, Afnan has strengthened my confidence in myself by convincing me that I'm smart enough to achieve whatever I put my mind to, and kept reinforcing that idea until I (naively) believed it. Anfal has inspired me to be strong-willed, determined, and resolute. Fotoon has taught me how to be compassionate and was an integral part in the fun and lighthearted moments. The unconditional love of my younger siblings, Rahaf, Taif, Mohammad, and Ibraheem, and their belief in me, have provided me with the strength I needed in my hardest moments; it also eased my stress as I realized that I already have what truly matters in life.

I was born in Riyadh and spent my first 24 years there, and moving thousands of miles away to Boston would have been extremely challenging without the wonderful friends I have here. I have had the fortune of sharing this journey with Arwa Alanqary, Abdulaziz Alhasan, and Mohammad Alrished. All of them were always there when I needed them. I would also like to thank Fahad Alhasoun, Sarah Alnegheimish, Mohammad Alhajri, Ebrahim Aljohani, Tony Tohme, and Mohammad Alsobay for their friendship and continued support.

I would also like to thank my thesis reader Youssef Marzouk, for his support of my research. I'm especially grateful for the Center for Computational Engineering (CCES) at King Abdulaziz City of Science and Technology for providing me with a life-changing opportunity and for sponsoring my studies. I am also grateful to Anas Alfaris for his mentorship and trust; and to Saleh Alshebeilli, for taking me under his wing and giving me the first taste of research during my undergraduate days.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Time series data is ubiquitous. Its analysis has diverse and crucial applications across many domains, including predicting the behavior of physical systems, demand in retail, stock prices, and weather patterns. One of the main inference tasks across these domains is accurate imputation and forecasting of multivariate time series in the presence of noisy and missing data.

The task of time series forecasting and imputation has been heavily studied within econometrics, statistics, electrical engineering, computer science, and applied mathematics [21, 50, 44, 25]. Indeed, a plethora of time series forecasting and imputation algorithms exist. Despite the ubiquity of time series applications and the abundance of excellent forecasting and imputation algorithms, predictive functionalities remain inaccessible to domain experts and decision-makers without a team of data scientists [49].

Arguably, one of the major bottlenecks towards this goal is the complex engineering and data processing required before training a prediction algorithm. Specifically, the complex and error-prone machine learning (ML) workflow which consist of: (i) taking data from a datastore or database (DB) into a particular work environment format (e.g. spark data-frame); (ii) applying proper data transformation and feature engineering tasks; and (iii) continuously tuning the algorithm's hyperparameters has made extracting actionable insights from data an unwieldy and inaccessible task. Indeed, a classical method such as ARIMA may require data transformation tasks (e.g.,

de-trending) informed by both domain knowledge and expertise in data science. On the other hand, expressive deep neural networks offer a more generic and empirically effective solution. However, a lack of clear understanding of when and how well such solutions work, and the continuous tuning required for the hyperparameters hinder its adoption for real-world applications.

The above illustrates the need for a principled, practically accessible, and performant solution for time series imputation and forecasting. This thesis investigate the feasibility of such solution by exploring the idea of directly integrating principled and performant forecasting and imputation algorithm into a time series DB, i.e., a proof-of-concept predictive time series DB. Specifically, we showcase tspDB: a time-series DB that enables predictive query functionality in any existing relational DB.

tspDB is a principled solution as it uses a simple and theoretically justified variant of the classical multivariate singular spectrum analysis (mSSA) method. Specifically, through careful analysis, we precisely answer the question of *when and why does mSSA work* in the following three chapters of this thesis. We do so by positing that multivariate time series data follows a spatio-temporal factor model we propose. We further provide empirical and theoretical evidence to show that the proposed model covers a wide range of time series dynamics. Finally, under this model, we provide finite-sample analysis of mSSA's estimation error for imputation and forecasting.

tspDB is a practical and accessible solution as it directly integrates with time series DBs and exposes a simple and easy-to-use interface. Crucially, building prediction models and producing predictions in tspDB is as easy as writing an SQL query, as shown in Figure 1-1. Pleasingly, the low computational footprint of the mSSA variant we propose makes it possible to integrate with time series DBs without hindering the DB's original performance. Further, the mSSA variant we use for tspDB allows to estimate the time-varying variance of a multivariate time series, thereby providing an estimate of the volatility of the time series and a quantification of the certainty of the prediction. Such uncertainty quantification is crucial in many practical applications.

Finally, tspDB is a performant solution. In terms of statistical accuracy, tspDB performs as well as or better than state-of-the-art time series methods in both impu-

```
PREDICT company1
FROM stock
WITH PREDICTION_INTERVAL = 95%
WHERE day = 101;
```

(a) A PREDICT query in tspDB.

```
CREATE PREDICTION_MODEL
ON stock(company1, company2,
   company3)
WITH day AS TIME_COLUMN;
```

(b) Building a prediction model in tspDB.

Figure 1-1: Proposed interface for tspDB. See Chapter 5 for details

tation and forecasting. See Table 1.1 for details.

Table 1.1: tspDB statistically outperforms other state-of-the-art algorithms, including LSTMs and DeepAR across many datasets. We use the average normalized root mean squared error (NRMSE) as our metric. See Chapter 6 for details.

	Mean Imputation (NRMSE)				Mean Forecasting (NRMSE)			
	Electricity	Traffic	Synthetic	Financial	Electricity	Traffic	Synthetic	Financial
tspDB	**0.391**	**0.494**	**0.253**	**0.283**	**0.483**	0.525	**0.196**	**0.358**
LSTM	NA	NA	NA	NA	0.551	**0.473**	0.444	1.203
DeepAR	NA	NA	NA	NA	0.484	0.474	0.331	0.395
TRMF	0.694	0.512	0.325	0.513	0.534	0.570	0.267	0.464
Prophet	NA	NA	NA	NA	0.582	0.617	1.005	1.296

This thesis is a detailed exposition on this principled, practical, and performant solution we propose. Chapters 2, 3 and 4 present and analyze the mSSA variant we propose. Specifically, it provides an answer to the question of why and how mSSA works. Chapter 5 focuses on *operationalizing* mSSA by presenting an instantiation of tspDB, built as an extension on top of the open-source DB management system PostgreSQL. Then through extensive testing, Chapter 6 shows that this solution is indeed statistically and computationally performant.

1.1 Overview of Contribution

As our primary contribution, we provide a principled, practical, and performant solution to time series imputation and forecasting. To that end, several algorithmic, theoretical, and empirical contributions were made. Below, we detail the various aspects of our contribution.

A Novel Spatio-temporal Factor Model for Multivariate Time Series. In

our setup, we consider a collection of N time series with T observations. Note that this multivariate time series can be collectively viewed as a $N \times T$ matrix. In this thesis, we introduce a spatio-temporal factor model that captures both the spatial structure (across N time series) and temporal structure (across the T observations) in a multivariate time series. To capture the spatial structure, i.e., the relationship across different time series, we model the multivariate time series matrix to be low-rank – that is, each time series can be viewed as a linear combination of a few *fundamental* time series (See Property 2.2.1). To capture the temporal structure, we further assume that each *fundamental* time series has an *(approximately) low-rank Hankel matrix*[1] representation (See Property 2.2.2). For $N = 1$, this subsumes the model considered to explain the success of SSA in [9] as a special case. The traditional modeling approach in time series analysis posits that a time series is a mixture of a trend (low-order polynomial), a seasonal (finite sum of harmonics), and a stationary (linear time-invariant function) component. Crucially, we show that each of these components (low-order polynomial, harmonics, and linear time-invariant function) indeed follow our model.

Novel Variant of mSSA to Estimate Mean and Variance. We propose a variant of mSSA that is a generalization of the method proposed in [9] and extends it in the following two ways: (i) it provides predictions for multivariate time series data; (ii) it allows one to estimate the time-varying variance of a multivariate time series (a la GARCH-like models), thereby addressing an important issue in time series analysis of how to estimate the volatility of the time series. Details of the algorithm can be found in Chapter 3.

Finite sample analysis of mSSA. Under the proposed spatio-temporal factor model, we establish that the mean imputation and (in-sample) forecasting prediction error scale as $1/\sqrt{NT}$, see Theorems 4.1.1 and 4.1.2. For $N = 1$, it implies that the SSA algorithm described above has imputation and forecasting error scaling as $1/\sqrt{T}$. That is, mSSA improves performance by a $\sqrt{N}$ factor over SSA by utilizing

[1]See Definition 2.1.1 for the definition of the Hankel matrix induced by a time series.

Table 1.2: Comparison of finite-sample results with relevant algorithms in the literature.

Method	Functionality		Mean Estimation		Variance Estimation	
	Multivariate time series	Variance Estimation	Imputation	Forecasting	Imputation	Forecasting
This Work	Yes	Yes	$1/\sqrt{NT}$	$1/\sqrt{NT}$	$1/\sqrt{NT}$	$1/\sqrt{NT}$
mSSA - Literature	Yes	No	–	–	–	–
SSA [28, 9]	No	No	$T^{-1/4}$	–	–	–
Neural Network [46, 19]	Yes	No	–	–	–	–
TRMF [58, 42]	Yes	No	$(\min(N,T))^{-1}$	–	–	–

information across the N time series. This also improves upon the prior work of [9] which established the weaker result that SSA has imputation error scaling as $1/T^{\frac{1}{4}}$ – also, [9] does not establish a result for the forecasting error of SSA. Further, existing matrix estimation based methods establish that the imputation prediction error scales as $1/\min(N,T)$. This is indeed the primary result of the works [58, 42] That is, while the algorithm stated in [58, 42] utilizes the temporal structure in addition to the spatial structure, the theoretical guarantees do not reflect it. This explains why the guarantees provided by such methods are weaker (since $1/\min(N,T) \geq 1/\sqrt{NT}$) than that obtained by mSSA. Further, we show that mSSA performs consistent imputation/forecasting of the time-varying variance parameter for a broad class of time series models, at rate $1/\sqrt{NT}$. See Table 1.2 for a summary of our theoretical results.

tspDB. One of the main deliverable of this work is an instantiation of this proposed solution, which we provide through tspDB. tspDB is a real-time prediction system for time series data, which aims to alleviate the data engineering bottleneck by providing direct access to predictive functionalities. In Chapter 5, we detail an instantiation of tspDB as an extension on top of PostgreSQL. We first show the simple SQL-like interface we propose to access tspDB's predictive functionalities. Further, we show through this proof-of-concept implementation that integrating the proposed mSSA method with the DB does not degrade the DB's computational performance in terms of insert throughput. We finally show that the query latency of the proposed prediction queries is of similar latency to that of standard SELECT queries.

Extensive Statistical and Computational Benchmarking of tspDB. In Chap-

ter 6, we extensively benchmark tspDB's statistical and computational performance against popular state-of-the-art prediction libraries. In Section 6.2, we verify tspDB's statistical accuracy by testing its performance on benchmark datasets of time series data from real-world applications (e.g., used in [58]) and also on synthetic datasets we generate. We compare tspDB's accuracy against state-of-the-art algorithms, including LSTMs [27], DeepAR (by Amazon [46]), Temporal Regularized Matrix Factorization (TRMF) [58] and Prophet (by Facebook [24]). We measure the imputation and forecasting prediction accuracy, for both mean and variance, of these various algorithms as we vary the level of data quality. We find that on standard time series benchmarks, using NRMSE as our metric (See 6.1.5 for the definition of NRMSE), tspDB outperforms all other methods in both mean imputation and forecasting and variance estimation.

In Section 6.3, we test how tspDB compares against the popular, open-source time series prediction libraries stated above—LSTM, DeepAR, TRMF, and Prophet in terms of computational performance. With respect to both ML model training time and prediction query latency, tspDB outperforms all other prediction algorithms we compare against—refer to Section 6.3 for a precise description of how ML model training time and prediction latency are measured. We highlight that compared to LSTM and DeepAR, tspDB's median ML model training time and prediction latency is 59-62x and 94-95x quicker, respectively.

1.2 Related Work

In this section, we discuss prior work in two areas relevant to this thesis's contributions, namely multivariate time series analysis and predictive modeling in DB systems.

1.2.1 Multivariate Time Series Analysis

Given the ubiquity of multivariate time series analysis, it will not be possible to do justice to the entire literature. We focus on a few relevant techniques, most relevant to compare against, either theoretically or empirically.

Singular Spectrum Analysis. We start by discussing the classical Singular Spectrum Analysis (SSA) method, and its multivariate variant (mSSA). A good overview of the literature on SSA can be found in [28]. The key steps of the classical SSA method are:

1. Create a Hankel matrix from the time series data.
2. Perform Singular Value Decomposition (SVD) of that Hankel matrix.
3. Group the singular values based on user belief of the model that generated the process.
4. Perform diagonal averaging to "Hankelize" the grouped rank-1 matrices outputted from the SVD to create a set of time series.
5. Learn a linear model for each "Hankelized" time series for the purpose of forecasting.

The theoretical analysis of this SSA method has been focused on proving that many univariate time series have a low-rank Hankel representation, and secondly on defining sufficient *asymptotic* conditions for when the singular values of the various time series components are separable, thereby justifying Step 3 of the method. Step 3 of the original SSA method requires user input and Steps 4 and 5 are not robust to noise and missing values due to the strong dependence across entries of the Hankel representation of the time series. To overcome these limitations, a simpler and practically useful version has been introduced in [9] and a finite-sample analysis of it was done. In that proposed variant, the Page matrix is used instead of the Hankel matrix (see Chapter 2 for a definition of these two representations), and only a single linear forecaster is used. This is in contrast to the user having to specify how the singular values need to be grouped in the classical SSA method. In this thesis, we improve upon the analysis of [9] by providing stronger bounds for imputation prediction error and give new bounds for forecasting prediction error, which were missing in [9]. The original mSSA method, like the original SSA method, involves the five steps described above, but first, the Hankel matrices induced by each of the N time series are stacked column-wise. In this work, as described in Section 3.1, we introduce a simpler, practically useful variant of mSSA. The theoretical analysis of mSSA (original or the variant

considered here) is entirely absent in the literature despite its empirical success and popularity (see [31, 30, 40]).

Matrix Estimation Based Multivariate Time Series Methods. There is a recent line of work in time series analysis (see [56, 58]), where multiple time series are viewed collectively as a matrix, and some form of matrix factorization is done. Most of these methods make strong prior model assumptions on the underlying time series, and the algorithm changes based on the assumptions made on the time-series dynamics that generated the data. Further, finite sample analysis, especially with respect to forecasting error, of such methods is usually lacking. We highlight one method, Temporal Regularized Matrix Factorization (TRMF) (see [58]), which we directly compare against due to its popularity, and as it achieves state-of-the-art imputation and forecasting empirical performance. The authors in [58] provide finite sample imputation analysis for an instance of the model considered in this work, but forecasting analysis is absent. Specifically, they establish that the imputation error scales as $1/\min(N, T)$. This is a consequence of the low-rank structure of the time series matrix. But they fail to utilize, at least in the theoretical analysis, the temporal structure. In our analysis of mSSA, we capture the temporal structure, and hence our imputation error scales as $1/\sqrt{NT}$, which is a stronger guarantee. For example, for $N = \Theta(1)$, their error bound remains $\Theta(1)$ for any T, while that of mSSA would vanish as T grows.

Time-Varying Variance Estimation. The time-varying variance is a key input parameter in many sequential prediction algorithms themselves. For example, in control systems, the widely used Kalman Filter uses an estimate of the per step variance for both filtering and smoothing. Similarly, in finance, the time-varying variance of each financial instrument in a portfolio is necessary for risk adjustment. The key challenge in estimating the variance of a time series (which itself might very well be time-varying) is that, unlike the actual time series itself, we do not get to directly observe the variance. Despite the vast time series literature, existing algorithms to estimate time-varying variance are mostly heuristics and/or make restrictive, para-

metric assumptions about how the variance (and the underlying mean) evolves; for example "Auto-Regressive Conditional Heteroskedasticity" (ARCH) and "Generalized Auto-Regressive Conditional Heteroskedasticity" (GARCH) models. See [14].

Deep Learning Methods. We take a brief note of the popular time series method in the recent literature. In particular, recently, neural network (NN) based approaches have been the most popular and empirically effective. Some industry-standard neural network methods include LSTMs, and DeepAR (an industry-leading NN library for time series analysis, see [46]).

1.2.2 Predictive Modeling and DBs

As stated earlier, our work is in line with exciting recent efforts, especially in industry, exploring the potential of DB management systems to support ML algorithms [1, 4, 5, 3, 6, 2, 32, 38]. In a complementary vein, there has been a line of work to automate the process of choosing an optimal ML workload for a given task [51, 49]. There has also been a line of work to exploit DBs to make certain key computationally expensive steps that are pervasive in training ML algorithms much more efficient [41, 52, 20, 17, 35]. Further, there have been attempts to provide a declarative SQL-like language for prediction [48, 32, 11]. Our proposed system, tspDB, is very much in line with these efforts. However, crucially, rather than facilitating the use of a variety of ML algorithms or sub-routines, we take the stance of abstracting the entire ML algorithm from the user, and providing a single interface to answer both standard DB queries and predictive queries. This is in line with [51, 49] effort to democratize ML. We finally note that the data imputation task has received considerable attention in the systems for ML community. For example, in [18, 12, 37, 45], they explore systems that directly integrate imputation of missing values as an in-built functionality.

1.3 Organization

The rest of this thesis is organized as follows: In Chapter 2, we describe our setup and introduce the multivariate time series model we utilize for our analysis. In Chapter 3,

we detail our variant of the mSSA algorithm for both mean and variance imputation and forecasting. In Chapter 4, we provide the results of our finite-sample analysis for both the mean and variance estimation algorithms. In Chapter 5, we demonstrate a scalable and incremental variant of the mSSA algorithm, as well as the details of instantiating tspDB as an extension on top of PostgreSQL. Finally, in Chapter 6, we showcase the extensive testing we conduct to benchmark the statistical and computational performance of tspDB against popular state-of-the-art prediction libraries.

1.4 Bibliographical Note

The content presented in Chapter 2, Chapter 3 and Chapter 4 is a part of an article titled "On Multivariate Singular Spectrum Analysis and its Variants" currently under submission [8]. Further, the content of Chapter 5 and Chapter 6 appears in the article "tspDB: Time Series Predict DB", which is currently under submission [7].

Chapter 2

Model and Setup

2.1 Setup and Notation

Setup. We consider the settings where we have access to $N \in \mathbb{Z}$ time series, each with $T > N \in \mathbb{Z}$ evenly spaced timestamped observations. We consider discrete time indexed by $t \in \{1, \ldots, T\} \coloneqq [T]$. We denote our N *latent* time series by $f_n : \mathbb{Z} \to \mathbb{R},\ n \in [N]$. In our setting, we observe noisy and partial observations of these latent time series, as we describe next.

Observation Model. For each time $t \in [T]$ and $n \in [N]$, we observe the random variable $X_n(t)$ as follows:

$$X_n(t) = \begin{cases} f_n(t) + \eta_n(t) & \text{with probability } \rho \\ \star & \text{with probability } 1 - \rho \end{cases} \tag{2.1}$$

Where $\star$ represents a missing observation, $\rho \in (0, 1]$ is the probability of observing each entry, and the noise $\eta_n(t)$ are independent mean-zero random variables. That is, $\mathbb{E}[X_n(t)] = f_n(t)$. Further, we denotes the time-varying variance of the noise $\mathbb{E}[\eta_n^2(t)]$ by $\sigma_n^2(t)$. Note $\sigma_n^2(t)$ is a key parameter of interest in time series analysis to do uncertainty quantification, especially in applications where the volatility in the time series is time-varying (e.g., ARCH and GARCH models [15]).

Note that though $\eta_n(t)$ is independent, the latent time series, $f_n(\cdot)$, is of course

strongly dependent across t and n. As is typical in time series analysis, one models the latent time series of interest, $f_n(\cdot)$, as a mixture of a *trend*, a *seasonal*, and a *stationary* component. Where the typical model used for *trend* is a low-order polynomial in time (e.g., linear trend); *seasonality* is often modeled by a finite sum of harmonics; and *stationarity* is often modeled by an autoregressive moving average (ARMA) or an ARMA-like process. To capture that temporal structure (e.g. polynomials and harmonics) as well as the "spatial" structure (i.e., across the multiple time series), we posit that $f_n(\cdot),\ n \in [N]$ collectively satisfy a spatio-temporal factor model as described in detail in Section 2.2.

Goal. Our goal is to develop a theoretically justified, universal method to impute and forecast the latent, time-varying mean and variance of the time series. Ideally, this method scales gracefully with the amount of missing data and noise in the observations, two key issues that plague time series data. In short, the objective is two-fold, for $n \in [N]$:

1. **Mean imputation and forecasting**. Estimating $f_n(t)$ for all $t \in [T]$ (imputation) as well as learning a model to forecast $f_n(t)$ for $t > T$ (forecasting).
2. **Variance imputation and forecasting** Estimating $\sigma_n(t)$ for all $t \in [T]$ (variance imputation) as well as learning a model to forecast $\sigma_n(t)$ for $t > T$ (variance forecasting).

Time Series as a Matrix: Page and Hankel Representation. Before describing the model, we define two crucial time series representation: the Hankel and Page matrix representation of time series. We first start by describing the Hankel matrix induced by a time series.

Definition 2.1.1 (Hankel Matrix)**.** *Given a time series $f : \mathbb{Z} \to \mathbb{R}$, its Hankel matrix associated with observations over T time steps, $\{1, \ldots, T\}$, and for some parameter $L \in \mathbb{Z}, L > 1$, is given by the matrix $\mathbf{H} \in \mathbb{R}^{L \times (T-L+1)}$ with $H_{ij} = f(i + j - 1)$ for $i \in [L], j \in [T - L]$*[1].

[1]Throughout this thesis, we use $\mathbf{A}_{ij}$ or equivalently $[\mathbf{A}]_{i,j}$ to denote the i-th row and j-th column entry of matrix $\mathbf{A}$

In words, the Hankel representation simply places overlapping segments of observations of size L into the columns of an $L \times (T - L + 1)$ matrix. See Figure 2-1 for a visual depiction of this transformation.

In the Page matrix representation, contiguous segments of size L of the time series are placed into *non-overlapping* columns, creating a matrix of size $L \times \lfloor T/L \rfloor$. see Figure 2-1 for a caricature of this representation. Below, we formally define the representation.

Definition 2.1.2 (Page Matrix)**.** *Given a time series $f : \mathbb{Z} \to \mathbb{R}$, its Page matrix representation over T observations and a parameter $L \geq 1$ is given by the matrix $\mathbf{Z} \in \mathbb{R}^{L \times \lfloor T/L \rfloor}$ with $Z_{ij} = f(i + (j-1) \times L)$ for $i \in [L], j \in [\lfloor T/L \rfloor]$.*

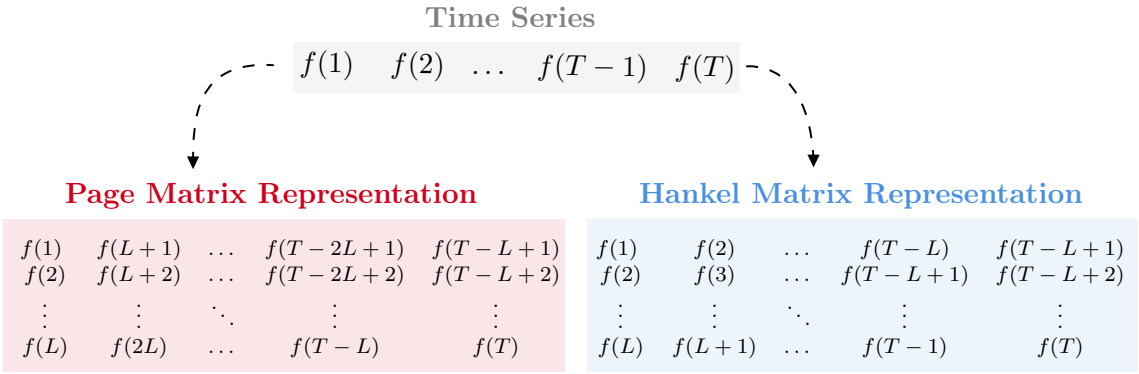


Figure 2-1: A visual depiction of the Hankel and Page matrix representation induced by the time series $f(t)$

2.2 Multivariate Time Series Model

In this section, we propose a spatio-temporal factor model for multivariate time series that explains the empirical success of mSSA (See chapter 6). First, we detail the proposed spatio-temporal model (Section 2.2.1); then, we provide a generalization of that model that allows for an even richer family of time series dynamics (Section 2.2.2); and finally, we give several examples of multivariate time series that fit our model (Section 2.2.3).

2.2.1 Spatio-Temporal Factor Model

The model we propose requires that the latent multivariate time series $f_1(\cdot), \ldots, f_N(\cdot)$ satisfy two properties: (i) Property 2.2.1, which captures the *spatial* structure across time series; (ii) Property 2.2.2, which capture the *temporal* structure within each time series.

In Property 2.2.1, we assume that each of the latent time series $f_1(\cdot), \ldots, f_N(\cdot)$ are related to each other through $R < N$ "fundamental" time series, precisely,

Property 2.2.1. *There exist* $R \in \mathbb{N}$, *where* $1 \leq R \ll \min(N, T)$ [2], $W_r : \mathbb{Z} \to \mathbb{R}$, *and* $\alpha_{nr} \in \mathbb{R}\ \forall r \in [R], n \in [N]$, *such that for any* $n \in [N], t \in [T]$, $f_n(t) = \sum_{r=1}^{R} \alpha_{nr}\, W_r(t)$, *where* $|\alpha_{nr}| \leq \Gamma_1$, $|W_r(t)| \leq \Gamma_2$ *for constants* $\Gamma_1, \Gamma_2 > 0$.

Property 2.2.1 effectively captures the "spatial" structure amongst the N time series. In other words, every time series $f_n(\cdot)$ can be expressed as weighted linear combination of at most R 'fundamental' time series. However, time series are often characterized by their "temporal" structure, i.e. structure across time in each time series. Such structure is not captured by Property 2.2.1, which calls for imposing additional structure on each "fundamental" time series $W_r(t)$. Let $\mathbf{H}_W^{(r)} \in \mathbb{R}^{L \times (T-L+1)}$ denote the Hankel matrix induced by $W_r(\cdot)$, as described in Definition 2.1.1.

Property 2.2.2. *For each* $r \in [R]$ *and for any* $T \geq 1$, *the Hankel Matrix* $\mathbf{H}_W^{(r)} \in \mathbb{R}^{L \times (T-L+1)}$ *associated with time series* $W_r(t)$, $t \in [T]$ *has rank at most* G *with* $G \ll T$.

Property 2.2.2 captures the temporal structure within each fundamental time series by imposing a low-rank structure on the Hankel matrix induced by it. Indeed, such property is observed in standard functions of time series dynamics. In fact, any finite sum of products of harmonics, low-degree polynomials, and exponentials have a low-rank Hankel matrix representation, as stated in Proposition 2.2.3 in Section 2.2.3.

Stacked Page Matrix is Low-Rank. Recall the definition of the page matrix representation (Definition 2.1.2), and let $\mathbf{Z}_f^{(n)}$ denote the Page matrix representation

[2] While not very precise, the qualitative statement $1 \leq R \ll \min(N, T)$ indicates that the number of fundamental time series is small relative to both T and N.

of the latent time series $f_n(t)$. Further, consider the stacked Page matrix $\mathbf{Z}_f$, obtained by a column-wise concatenation of the Page matrices $\mathbf{Z}_f^{(1)}, \ldots, \mathbf{Z}_f^{(N)}$. Specifically,

$$\mathbf{Z}_f = \begin{bmatrix} \mathbf{Z}_f^{(1)} & \mathbf{Z}_f^{(2)} & \ldots & \mathbf{Z}_f^{(N)} \end{bmatrix}. \tag{2.2}$$

Under the spatio-temporal factor model satisfying Properties 2.2.1 and 2.2.2, we establish the following low-rank property of the Page matrix of any particular time series, as well as that of the stacked Page matrix $\mathbf{Z}_f$.

Proposition 2.2.1. *Let Properties 2.2.1 and 2.2.2 hold. Then for any $L \leq \lfloor T/2 \rfloor$ with any $T \geq 1$, the rank of $\mathbf{Z}_f^{(n)}$ for $n \in [N]$ is at most RG. Further, the rank of the stacked Page matrix $\mathbf{Z}_f$ is also at most RG.*

The proof can be found in Appendix B.1.1 where a more general version of the Proposition is established.

Finally, we end this subsection with some further assumptions on the per-step noise in our observations. Recall that, as described in Equation 2.1, for each $n \in [N]$ and $t \in [T]$, we observe $X_n(t) = f_n(t) + \eta_n(t)$ with probability $\rho \in (0, 1]$ independently. We will assume that the noise $\eta_n(\cdot), n \in [N]$ satisfy the following property.

Property 2.2.3. *For $n \in [N], t \in [T]$, $\eta_n(t)$ are independent sub-gaussian random variables, with $\mathbb{E}[\eta_n(t)] = 0$ and $\|\eta_n(t)\|_{\psi_2} \leq \gamma$*[3]*.*

2.2.2 Extension to Approximate Low-Rank Representation

In this section, we extend the model presented in Section 2.2.1 by relaxing Property 2.2.2 to only hold *approximately*. That is, we extend our model to time series whose Hankel representation is *approximately* low-rank. More concretely, we introduce the definition of the approximate rank of a matrix below.

Definition 2.2.1 (ϵ-approximate rank)**.** *Given $\epsilon > 0$, a matrix $M \in \mathbb{R}^{a \times b}$ is said to have ϵ-approximate rank $r \geq 1$ if there exists a rank r matrix $M_r \in \mathbb{R}^{a \times b}$ such that $\|M - M_r\|_\infty < \epsilon$.*

[3]The $\|\cdot\|_{\psi_\alpha}$ denotes the sub-gaussian norm, which is defined as follows: for a random variable X, the sub-gaussian norm is defined as $\|X\|_{\psi_2} = \inf\left\{t > 0 : \mathbb{E}\exp\left(X^2/t^2\right) \leq 2\right\}$, see [54] for details.

In our extended model, we want to include time series whose Hankel representation has an ϵ-approximate rank G. In this work, we refer to such time series as (G, ϵ)-Hankel Time Series. Precisely,

Definition 2.2.2 ((G, ϵ)-Hankel Time Series)**.** *For a given $\epsilon \geq 0$ and $G \geq 1$, a time series $f : \mathbb{Z} \to \mathbb{R}$ is called a (G, ϵ)-Hankel time series if for any $T \geq 1$, its Hankel matrix has ϵ-approximate rank G.*

We extend model of Section 2.2.1 by replacing Property 2.2.2 by the following.

Property 2.2.4. *For each $r \in [R]$ and for any $T \geq 1$, the Hankel Matrix $\mathbf{H}_W^{(r)} \in \mathbb{R}^{L \times (T-L+1)}$ associated with time series $W_r(t), t \in [T]$ has ϵ-approximate rank at most G with $G \ll T$ for $\epsilon > 0$. That is, for each $r \in [R]$, W_r. is a (G, ϵ)-Hankel time series.*

Analogous to to Proposition 2.2.1, We state an implication of the new property.

Proposition 2.2.2. *Let Properties 2.2.1 and 2.2.4 hold. For any $L \leq \lfloor T/2 \rfloor$ with any $T \geq 1$, the stacked Page matrix $\mathbf{Z}_f$ has ϵ'-rank at most RG for $\epsilon' = R\Gamma_1\epsilon$.*

The proof of this property can be found in Appendix B.1.1.

2.2.3 Examples Of (G, ϵ)-Hankel Multivariate Time Series Dynamics

In this section, we establish important examples of time series dynamics that satisfy our approximate model, i.e., they are (G, ϵ)-Hankel time series. In particular, we show that linear recurrent formula (LRF) (Proposition 2.2.3), and differentiable periodic function (Proposition 2.2.4) are (G, ϵ)-Hankel multivariate time series. We further show that (G, ϵ)-Hankel time series are closed under addition and multiplication (Proposition 2.2.5).

Example 1. **(G, ϵ)-LRF Time Series.** We start by defining a linear recurrent formula (LRF), which is a standard model for linear time-invariant systems.

Definition 2.2.3 ((G, ϵ)-LRF)**.** *For $G \in \mathbb{N}$ and $\epsilon \geq 0$, a time series f is said to be a (G, ϵ)-Linear Recurrent Formula (LRF) if for all $T \in \mathbb{Z}$ and $t \in [T]$, there exists $g : \mathbb{Z} \to \mathbb{R}$ such that*

$$f(t) = g(t) + h(t),$$

where for all $t \in \mathbb{Z}$, (i) $g(t) = \sum_{l=1}^{G} \alpha_l g(t-l)$ with constants $\alpha_1, \ldots, \alpha_G$, and (ii) $|h(t)| \leq \epsilon$.

Now we establish a time series f that is a (G, ϵ)-LRF is also (G, ϵ)-Hankel.

Proposition 2.2.3. *If f is (G, ϵ)-LRF representable, then it is a (G, ϵ)-Hankel time series.*

Proof. Proof is immediate from Definitions 2.2.2 and 2.2.3. □

LRF's cover a broad class of time series functions, including any finite sum of products of harmonics, polynomials and exponentials. For example, consider a time series described by:

$$f(t) = \sum_{a=1}^{A} \exp(\alpha_a t) \cdot \cos(2\pi\omega_a t + \phi_a) \cdot P_{m_a}(t), \tag{2.3}$$

where $\alpha_a, \omega_a, \phi_a \in \mathbb{R}$ are parameters, P_{m_a} is a degree $m_a \in \mathbb{N}$ polynomial in t. It can be easily verified that is a $(G, 0)$-LRF, where $G \leq A(m_{\max}+1)(m_{\max}+2)$ with $m_{\max}$ denoting the maximum degree of the polynomials. Specifically, $m_{\max} = \max_{a \in A} m_a$ (See Proposition 5.2 in [9]).

Example 2. "Smooth" and Periodic Time Series. We establish that any differentiable periodic function is (G, ϵ)-LRF and hence (G, ϵ)-Hankel for appropriate choices of G and ϵ.

Definition 2.2.4 ($C^k(R, \mathsf{PER})$)**.** *For $k \geq 1$ and $R > 0$, we use $C^k(R, \mathsf{PER})$ to denote the class of all time series $f : \mathbb{R} \to \mathbb{R}$ such that it is R periodic, i.e. $f(t+R) = f(t)$ for all $t \in \mathbb{R}$ and the k-th derivative of f, denoted $f^{(k)}$, exists and is continuous.*

Proposition 2.2.4. *Any* $f \in C^k(R, \mathsf{PER})$ *is* $\left(4G, C(k,R)\frac{\|f^{(k)}\|}{G^{k-0.5}}\right)$*−Hankel representable, for any* $G \geq 1$*. Here* $C(k,R)$ *is a term that depends only on* k, R *and* $\|f^{(k)}\|^2 = \frac{1}{R}\int_0^R (f^{(k)}(t))^2 dt$*.*

Refer to the proof of this proposition in Appendix B.1.2.

Example 3. Sum and Product of (G,ϵ)-Hankel Time Series. We present a key property of the model class satisfying Property 2.2.4, i.e. time series that have an approximate low-rank Hankel matrix representation. To that end, we define 'addition' and 'multiplication' for time series. Given two time series $f_1,\ f_2 : \mathbb{Z} \to \mathbb{R}$, define their addition, denoted $f_1 + f_2 : \mathbb{Z} \to \mathbb{R}$ as $(f_1 + f_2)(t) = f_1(t) + f_2(t)$, for all $t \in \mathbb{Z}$. Similarly, their multiplication, denoted $f_1 \circ f_2 : \mathbb{Z} \to \mathbb{R}$ as $(f_1 \circ f_2)(t) = f_1(t) \times f_2(t)$, for all $t \in \mathbb{Z}$. Now, we state a key property for the model class satisfying Property 2.2.4.

Proposition 2.2.5. *For* $i \in \{1,2\}$*, let* f_i *be a* (G_i, ϵ_i)*-Hankel time series for* $G_i \geq 1,\ \epsilon_i \geq 0$*. Then,* $f_1 + f_2$ *is a* $(G_1 + G_2, \epsilon_1 + \epsilon_2)$*-Hankel time series and* $f_1 \circ f_2$ *is a* $\Big(G_1 G_2, 3\max(\epsilon_1, \epsilon_2) \cdot \max(\|f_1\|_\infty, \|f_2\|_\infty)\Big)$*-Hankel time series.*

Refer to the proof of this proposition in Appendix B.1.3.

Chapter 3

Algorithm

In this chapter, motivated by the model described in Section 2.2, we describe the proposed variant of the multivariate Singular Spectrum Analysis (mSSA) algorithm. In Section 3.1, we describe the mean imputation and mean forecasting algorithm, which provides an estimate of $f_n(t)$, the latent time series of interest. In Section 3.2, we describe the variance imputation and variance forecasting algorithm, which provides an estimate of $\sigma_n^2(t)$, the latent time-varying variance of the noise. In Section 3.3, we describe some of the concerns regarding applying this algorithm in practice. Specifically, we discuss the algorithm's scalability, how it can be updated incrementally, and when it is suitable to be applied to real-world data.

3.1 Mean Estimation Algorithm

Multivariate singular spectrum analysis (mSSA) is a known method to impute and forecast a multivariate time series. Below, we describe a novel and practically useful variant of mSSA, which naturally allows for both imputation and forecasting. Note that in Section 1.2, we compare the original mSSA method with this variant and discuss key differences.

Recall that $X_n(t), n \in [N], t \in [T]$ denote the (potentially noisy, missing) observation of the n-th time series, X_n, at the t-th time step. For any $t, s \in [T]$ such that $t < s$, let $X_n(t:s) := [X_n(t), \ldots, X_n(s)]$. Further, Let integers $L, P \geq 1$ be such that

$P = \lfloor T/L \rfloor$. Let $\bar{P} := N \times P$. Let $\widehat{\rho}$ denotes the fraction of observed entries, that is $\widehat{\rho} := \frac{\max(1, \sum_{n=1}^{N} \sum_{t=1}^{T} \mathbb{1}(X_n(t) \neq \star))}{NT}$

Imputation. The key steps of mean imputation are as follows.

1. (Form Page Matrix) Transform $X_1(1:T), \ldots, X_N(1:T)$ into a stacked Page matrix [1] $\mathbf{Z}_X \in \mathbb{R}^{L \times \bar{P}}$ with $L \leq \bar{P}$. Fill all missing entries in the matrix by 0.
2. (Singular Value Thresholding) Let SVD of $\mathbf{Z}_X = \mathbf{U}\mathbf{S}\mathbf{V}^T$, where $\mathbf{U} \in \mathbb{R}^{L \times L}$, $\mathbf{V} \in \mathbb{R}^{\bar{P} \times L}$ represent left and right singular vectors and $\mathbf{S} = \text{diag}(s_1, \ldots, s_L)$ the diagonal matrix of singular values $s_1 \geq \cdots \geq s_L \geq 0$. Obtain $\widehat{\boldsymbol{M}}_f = \frac{1}{\widehat{\rho}} \mathbf{U}\mathbf{S}_k\mathbf{V}^T$ by setting all but top k singular values to 0, i.e. $\mathbf{S}_k = \text{diag}(s_1, \ldots, s_k, 0, \ldots, 0)$ for some $k \in [L]$.
3. (Output) $\hat{f}_n(i + (j-1)L) := \left[\widehat{\boldsymbol{M}}_f\right]_{i,[j+P(n-1)]}$, $i \in [L], j \in [P]$.

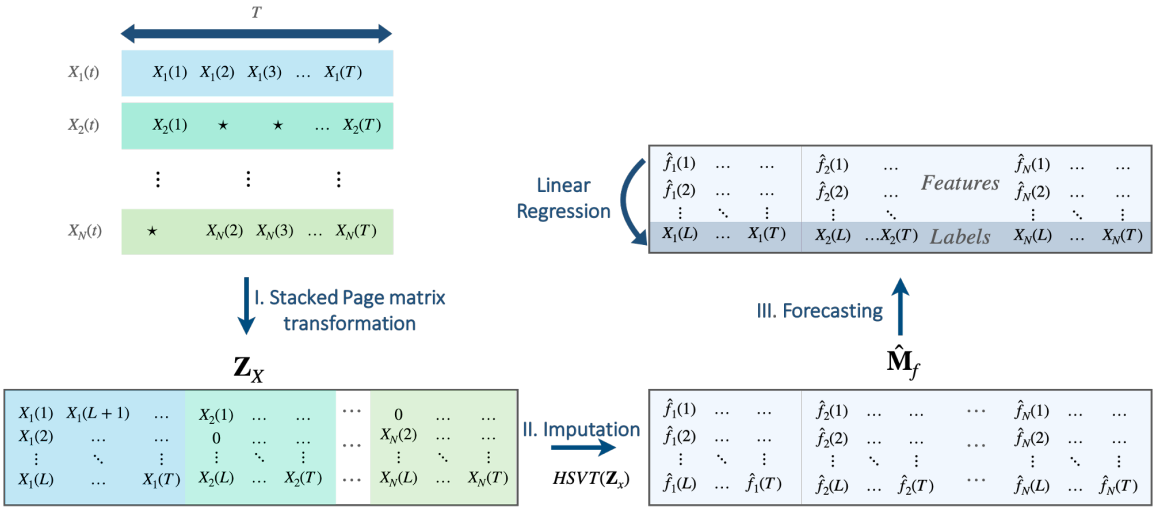


Figure 3-1: Key steps of our proposed variant of the mSSA algorithm.

Forecasting. Forecasting includes an additional step of fitting a linear model on the de-noised matrix.

1. (Form Sub-Matrices) Let $\bar{\mathbf{Z}}_X \in \mathbb{R}^{L-1 \times \bar{P}}$ be a sub-matrix of $\mathbf{Z}_X$ obtained by removing its last row. Let $\mathbf{Z}_X^L$ denote the last row.
2. (Singular Value Thresholding) Let SVD of $\bar{\mathbf{Z}}_X = \bar{\mathbf{U}}\bar{\mathbf{S}}\bar{\mathbf{V}}^T$, where $\bar{\mathbf{U}} \in \mathbb{R}^{L-1 \times L-1}$, $\bar{\mathbf{V}} \in \mathbb{R}^{\bar{P} \times L-1}$ represent left and right singular vectors and $\bar{\mathbf{S}} = \text{diag}(s_1, \ldots, s_{L-1})$ the diagonal matrix of singular values $\bar{s}_1 \geq \cdots \geq \bar{s}_{L-1} \geq 0$. Obtain $\widehat{\bar{\boldsymbol{M}}}_f = \frac{1}{\widehat{\rho}} \bar{\mathbf{U}}\bar{\mathbf{S}}_k\bar{\mathbf{V}}^T$

[1] Refer to the definition of the stacked Page matrix in Equation 2.2 and Definition 2.1.2

by setting all but top k singular values to 0, i.e. $\bar{\mathbf{S}}_k = \text{diag}(\bar{s}_1, \ldots, \bar{s}_k, 0, \ldots, 0)$ for some for some $k \in [L-1]$.

3. (Linear Regression) $\hat{\beta} = \arg\min_{b \in \mathbb{R}^{L-1}} \|\mathbf{Z}_X^L - (\widehat{\boldsymbol{M}}_f)^T b\|_2^2$.
4. 4. (Output) $\bar{f}_n(T+1) \coloneqq X_n(T-(L-1):T)^T \hat{\beta}$.

Figure 3-1 provides a visual depiction of the core steps of both the imputation and forecasting algorithm.

3.2 Variance Estimation Algorithm

In this section, we extend our mSSA variant to estimate time-varying variance. The procedure is motivated by the following simple observation. Note that mSSA is designed to estimate the underlying latent 'mean' time series from its noisy, partial observation. For example the mean of the observations for time series $n \in [N]$ at time $t \in [T]$ is $\mathbb{E}[X_n(t)] = f_n(t)$. Instead, if we apply mSSA to the squared observations, $X_n^2(t)$, we will recover an estimate of $\mathbb{E}[X_n^2(t)]$. However, observe that $\mathbb{V}\text{ar}[X_n(t)] = \mathbb{E}[X_n^2(t)] - \mathbb{E}[X_n(t)]^2$. Therefore, by applying mSSA twice, once on $X_n(t)$ and once on $X_n^2(t)$ for $n \in [N]$ and $t \in [T]$, and subsequently taking the component-wise difference of the estimates produced will lead to an estimate of the variance. This suggests a simple algorithm for variance estimation, where we run the mean estimation algorithm twice, once on $\mathbf{Z}_X$ and once on $\mathbf{Z}_{X^2}$ (the stacked page matrix of the squared observations). Then, for both forecasting and imputation, a simple post-processing step is done where the square of the estimate produced from running the algorithm on $\mathbf{Z}_X$ is subtracted from the estimate produced from $\mathbf{Z}_{X^2}$. The procedure is detailed precisely in what follows.

Imputation. Below are steps of variance imputation.

1. (Impute $\mathbf{Z}_X, \mathbf{Z}_{X^2}$) Use the mean imputation algorithm on $X_1(1:T), \ldots, X_N(1:T)$ (i.e., $\mathbf{Z}_X$) and $X_1^2(1:T), \ldots, X_N^2(1:T)$ (i.e., $\mathbf{Z}_{X^2}$), to produce the de-noised Page matrices $\widehat{\boldsymbol{M}}_f$ and $\widehat{\boldsymbol{M}}_{f^2+\sigma^2}$, respectively.
2. (Output) Construct $\widehat{\boldsymbol{M}}_{f^2} \in \mathbb{R}^{L \times \bar{P}}$, where $\left[\widehat{\boldsymbol{M}}_{f^2}\right]_{ij} \coloneqq \left(\left[\widehat{\boldsymbol{M}}_f\right]_{ij}\right)^2$, and produce

estimates,

$$\hat{\sigma}_n^2(i+(j-1)L) \coloneqq \left[\widehat{\boldsymbol{M}}_{f^2+\sigma^2}\right]_{i,[j+P\times(n-1)]} - \left[\widehat{\boldsymbol{M}}_{f^2}\right]_{i,[j+P\times(n-1)]} \quad \forall i \in [L], j \in [P].$$

Forecasting. The variance forecasting procedure is similar to that of mean forecasting, except for an additional post processing step. Precisely,

1. (Form Sub-Matrices) Let $\bar{\mathbf{Z}}_X \in \mathbb{R}^{L-1\times\bar{P}}$ be a sub-matrix of $\mathbf{Z}_X$ obtained by removing its last row. Let $\mathbf{Z}_X^L$ denote the last row. Similarly, let $\bar{\mathbf{Z}}_{X^2} \in \mathbb{R}^{L-1\times\bar{P}}$ be a sub-matrix of $\mathbf{Z}_{X^2}$ obtained by removing its last row, and denote that last row by $\mathbf{Z}_{X^2}^L$.
2. (Forecast with $\bar{\mathbf{Z}}_X, \mathbf{Z}_X^L, \bar{\mathbf{Z}}_{X^2}, \mathbf{Z}_{X^2}^L$) Using the mean forecasting algorithm on $X_1(1:T), \ldots, X_N(1:T)$ and $X_1^2(1:T), \ldots, X_N^2(1:T)$, to produce forecast estimates $\bar{f}_n(T+1)$ and $\overline{f_n^2+\sigma_n^2}(T+1)$, respectively.
3. (Output) Produce the variance estimate

$$\bar{\sigma}_n^2(T+1) \coloneqq \overline{f_n^2+\sigma_n^2}(T+1) - (\bar{f}_n(T+1))^2.$$

3.3 mSSA in Practice

3.3.1 Limitations

The algorithms for mean and variance estimation described above, as written, are meant for batch updating (i.e., they get to observe all data at once). However, for computational efficiency in any real-world application, the prediction model needs to be trained and updated incrementally, which in turn requires making these algorithms incremental. These limitations are the main motivation of building a scalable incremental version of this algorithm, as described in Chapter 5.

In particular, to make mSSA incremental, we propose a simple incremental "meta"-algorithm. In this meta-algorithm, the model is updated *incrementally* and as new data points arrive in an efficient and scalable manner. We defer the description of this algorithm to Section 5.3, where we discuss this incremental variant and its integration

with a time series database system to create *tspDB: time series predict database*.

3.3.2 A Diagnostic Test for Real-World Data

Despite showing that the proposed model admits a rich families of time series dynamics, it is vital to have a data-driven diagnostic test that can help identify scenarios when the model of Section 2.2.1 hold. We discuss one such test here.

Recall that the primary representation utilized by mSSA, as described next in Section 3.1, is the stacked Page matrix (with parameter L). Proposition 2.2.1 and Proposition 2.2.2 suggests a "data driven diagnosis test" to verify whether mSSA is likely to succeed as per the results of this work. Specifically, if the (effective) rank[2] of the Page matrix associated with any of the univariate components $f_n(\cdot)$ and the (effective) rank of stacked Page matrix associated with the multivariate time series with N component are *very different, then mSSA may not be effective* compared to SSA, but if they are *very similar then mSSA is likely to be more effective* compared to SSA. Our finite-sample results in Chapter 4 indicate that the optimal value for L is $\sqrt{NT}$ assuming $N < T$. Thus as a further test, if the effective rank of the stacked Page matrix does not scale much slower than L for $L \sim \sqrt{NT}$, then SSA (and mSSA) are unlikely to be effective methods.

Table 3.1 compares the (effective) rank of the stacked Page matrices for different benchmark time series data sets. The value of T equals 3993, 26304 and 10560 for the Financial, Electricity and Traffic datasets respectively (see Section 6.1.1 for details on the datasets). We set $L = \lfloor \sqrt{NT} \rfloor$ for all datasets. When $N = 1$, this corresponds to L equals 63, 162 and 102 for the Financial, Electricity and Traffic datasets respectively. Table 3.1 shows the effective rank in each dataset as we vary N. As can be seen, for $N = 1$, the effective rank is much smaller than L (or T) suggesting that SSA is likely to be effective. For Electricity and Financial datasets, the rank does not change by much as we increase N. However, relatively the rank does increase substantially for the Traffic dataset. This might explain why mSSA

[2]We define effective rank of a matrix as the minimum number of singular values capturing $> 90\%$ of its spectral energy.

is relatively less effective for the Traffic dataset in contrast to the Financial and Electricity datasets as seen in Table 6.2 in Section 6.2.

Table 3.1: Effective rank of stacked Page matrix across benchmarks as we vary N.

Dataset	N = 1	N =10	N = 100	N = 350
Electricity	19	37	44	31
Financial	1	3	3	6
Traffic	14	32	69	116

Chapter 4

Theoretical Results

In this chapter, we provide the main theoretical results for the introduced mSSA variant. Specifically, we provide bounds on the imputation and forecasting *in-sample prediction error* for mSSA under the spatio-temporal model introduced in Chapter 2. We state the mean squared error bounds for both *mean* imputation and *mean* forecasting in Section 4.1 using both the exact low-rank representation (Property 2.2.2) and the approximate low-rank representation (Property 2.2.4). In Section 4.2, we extend the results to the *variance* imputation and forecasting problem, where we first impose some structure on the time-varying variance $\sigma_n^2(t)$ and then provide mean squared error bounds for both *variance* imputation and *variance* forecasting algorithms.

4.1 Mean Estimation Error

4.1.1 Metrics and Setup

We start the section by defining the metric we use to measure prediction error. In both imputation and forecasting, we use the mean squared error as our metric. Precisely, let the imputed values be denoted by $\hat{f}_n(t),\ n \in [N],\ t \in [T]$, then we define the prediction error for imputation as

$$\mathsf{ImpErr}(N,T) = \frac{1}{NT}\sum_{n=1}^{N}\sum_{t=1}^{T}\mathbb{E}\big[(f_n(t)-\hat{f}_n(t))^2\big]. \tag{4.1}$$

Recall that the precise steps of the imputation algorithm are detailed in Section 3.1.

For forecasting, we define the (in-sample) prediction error as the sum of prediction errors for the last row of the stacked Page matrix. More concretely, let that the (in-sample) forecasted estimate be denoted by $\bar{f}_n(t),\ n \in [N],\ t \in \{L, 2L, \ldots, T\}$, then we define the prediction error for forecasting as

$$\mathsf{ForErr}(N,T,L) = \frac{L}{NT}\sum_{n=1}^{N}\sum_{m'=1}^{T/L}\mathbb{E}\big[(f_n(L\times m')-\bar{f}_n(L\times m'))^2\big]. \tag{4.2}$$

Again, recall that the precise steps of the of the forecasting algorithm are detailed in Section 3.1. In (4.1) and (4.2), the expectation is with respect to the randomness in observations due to noise and missing values. Before stating the main results, we will utilize the following model restriction.

Property 4.1.1 (Balanced spectra)**.** *Denote the $L \times (NT/L)$ stacked Page matrix associated with all N time series $f_1(\cdot), \ldots, f_N(\cdot)$ as $\mathbf{Z}_f$. Under the setup of Proposition 2.2.1, $\mathrm{rank}(\mathbf{Z}_f) = k \geq 1$ and $k \leq R \times G$. Then, for $L = \sqrt{NT}$ with $N \leq T$, $\mathbf{Z}_f$ is such that $\sigma_k(\mathbf{Z}_f) \geq c\sqrt{NT}/\sqrt{k}$ some absolute constant $c > 0$, where σ_k is the k-th largest singular value of $\mathbf{Z}_f$.*

Note that if $\sigma_k = \Theta(\sigma_1)$, then one can verify that Property 4.1.1 holds. Indeed, assuming that the non-zero singular values are 'well-balanced' is standard in the matrix/tensor estimation literature.

Additionally, we will utilize an analogous property under the approximate low-rank setup of Proposition 2.2.2.

Property 4.1.2 (Approximately balanced spectra)**.** *Under the setup of Proposition 2.2.2, we can represent the $L \times (NT/L)$ stacked Page matrix associated with all N time series $f_1(\cdot), \ldots, f_N(\cdot)$ as $\mathbf{Z}_f = \tilde{\boldsymbol{M}} + \boldsymbol{E}$ with $\mathrm{rank}(\tilde{\boldsymbol{M}}) = k \geq 1$ and $k \leq R \times G$*

and $\|\boldsymbol{E}\|_\infty \leq R\Gamma_1\epsilon$. *Then, for* $L = \sqrt{NT}$, $\tilde{\boldsymbol{M}}$ *is such that* $\sigma_k(\tilde{\boldsymbol{M}}) \geq c\sqrt{NT}/\sqrt{k}$ *for some absolute constant* $c > 0$, *where* σ_k *is the* k*-th largest singular value of* $\tilde{\boldsymbol{M}}$.

Property 4.1.2 is more general than Property 4.1.1, where the balanced spectra is attributed to the *low-rank approximation* of the stacked page matrix.

4.1.2 Imputation

In this section, we state the main results related to the prediction error of the mean imputation algorithm. In what follows, let $C(c, \Gamma_1, \Gamma_2, \gamma)$ be a constant that depends only on model parameters $c, \Gamma_1, \Gamma_2, \gamma$. Recall that R, Γ_1, Γ_2 are parameters of the spatio-temporal factor model, and are defined in Property 2.2.1; while G is the maximum rank of the Hankel matrices induced by the "fundamental" time series, as defined in 2.2.2; and γ is a parameter of the noise, and is defined in Property 2.2.3; and c is a parameter of the balanced spectra, and is defined in Property 4.1.1.

Theorem 4.1.1 (Imputation)**.** *Let Properties 2.2.1, 2.2.2, 2.2.3 and 4.1.1 hold. For a large enough absolute constant* $C > 0$, *let* $\rho \geq C\frac{\log NT}{\sqrt{NT}}$. *Then with* $L = \sqrt{NT}$,

$$\mathsf{ImpErr}(N, T) \leq C(c, \Gamma_1, \Gamma_2, \gamma)\left(\frac{R^3 G \log NT}{\rho^4\sqrt{NT}}\right) \tag{4.3}$$

More generally, let the approximate spatio-temporal factor model hold. That is, let Properties 2.2.1, 2.2.4, 2.2.3 and 4.1.2 hold. Then,

$$\mathsf{ImpErr}(N, T) \leq C(c, \Gamma_1, \Gamma_2, \gamma)\left(\frac{R^3 G \log NT}{\rho^4\sqrt{NT}} + \frac{R^4 G(\epsilon + \epsilon^2)}{\rho^2}\right) \tag{4.4}$$

The imputation error bound in 4.4 scale as, $\mathcal{O}\left(\frac{\log NT}{\sqrt{NT}} + (\epsilon + \epsilon^2)\right)$. The first term vanishes as we collect more data, while the second term is due to the approximate low-rank assumption. The proof of this theorem is in Appendix B.2.1

4.1.3 Forecasting

In this section, we state the main results for the mean forecasting algorithm. We start by proving that given our spatio-temporal factor model, there exists a linear model that provides good forecasting estimates. After that, we give a formal results of the (in-sample) forecasting error.

Existence of Linear Model for Forecasting. Recall from Section 3.1, we learn a linear regression model between the last row of that stacked Page matrix $\mathbf{Z}_X$ and the $L-1$ rows above it (after de-noising the sub-matrix induced these $L-1$ rows via HSVT). Hence, we first establish that there exists a linear model which provides a good solution in the case where theres is no noise or missing values. To that end, let $\mathbf{Z}_f^L$ denote the Lth row of $\mathbf{Z}_f$ and $\bar{\mathbf{Z}}_f$ denote the sub-matrix of $\mathbf{Z}_f$ formed by selecting top $L-1$ rows. In the proposition below, we show that there exists a linear relationship between $\mathbf{Z}_f^L$ and $\bar{\mathbf{Z}}_f$.

Proposition 4.1.1. *Let Properties 2.2.1 and 2.2.2 hold. Then there exists $\beta^* \in \mathbb{R}^{L-1}$ such that $(\mathbf{Z}_f^L)^T = (\bar{\mathbf{Z}}_f)^T\beta^*$. Further, $\|\beta^*\|_0 \leq RG$.*

For the approximate low-rank setting, we show that the existence of a linear model using the following analogous Proposition.

Proposition 4.1.2. *Let Properties 2.2.1 and 2.2.4 hold. Then, there exists $\beta^* \in \mathbb{R}^{L-1}$, such that $\|(\mathbf{Z}_f^L)^T - (\bar{\mathbf{Z}}_f)^T\beta^*\|_\infty \leq R\Gamma_1(1+\|\beta^*\|_1)\epsilon$., Further $\|\beta^*\|_0 \leq RG$.*

The proof for Proposition 4.1.2, which is a more general form of Proposition 4.1.1, is in Appendix B.2.3. We now state the main result for the mean forecasting error for mSSA.

Theorem 4.1.2 (Forecasting)**.** *Let Properties 2.2.1, 2.2.2, 2.2.3 and 4.1.1 hold. Then, with $L = \sqrt{NT}$ and with β^* defined in Proposition 4.1.1, we have*

$$\mathsf{ForErr}(N,T,L) \leq C(c,\gamma,\Gamma_1,\Gamma_2)\max(1,\|\beta^*\|_1^2)\Big(\frac{R^3G\log NT}{\rho^4\sqrt{NT}}\Big). \tag{4.5}$$

More generally, let the approximate spatio-temporal factor model hold. That is, let Properties 2.2.1, 2.2.4, 2.2.3 and 4.1.2 hold. Then, with β^* defined in Proposition 4.1.2, we have

$$\begin{aligned}&\mathsf{ForErr}(N,T,L) \leq \\ &C(c,\gamma,\Gamma_1,\Gamma_2)\max(1,\|\beta^*\|_1,\|\beta^*\|_1^2)\Big(\frac{R^3 G\log NT}{\rho^4\sqrt{NT}}+\frac{R^4 G(\epsilon+\epsilon^2)}{\rho^2}\Big).\end{aligned} \tag{4.6}$$

Similar to the imputation bound, the error bound in scale as, $\mathcal{O}\left(\frac{\log NT}{\sqrt{NT}}+(\epsilon+\epsilon^2)\right)$. The first term vanishes as we collect more data, while the second term is due to the approximate low-rank assumption. We finally remark that the proof of Theorem 4.1.2 is in Appendix B.2.2.

4.2 Variance Estimation Error

As detailed in Section 3.2, we extend our mSSA variant to estimate time-varying variance. To facilitate the analysis of the variance imputation and forecasting algorithm, we first describe the model we assume to govern the time-varying variance, which we describe next.

4.2.1 Setup

The model we describe for the time-varying variance is analogous to the spatio-temporal model we described in Chapter 2 for the latent mean of the time series. Specifically, for $n \in [N],\ t \in [T]$, let $\sigma_n^2(t) = \mathbb{E}[\eta_n^2(t)]$ be the time-varying variance of the time series observations, i.e. if $\rho = 1$ then $\sigma_n^2(t) = \mathbb{V}\mathrm{ar}[X_n(t)] = \mathbb{E}[X_n^2(t)] - f_n^2(t)$. To capture the "spatial" and "temporal" structure across the N latent time-varying variances, we assume that the latent variance time series $\sigma_1^2(t),\ldots,\sigma_N^2(t)$ satisfies Properties 4.2.1 and 4.2.2. These properties are analogous to those assumed about the latent time series $f_1(t),\ldots,f_N(t)$ in Properties 2.2.1 and 2.2.2.

Property 4.2.1. *There exist $R' \in \mathbb{N}$, where $1 \leq R' \ll \min(N,T)$, $W_r' : \mathbb{Z} \to$*

$\mathbb{R}$, *and* $\alpha'_{nr} \in \mathbb{R}$ $\forall r \in [R'], n \in [N]$, *such that for any* $n \in [N], t \in [T]$, $\sigma_n^2(t) = \sum_{r=1}^{R'} \alpha'_{nr}\ W'_r(t)$, *where* $|\alpha'_{nr}| \leq \Gamma'_1$, $|W'_r(t)| \leq \Gamma'_2$ *for constants* $\Gamma'_1, \Gamma'_2 > 0$.

Like Property 2.2.1, the above property captures the "spatial" structure within the N time series describing the latent variance. To capture the "temporal" structure, next we introduce an analogue of Property 2.2.2.

To that end, for each $r \in [R']$, let $\mathbf{H}_{W'}^{(r)} \in \mathbb{R}^{L \times (T-L+1)}$ denote the Hankel matrix induced by $W'_r(\cdot)$, as described in Definition 2.1.1.

Property 4.2.2. *For each* $r \in [R']$, *the Hankel Matrix* $\mathbf{H}_{W'}^{(r)}$ *induced by the time series* $W'_r(t), t \in [T]$ *has rank at most* G' *with* $G' \ll T$.

Before stating the variance imputation results, we introduce the following additional property, which is analogous to Property 4.1.1.

Property 4.2.3 (Balanced spectra)**.** *Denote the* $L \times (NT/L)$ *stacked Page matrix associated with all* N *time series* $\sigma_1^2(\cdot), \ldots, \sigma_N^2(\cdot)$ *by* $\mathbf{Z}_\sigma$. *Due to Properties 4.2.1 and 4.2.2, and a simple variant of Proposition 2.2.1, we have rank*$(\mathbf{Z}_\sigma) = k' \geq 1$ *and* $k' \leq R' \times G'$. *Then, for* $L = \sqrt{NT}$, $\mathbf{Z}_\sigma$ *is such that* $\sigma'_k(\boldsymbol{M}) \geq c\sqrt{NT}/\sqrt{k'}$ *for some absolute constant* $c > 0$, *where* σ'_k *is the* k*-th singular value, ordered by magnitude, of* $\mathbf{Z}_\sigma$.

Finally, similar to the mean estimation setup, we need a property that establishes that there exists a linear model for the latent variance time series. Let $\mathbf{Z}_{f^2+\sigma^2}$ be the stacked Page matrix induced by the time series $f_1^2(\cdot) + \sigma_1^2(\cdot), \ldots, f_N^2(\cdot) + \sigma_N^2(\cdot)$. Let $\mathbf{Z}^L_{f^2+\sigma^2}$ denote the Lth row of $\mathbf{Z}_{f^2+\sigma^2}$ and $\bar{\mathbf{Z}}_{f^2+\sigma^2}$ denote the sub-matrix of $\mathbf{Z}_{f^2+\sigma^2}$ formed by selecting top $L-1$ rows. Similar to Proposition 4.1.1, we show exists a linear relationship between $\mathbf{Z}^L_{f^2+\sigma^2}$ and $\bar{\mathbf{Z}}_{f^2+\sigma^2}$. For proof, we refer the reader to the proof of property 4.1.2, of which the following property is a special case.

Proposition 4.2.1. *Let Properties 2.2.1, 2.2.2, 4.2.1 and 4.2.2 hold. Then there exists* $\beta^* \in \mathbb{R}^{L-1}$ *such that* $(\mathbf{Z}^L_{f^2+\sigma^2})^T = (\bar{\mathbf{Z}}_{f^2+\sigma^2})^T \beta^*$. *Further,* $\|\beta^*\|_0 \leq (RG)^2 + R'G'$.

4.2.2 Variance Results

We now establish the estimation error for the variance estimation algorithm under the spatio-temporal model above.

Theorem 4.2.1 (Variance Imputation)**.** *Let Properties 2.2.1, 2.2.2, 2.2.3, 4.1.1, 4.2.1, 4.2.2, and 4.2.3 hold. Additionally let* $|\hat{f}_n(t)| \leq \Gamma_3$ *for all* $n \in [N], t \in [T]$*. Lastly, let* $L = \sqrt{NT}$ *and* $\rho = 1$*. Then the variance prediction error is bounded above as*

$$\frac{1}{NT}\sum_{n=1}^{N}\sum_{t=1}^{T}\mathbb{E}[\left(\sigma_n(t)^2 - \hat{\sigma}_n^2(t)\right)^2] \leq \tilde{C}\left(\frac{G^2G'\log^2 NT}{\sqrt{NT}}.\right). \tag{4.7}$$

where $\tilde{C}$ *is a constant dependent (polynomially) on model parameters* $c, \Gamma_1, \Gamma_2, \Gamma_3, \Gamma_1', \Gamma_2', \gamma, R, R'$*.*

Theorem 4.2.2 (Variance Forecasting)**.** *Let conditions of Theorem 4.2.1 hold. And let* β_1^*, β_2^* *be defined as in Proposition 4.1.1 and Proposition 4.2.1 respectively.Then the variance forecasting error is bounded above as*

$$\frac{L}{NT}\sum_{n=1}^{N}\sum_{m'=1}^{T/L}\mathbb{E}\big[(\sigma_n^2(L\times m') - \bar{\sigma}_n^2(L\times m'))^2\big] \leq \tilde{C}\Big(\max(1, \|\beta_2^*\|_1^2, \|\beta_1^*\|_1^2)\left(\frac{G^2G'\log^2 NT}{\sqrt{NT}}\right)\Big).$$

where $\tilde{C}$ *is a constant dependent (polynomially) on model parameters* $c, \Gamma_1, \Gamma_2, \Gamma_3, \Gamma_1', \Gamma_2', \gamma, R, R'$*.*

Proof of Theorems 4.2.1 and 4.2.2 can be found in Appendix B.2.4.

Chapter 5

tspDB: A Scalable Time Series Prediction System

An important goal in the machine learning community is to make ML more broadly accessible to decision-makers and domain experts [43]. Arguably, the major bottleneck towards this goal is not the lack of access to prediction algorithms, for which many excellent open-source ML libraries exist. Rather, it is the complex engineering and data processing required to transform raw data into actionable insights; and to do that in a scalable and efficient manner. In this chapter, we present tspDB: the time series predict DB. In tspDB, we explore the feasibility of directly integrating a prediction system on top of the DB layer, making predictions in the context of time series accessible through simple SQL-like interface.

In Section 5.1, we further motivate why such integration is needed; In Section 5.2, we provide a description of tspDB's easy-to-use interface; In Section 5.3, we detail the algorithmic modification needed to enable the system's performance; Finally, in Section 5.4, we describe an instantiation of tspDB through the integration with PostgreSQL.

5.1 Motivation

Need For tspDB. An important goal in the Systems for ML community, as stated in the white paper of the Conference on Machine Learning and Systems, has been to make ML more broadly accessible [43]. Arguably, the major bottleneck towards this goal is not the lack of access to prediction algorithms, for which a plethora of excellent open-source ML libraries exist. Rather, it is the complexity of the machine learning workflow. For example, the data engineering and processing required to take data from a datastore or database (DB) into a particular work environment format (e.g. spark data-frame) so that a prediction algorithm can be trained in a scalable manner is a challenging task without a team of data scientists. See Figure 5-1 for a simplified visual depiction of the ML workflow as it stands; we aim to alleviate much of this "unwieldy" data engineering and processing required by building tspDB, a system that directly integrates prediction functionality on top of a DB in real-time.

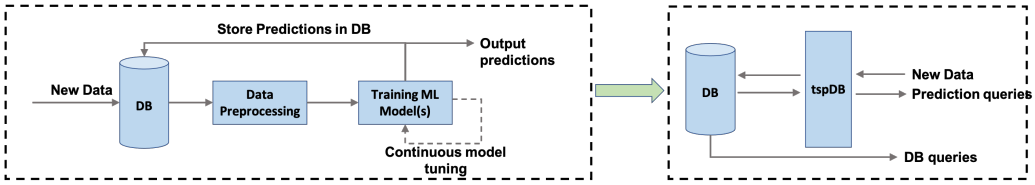


Figure 5-1: Pictorial depiction of data engineering—data transformation, feature extraction, model training/tuning—as a major bottleneck of the ML workflow. tspDB aims to alleviate it by direct integration of prediction functionality on top of a DB.

Towards easing this bottleneck, and increasing accessibility of ML, our objective is to explore the feasibility of directly integrating a prediction system on top of the DB layer. This is in line with recent efforts to "democratize" ML [51, 49]. Indeed, the authors in [11] make a compelling case of the potential gains to be had by direct integration of predictive functionality on top of DBs—by drawing an analogy with the now scalable, robust and mature data management capabilities of DBs, they argue that a similar approach to the management of predictive models can lead to large improvements in both computational performance and accessibility of ML.

Operationalizing mSSA. In this work, we *operationalize* the proposed mSSA algorithm to provide predictive capabilities directly in the DB. The choice of the mSSA

algorithm as the predictive "engine" of tspDB is not arbitrary; rather there are many attractive properties of mSSA that justify its use. In particular, in addition to it being a principled approach with sound theoretical guarantees, mSSA has the following *practical* properties:

1. **mSSA is universal.** In a time series database, we may not have control nor prior information about the generating process of the time series. However, in modern databases, the performance is often agnostic to the data generating process. For example, querying and inserting data should be efficient for most real-world datasets and is not highly dependent on the data generating process[1]. Thus, it is natural to expect that the predictive capabilities, once integrated in the database, should work well for time series of different types (e.g., floats, integers, binary) and generating processes (e.g., harmonics, polynomial trend, ARMA process). The theoretical results for mSSA hold for a rich model of multivariate time series that include a wide variety of time series dynamics (See Chapter 2, which makes it suitable to our use case. In addition, an important feature of our mSSA variant is that it is also "noise agnostic", i.e., it effectively imputes and forecasts both the time-varying mean and variance *regardless* of the noise model—e.g. Gaussian noise (continuous observations), Poisson and Binomial noise (integer observations), Bernoulli noise (binary observations). See Section 6.2.3 for an empirical validation of this claim.
2. **mSSA supports uncertainty quantification.** In any real-world application of time series analysis, along with estimating the underlying mean, quantifying prediction uncertainty (i.e., providing confidence intervals) is necessary to make meaningful decisions. As detailed in Section 3.2, we extend mSSA to estimate the underlying time-varying variance, and hence using mSSA as the predictive engine of tspDB provides the crucial uncertainty quantification property.
3. **mSSA is scalable and incremental.** One important aspect of our system design is to minimize the computational overhead added by tspDB to the orig-

[1]The statistical properties of the data will naturally have an effect on some functionalities such as the performance of some indices.

inal DB system (e.g., PostgreSQL). For that, we need our algorithm to be (i) scalable with the size of data; and (ii) updated incrementally and efficiently as new data-points come in. While the mSSA algorithm as described in Chapter 3 does not support incremental updates nor does it scale well with the size of data, one can use a scalable and incremental variant of mSSA that has excellent statistical and computational performance, as we propose in Section 5.3.

4. **mSSA is statistically performant.** Through extensive benchmarking, we show that mSSA has a superior statistical performance against popular state-of-the-art prediction libraries –results of these experiments are summarized in Chapter 6.

5.2 Prediction Interface

An important goal of tspDB is to design an 'easy-to-use' interface to make predictions, which: (i) alleviates the need for error-prone data engineering; (ii) directly gives access to predictive functionality in real-time for time series data. In particular, we wish to provide *easy* access to imputation and forecasting functionality for multivariate time series data. Towards this goal, there has been considerable recent work, especially in industry, exploring the feasibility of DBs to automate and natively support ML workloads ([1, 4, 5, 3, 6, 2]). The focus of these works has been to expose an interface that allows users to select from an array of ML algorithms (e.g. generalized linear models, random forests, neural networks) and train them (plus tune hyper-parameters) in the DB itself.

In contrast, a significant point of departure of tspDB is how an end user interfaces with the system to produce predictions. In particular, to make ML more broadly accessible, we take a different approach and abstract the ML model from the user, and instead *strive for a single interface to answer both standard DB queries and predictive queries*. In particular, in tspDB, a predictive query has the same form as a standard SELECT query. The key difference is that in a predictive query, the response is a *prediction*, rather than a retrieval of available data. For example, consider a

relation of the stock price of three companies over 100 days: `stock(day:timestamp, company1:float, company2:float, company3:float)`. An example of a predictive query in this setting is shown in Figure 5-2a. Note, day 101 is a forecast, i.e., a prediction, since the DB only contains data for the first 100 days. Similarly, if the query is changed to `WHERE day = 10`, then we get the imputed (or de-noised) value for the stock price on the tenth day. In this case, the PREDICT query response differs from a standard SELECT query as rather than simply retrieving the available data for that day, which may possibly even be missing, a predicted de-noised value is returned.

To enable PREDICT queries, we need to build a prediction model using the available multivariate time series data. Continuing the example from above, a user can build a prediction model in tspDB as shown in Figure 5-2b using the CREATE query in tspDB. Note, importantly, the prediction model can be trained over multiple time series columns, i.e., is built by simultaneously using data from multiple time series, in this case the stock prices for the three companies. Thus, the CREATE and PREDICT queries in tspDB are very much like building a DB index and making a SELECT query in SQL.

```
PREDICT company1
FROM stock
WITH PREDICTION_INTERVAL = 95%
WHERE day = 101;
```

(a) A PREDICT query in tspDB.

```
CREATE PREDICTION_MODEL
ON stock(company1, company2,
   company3)
WITH day AS TIME_COLUMN;
```

(b) Building a prediction model in tspDB.

Figure 5-2: Proposed interface for tspDB

5.3 Algorithm: A Scalable and incremental Variant

As mentioned in Section 3.3.1, the mSSA variant we propose in Chapter 3 for mean and variance estimation are meant for batch updating (i.e., they get to observe all data at once). However, for computational efficiency in any real-world application, the prediction model need to be built and updated incrementally, which in turn requires

making these algorithms incremental. Specifically, (i) their computational complexity should scale well with volume of data inserted; (ii) statistical accuracy should not degrade as more data points are incrementally incorporated into the model. We address these challenges next.

Incremental Singular Value Decomposition. The key computationally expensive step in imputation and forecasting is computing a singular value decomposition (SVD) to do singular value thresholding (SVT) of the relevant Page matrices. Particularly, the complexity of computing the SVD of an $n \times n$ matrix is $\mathcal{O}(n^3)$. Therefore, to make the algorithms incremental, we need an incremental version of SVD. To that end, we use the incremental SVD method developed in the Latent Semantic Indexing literature [59]. This method updates the decomposition by computing a QR decomposition and SVD of a much smaller matrix, whose size depends primarily on the size of the update. Hence, the complexity of this method is much lower compared to doing the full SVD. In particular, the complexity of the SVD is $\mathcal{O}(k^3+(L+\bar{P})k^2+(L+\bar{P})kp+p^3)$, where k is the number of retained singular values, p is the number of new added columns, L, $\bar{P}$ are the number of rows and columns for the stacked Page matrix, respectively. The method used is clearly outlined in algorithm 1 for a general $L \times N$ matrix $\mathbf{M}$.

A Simple Algorithm To Achieve Scalability. The incremental SVD is useful but comes with a few issues: (i) the quality of the incremental SVD algorithm degrades with the number of data points (relative to doing the standard batch SVD); (ii) more fundamentally, if the total number of observations in the time series grow very large, then the computational complexity of the algorithm remains prohibitive.

We address this by proposing a simple incremental "meta"-algorithm, which has three hyper-parameters: $T_0, T' \in \mathbb{Z}$ and $\gamma \in (0, 1]$. T_0 is an algorithm-specific parameter which indicates the minimum number of observations required by the algorithm to train its model. The time series observations will be segmented into several intervals of length T and each interval, indexed by $i \in \mathbb{Z}$, will be used to train a sub-model denoted as M_i. As a result, a "complete" model can consist of several sub-models. Adjacent sub-models, M_i, M_{i+1}, will overlap where each M_{i+1} starts with the last $T/2$

Algorithm 1 Incremental SVD algorithm ([59])

Input $\mathbf{M} \in \mathbb{R}^{L\times N}, \mathbf{M} \approx \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^T$, $\mathbf{D} \in \mathbb{R}^{L\times p}$
Output $\mathbf{U}_k^* \in \mathbb{R}^{L\times k}, \mathbf{\Sigma}_k^* \in \mathbb{R}^{k\times k}, \mathbf{V}_k^{*T} \in \mathbb{R}^{k\times(N+p)}$

1: $\hat{\mathbf{D}} \leftarrow (\mathbf{I}_n - \mathbf{U}_k\mathbf{U}_k^T)\mathbf{D}$, where $\mathbf{I}_n$ is the identity matrix
2: $\hat{\mathbf{D}} = \mathbf{Q_D}\mathbf{R_D}$ (QR decomposition of $\hat{\mathbf{D}}$)
3: $\tilde{\mathbf{M}} = \begin{bmatrix} \mathbf{\Sigma}_k & \mathbf{U}_k^T\mathbf{D} \\ 0 & \mathbf{R_D} \end{bmatrix}$
4: $\tilde{\mathbf{M}} \approx \begin{bmatrix} \tilde{\mathbf{U}}_k & \tilde{\mathbf{U}}_p \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{\Sigma}}_k & 0 \\ o & \tilde{\mathbf{\Sigma}}_p \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{V}}_k & \tilde{\mathbf{V}}_p \end{bmatrix}^T$. (SVD of matrix $\tilde{\mathbf{M}}$)
5: $\begin{bmatrix} \mathbf{M} & \mathbf{D} \end{bmatrix} \approx \begin{bmatrix} \mathbf{U}_k & \mathbf{Q_D} \end{bmatrix} \tilde{\mathbf{U}}_k \quad \tilde{\mathbf{\Sigma}}_k \quad (\begin{bmatrix} \mathbf{V}_k & 0 \\ 0 & \mathbf{I}_p \end{bmatrix} \tilde{\mathbf{V}}_k)^T$.
(The Updated SVD)
6: $\mathbf{U}_k^* \leftarrow \begin{bmatrix} \mathbf{U}_k & \mathbf{Q_D} \end{bmatrix} \tilde{\mathbf{U}}_k$
7: $\mathbf{\Sigma}_k^* \leftarrow \tilde{\mathbf{\Sigma}}_k$
8: $\mathbf{V}_k^* \leftarrow (\begin{bmatrix} \mathbf{V}_k & 0 \\ 0 & I_p \end{bmatrix} \tilde{\mathbf{V}}_k)^T$

entries from M_i. Given this, the number of sub-models at time index t for N time series will be $\max(1, \lfloor \frac{2N(t-1)}{T'} \rfloor)$

Below, we detail how the proposed framework operates incrementally. Let $t' \coloneqq N \times t$ denote the number of observations seen thus far. Depending on t', there are three scenarios:

Case 1. $(t' < T_0)$:

1. Given that the minimum observation count has not been met, the model will output the average of the observations $X_n(0:t)$ (for both imputation and forecasting).

Case 2. $(T_0 \leq t' \leq T')$: The first sub-model M_0 is trained incrementally with occasional re-training at times separated per a geometric sequence determined by some parameter $\gamma > 0$:

1. *If* $t' = \lfloor T_0(1+\gamma)^\ell \rfloor$ for $0 \leq \ell \leq q_0$ where $q_0 = \left\lfloor \frac{\ln(T'/T_0)}{\ln(1+\gamma)} \right\rfloor$: Re-train M_0 using $X_1(0:t), \ldots, X_N(0:t)$ (i.e, do full batch SVD).
2. *Else*: incrementally update the model M_0 (i.e., do incremental SVD using the method in Algorithm 1).

Case 3. $(t' > T')$: Here, time is divided into segments of length T' with one last

segment of length less than T'. The segments of length T', if they contain new points, are trained using the original algorithm separately while the last segment of length $\leq T'$ may be trained incrementally with occasional re-training as in Cases 1 and 2. Specifically,

1. Identify M_i where $i = i(t) = \max(0, \lfloor \frac{2t'}{T'} \rfloor - 1)$, and denote the first time index of M_i as $s_i = \frac{iT'}{2}$.
2. *If* $(t' - s_i) = \lfloor T_0(1+\gamma)^\ell \rfloor$ for $0 \leq \ell \leq q$ where $q = \left\lfloor \frac{\ln(2)}{\ln(1+\gamma)} \right\rfloor$: Re-train M_i with observations $X_1(s_i : t), \ldots X_N(s_i : t)$.
3. *Else*: incrementally update M_i.

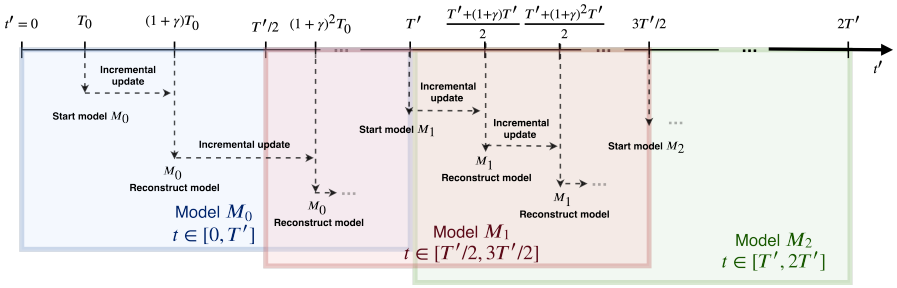


Figure 5-3: Incremental variant of proposed algorithm.

Figure 5-3 illustrates how the proposed segmentation is carried out, and at what points the model is trained fully and where it is incrementally updated.

tspDB Algorithm Hyper-parameters. This incremental variant of mSSA has the following hyper-parameters: L, the number of rows in the stacked page matrix; k, the rank chosen in the mean estimation algorithm; k_{var} the rank chosen in the variance estimation algorithm; T_0, T', and γ, the parameters of the incremental meta-algorithm. In tspDB, we set these parameters in an automated manner, as described and justified in Section 6.1.3.

5.4 Integration With PostgreSQL

In this section, we describe an instantiation of tspDB as an extension on top of the open-source database management system PostgreSQL. As we said earlier, tspDB allows for creating a prediction model on a single or or multiple columns for a mul-

tivariate time series table, which in turn enables predictive queries. In this section, we will describe the overall architecture of the system in terms of its two high-level functionalities: (i) building and updating prediction models (See Section 5.4.1); and (ii) answering predictive queries (See Section 5.4.2).

5.4.1 Building a Prediction Model in tspDB

The process of creating a prediction model is implemented in PostgreSQL as a user-defined function (UDF). This process, which is triggered by the `CREATE PREDICTION_MODEL` command, has three steps: (i) preprocessing the multivariate time series; (ii) training the prediction models; (iii) storing the prediction models. See Figure 5-4.

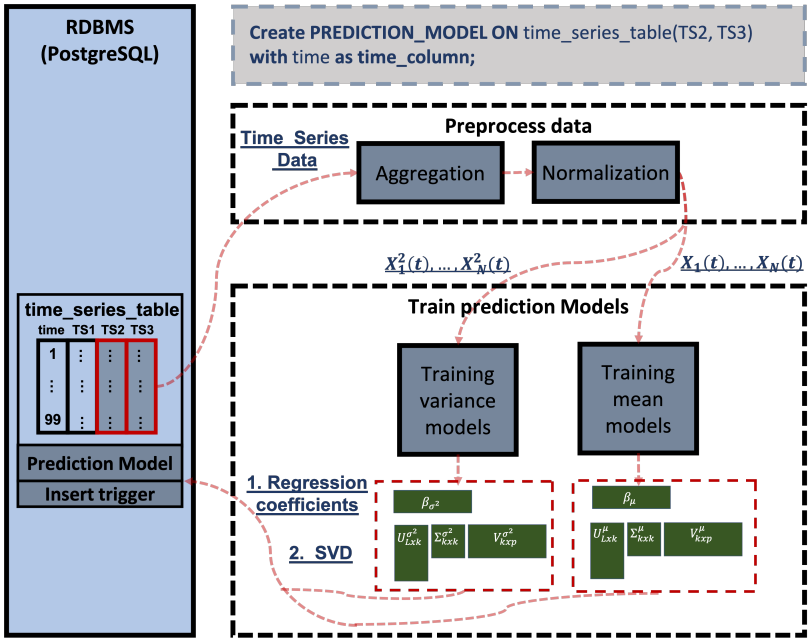


Figure 5-4: The workflow of creating a prediction model in tspDB.

Data Preprocessing. We aggregate each column into an equally-spaced time series. This is done by taking the average of the time series values over a certain time interval. By default, this interval (which can be alternatively selected by the user) is determined by the median of the time difference between consecutive observations in the DB. Then, each time series is normalized to have a zero-mean and unit variance. We do this to limit the negative effect of wide variation in the scales of individual

time series on training accuracy. Finally, the preprocessing module output the aggregated and scaled observations (denoted as $X_1(t), \ldots, X_N(t)$) and their squared values $(X_1^2(t), \ldots, X_N^2(t))$. We note that the aggregation procedure can result in missing values, which is left as such in the input to the mSSA algorithm (step 2).

Training Prediction Models. The main part is this process is training the prediction models for both mean and variance estimation. For this task, we use the scalable mSSA variant described in Section 5.3. As described in Section 5.3, a "full" prediction model will potentially consist of several sub-models, depending on the size of the datasets. While this helps with the scalability of the algorithm, it makes the prediction process much more involved. For example, imputing a missing value would require an initial step of determining to which sub-model(s) does this time step belong to. To mitigate this overhead, an efficent and carefully designed schema is necessary to minimize the predictive query latency. We describe this schema next.

Storage. The prediction models are effectively stored by storing (for both mean and variance models): (i) the appropriate truncated SVD (left/right singular vectors, and singular values); (ii) the regression coefficients. Importantly, we choose to store all of these parameters in the standard relational DB; thus, in response to a query, the DB itself is used to make predictions. The specific schema used to store model parameters, and the relationship between them, is depicted in Figure 5-5.

An important choice/contribution here is identifying the 'Page Matrix' as the de facto data structure representation for a time series, and that this matrix is low-rank for a rich class of generating processes (for both mean and variance). Hence this low-rank structure allows for efficient data management by implying one can simply store the first few singular vectors (an implicit data compression scheme in itself) and yet retain all relevant statistical information. Refer to Figure 5-5 for details.

For a forecasting predictive query, we average the values of linear regression coefficients stored across models. Hence, we create a standard *materialized view* of the coefficient table to precompute the average weights of the last few models, for efficient querying.

Finally, note that an insert trigger is created along with the prediction model to

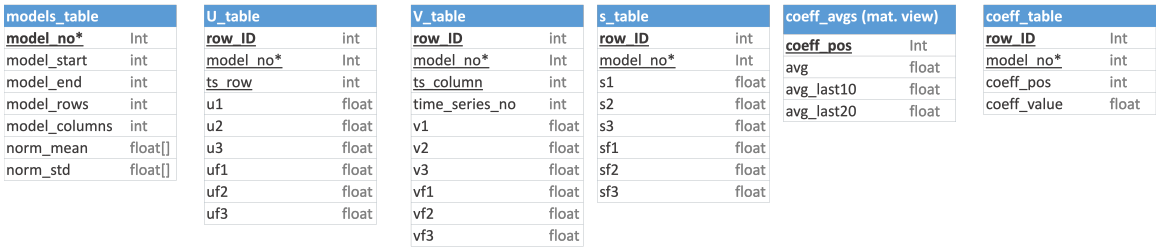


Figure 5-5: The schema used to store the prediction models. Note that bold attributes represent the primary keys, while underlined attributes are columns indexed by a B-tree. The columns u1, v1, s1, ... uk1, vk1, sk1) correspond to $\mathbf{Z}_X$ and the columns uf1, vf1, sf1, ..., ufk2 ,vfk2, sfk2 correspond to $\bar{\mathbf{Z}}_X$. Refer back to Section 3.1 for definition of $\mathbf{Z}^X, \bar{\mathbf{Z}}^X$. Note that we store the variance models in the exact same schema as well.

ensures that it is updated as new data points are inserted.

5.4.2 Prediction Queries' Plan

Our interest is in answering a prediction query PREDICT, as described in Section 5.2. The PREDICT query is one of two types: imputation or forecasting. For both settings, the index needs to provide a response, both of the estimated mean and the associated confidence interval (upper and lower bound). In particular, the atomic response boils down to providing an estimation of the mean of the time series at a given t with confidence interval of $c\%$ for $c \in (0, 100)$. The prediction query creates a response by estimating the mean, say $\hat{f}_n(t)$ and the standard deviation $\hat{\sigma_n}(t)$. Then using Gaussian approximation, the $c\%$ confidence interval is given by, $[\hat{f}_n(t) - \sigma_n(t)\Phi^{-1}(\frac{1}{2} + \frac{c}{200}), \hat{f}_n(t) + \sigma_n(t)\Phi^{-1}(\frac{1}{2} + \frac{c}{200})]$ where $\Phi : \mathbb{R} \to [0, 1]$ denote the Cumulative Density Function of standard Normal distribution with mean 0 and variance 1. One could alternatively utilize Chebyshev's inequality to obtain a more conservative answer as $[\hat{f}_n(t) - \frac{\sigma_n(t)}{\sqrt{1-\frac{c}{100}}}, \hat{f}_n(t) + \frac{\sigma_n(t)}{\sqrt{1-\frac{c}{100}}}]$. Now we describe an algorithm to respond to PREDICT query for given t and time series n with $c\%$ confidence interval. To start with, we determine whether it is an imputation task, i.e. $t \in [T]$ or a forecasting task, i.e. $t > T$.[2] For each of these cases, we respond as follows.

[2]Recall that T is the total number of observations for each time series, While T' is the parameter of our meta-algorithm described in Figure 5-3

Imputation: $\hat{f}_n(t) : t \in [T], n \in [N]$ [3].

1. (Find sub-model) Let $i = i(t) = \max(0, \lfloor \frac{2t \times N}{T'} \rfloor - 1)$. If $i = 0$ and $t < T'/2N$, use $\mathcal{I}(t) = \{0\}$ else $\mathcal{I}(t) = \{i(t), i(t)+1\}$.
2. (Find Row, Column Indices) Let $t_{\text{row}}(j) = \left(t - \frac{jT'}{2N}\right) \mod L$, $t_{\text{col}}(j) = N \left\lfloor \left(t - \frac{jT'}{2}\right)/L \right\rfloor + (n-1)$ be row, column indices of matrix corresponding to the models with $j \in \mathcal{I}(t)$.
3. (Find Truncated SVD for Mean) Query left, right singular vectors U^j, V^j respectively from `U_table`, `V_table` for model corresponding to mean values with index $j \in \mathcal{I}(t)$ along with singular values S^j from `s_table`.
4. (Produce Mean Estimate) Set: $\hat{f}_n(t) = \frac{1}{|\mathcal{I}(t)|} \sum_{j \in \mathcal{I}(t)} \sum_k U^j_{t_{\text{row}}(j)k} V^j_{t_{\text{col}}(j)k} S^j_k$.
5. (Find Truncated SVD for Second Moment) Query left, right singular vectors $\tilde{U}^j, \tilde{V}^j$ respectively from `V_table`, `U_table` for model corresponding to second moment with index $j \in \mathcal{I}(t)$ along with singular values $\tilde{S}^j$ from `s_table`.
6. (Produce Variance Estimate) Set $\hat{\sigma_n}^2(t) = \max(0, \widehat{f_n^2 + \sigma_n^2}(t) - \hat{f}_n(t)^2)$, where: $\widehat{f_n^2 + \sigma_n^2}(t) = \frac{\sum_{j \in \mathcal{I}(t)} \sum_k \tilde{U}^j_{t_{\text{row}}(j)k} \tilde{V}^j_{t_{\text{col}}(j)k} \tilde{S}^j_k}{|\mathcal{I}(t)|}$.
7. (Output Prediction Interval) Output interval using $\hat{f}_n(t), \hat{\sigma_n}(t)$ for queried confidence $c\%$.

Forecasting: $\hat{f}_n(t) : t > T, n \in [N]$.

1. (Retrieve History) Query the last L-1 observations $X_n(T - L + 1 : T))$.
2. For $t_0 \in \{T - L + 1, \ldots, T\}$, set $g^m_n(t_0) = X_n(t_0)$ if $X_n(t_0)$ is not missing, otherwise set it to zero. Similarly, set $g^v_n(t_0) = X^2_n(t_0)$
3. (Obtain Coefficients) Obtain a certain coefficients average from the materialized view (e.g. average of last 10 models) for means $\hat{\beta}^m$ and variances $\hat{\beta}^v$.
4. (Sequential Forecasting) For $\tau \in \{T+1, \ldots, t\}$ produce estimate of means and variances as $g^m_n(\tau) = \sum_{\ell=1}^{L-1} g^m_n(\tau - \ell)\hat{\beta}^m_\ell$, and $g^v_n(\tau) = \sum_{\ell=1}^{L-1} g^v_n(\tau)(\tau - \ell)\hat{\beta}^v_\ell$.
5. (Output Prediction Interval) Output estimate $\hat{f}_n(t) = g^m_n(t)$ and its prediction interval using $\hat{\sigma_n}(t) = g^v_n(t)$ for queried confidence $c\%$.

[3] Note that answering an imputation query for a given range $\{t_1, \ldots, t_2\}$ follows a similar procedure, except that it will potentially query multiple rows from `V_table`, `U_table`, and `s_table`.

5.4.3 Open-source implementation

Finally note that the open source implementation of tspDB is available at tspdb.mit.edu with the following resources:

1. Installation instructions available at tspdb.mit.edu/installation/ for different platforms including Windows, MacOS and Ubuntu.
2. A full demo on standard multivariate time series using the interactive Google Colab environment available at tspdb.mit.edu/demo.
3. A detailed reference for the extension API available at tspdb.mit.edu/API.

In addition, we released a Python implementation of our mSSA variant, including detailed documentation in the GitHub repository github.com/AbdullahO/mSSA.

Chapter 6

Experiments

In this chapter, we detail the extensive testing we conduct to benchmark the performance of tspDB against popular state-of-the-art prediction libraries. In our evaluation, we consider both the statistical and computational performance. Indeed, as stated in [43], given the growing demand to embed ML functionality in high-performance systems, especially in applications with time series data (e.g. financial systems, control systems), it is increasingly vital that we evaluate ML algorithms/systems not just through prediction accuracy, but through computational metrics as well. Therefore, in addition to the statistical performance, we consider two important computational metrics in our evaluations: (i) the time it takes to train a ML model in tspDB; (ii) the time it takes to answer a prediction query, i.e., prediction query latency. Throughout this chapter, along with statistical accuracy, we will benchmark computational performance through these two computational metrics.

In Section 6.1, we detail the experimental setup, e.g., datasets, machine configuration, algorithm hyper-parameters used. In Section 6.2 and 6.3, we extensively benchmark tspDB's statistical and computational performance against other state-of-the-art time series prediction methods. One important benchmark we consider is how much overhead does the added predictive functionalities have on the performance of the original DB management system (e.g., PostgreSQL). To that end, in Section 6.3.2, we benchmark tspDB's computational performance against PostgreSQL with respect to standard DB metrics, specifically the insert throughput and query latency.

Finally, we detail two additional set of experiments we conduct to study the statistical and computational tradeoffs in our implementation of tspDB in Section 6.4.

6.1 Setup

6.1.1 Datasets Description

Throughout the experiments, we use three real-world datasets that are standard benchmarks in time series analysis as well as three synthetic datasets. In this section, we describe these datasets in details.

Electricity Dataset. Obtained from the UCI repository, it records 15-minutes electricity loads of 370 households [53]. We follow the preprocessing done in [58, 47, 46]: data is aggregated into hourly intervals; the first 25968 time-points are used for training; and day-ahead forecasts for the next seven days (i.e. 24-step ahead for 7 windows) are made.

Traffic Dataset. Obtained from the UCI repository, it records occupancy rate of traffic lanes in San Francisco [39]. We follow the preprocessing done in [58, 47, 46]: data is aggregated into hourly intervals; first 10392 time-points are used for training; day-ahead forecasts for the next seven days (i.e. 24-step ahead for 7 windows) are made.

Financial Dataset. Obtained from Wharton Research Data Services (WRDS), it records average daily stocks prices of 839 companies from October 2004 – November 2019 [57]. To limit number of time series for ease of experimentation, we remove stocks with average prices below 30$ across the available period, and those with null values. This preprocessing gives 839 time series (i.e. stock prices) each with 3993 daily readings. We use the first 3813 time points for training. For forecasting, we forecast 180 days ahead one day at a time (i.e. 1-step ahead forecast for 180 windows). We do so as this is a standard goal in finance.

Synthetic Dataset I (Mean Estimation). We generate observations of $n \times m$ time series over T observations $X \in \mathbb{R}^{n \times m \times T}$ by first randomly generating the two

vectors $U \in \mathbb{R}^n = [u_1, \ldots, u_n]$ and $V \in \mathbb{R}^m = [v_1, \ldots, v_m]$. Then, we generate r mixtures of harmonics where each mixture $g_k(t), k \in [r]$, is generated as: $g_k(t) = \sum_{h=1}^{4} \alpha_h \cos(\omega_h t/T)$ where the parameters are selected randomly such that $\alpha_h \in [-1, 10]$ and $\omega_h \in [1, 1000]$. Each observation is constructed as follows: $X_{i,j}(t) = \sum_{k=1}^{r} u_i v_j g_k(t)$, $i \in [n]$, $j \in [m]$. In our experiment, we select $n = 20$, $m = 20$, $T = 15000$, and $r = 4$. This gives us 400 time series each with 15000 observations. In the forecasting experiments, we use the first 14000 points for training. The goal here is to do 10-step ahead forecasts for the final 1000 points.

Synthetic Dataset II (Variance Estimation). With $k \in \{1, 2, 3, 4\}$, we generate three different sets of time series as follows: (i) four harmonics $g_k^{har}(t) = \sum_{i=1}^{4} \alpha_i \cos(\omega_i t/T)$ where $\alpha_i \in [-1, 10]$ and $\omega_i \in [1, 1000]$; (ii) four AR processes $g_k^{AR}(t) = \sum_{i=1}^{3} \phi_i g_k^{AR}(t-i) + \epsilon(t)$ where $\epsilon(t) \sim \mathcal{N}(0, 0.1)$, $\phi_i \in [0.1, 0.4]$; (iii) four trends: $g_k^{trend}(t) = \eta t$ where $\eta \in [10^{-4}, 10^{-3}]$. Then we generate a tensor by sampling two random vectors $U \in \mathbb{R}^{20} = [u_1, \ldots, u_{20}]$ and $V \in \mathbb{R}^{20} = [v_1, \ldots, v_{20}]$. Next we generate three $20 \times 20 \times 1500$ tensors as follow: (i) A mixture of harmonics: $F_{i,j}^1(t) = \sum_{k=1}^{4} u_i v_j g_k^{har}(t)$; (ii) A mixture of harmonics + trend: $F_{i,j}^2(t) = \sum_{k=1}^{4} u_i v_j (g_k^{har}(t) + g_k^{trend}(t))$; (iii) A mixture of harmonics + trend+ AR: $F_{i,j}^3(t) = \sum_{k=1}^{4} u_i v_j (g_k^{trend}(t) + g_k^{har}(t) + g_k^{AR}(t))$. Here $i, j \in [1, \ldots, 20]$. For each tensor, we normalize its values to be between 0 and 1 and use three observations models to generate three tensors from each: (i) Gaussian: $X_{ij}^q(t) \sim \mathcal{N}(F_{i,j}^1(t), F_{i,j}^q(t))$; (ii) Bernoulli: $X_{ij}^q(t) \sim \text{Bernoulli}(F_{i,j}^q(t))$; (iii) Poisson: $X_{ij}^q(t) \sim \text{Pois}(F_{i,j}^q(t))$. Where $q \in \{1, 2, 3\}$. Note, this give us a total of nine observation tensors for our variance experiments. For the forecasting experiments, we use the first 14000 points for training and the last 1000 time steps for testing.

Synthetic Dataset III (Robustness to Observation Models). We generate a f as a sum of harmonics; $f(t) = \sum_{i=1}^{4} \alpha_i \cos(\omega_i t/T)$, where $t \in [T]$, $T = 100000$, $\alpha_i \in [-1.5, 1.5]$ and $\omega_i \in [1, 100]$ (randomly selected). We normalize $f(t)$ to be between 0 and 1, and then generate three different observation models: (i) $X_{Gauss}(t) = f(t) + \epsilon(t)$, where $\epsilon(t) \sim \mathcal{N}(0, 0.5)$ (float); (ii) $X_{Bernoulli}(t) \sim \text{Bernoulli}(f(t))$, i.e. each observation at time t follows a Bernoulli distribution with mean $f(t)$ (boolean); (iii)

$X_{Pois}(t) \sim \text{Pois}(f(t))$, i.e. each observation at time t follows a poisson distribution with mean $\lambda = f(t)$ (integer). The goal is to recover (i.e. impute) and forecast $f(t)$ for each set of observations. Note that we use the first 96000 observations for training, and the last 4000 points for testing.

6.1.2 Machine and DB Configuration

In all experiments, we use an Intel Xeon E5-2683 machine with 16 cores, 132 GB of RAM, and an SSD storage. In Table 6.1, we detail the relevant settings used for PostgreSQL 12.1.

Table 6.1: The used configurations for PostgreSQL 12.1.

Parameter	Value
Shared_buffers	30GB
maintenance_work_mem	2GB
effective_cache_size	80GB
default_transaction_isolation	'read committed'
wal_buffers	16MB
max_parallel_workers	16
max_worker_processes	16
work_mem	54MB

6.1.3 tspDB Hyperparameters

All experimental results are produced using the default settings in the open-source implementation of tspDB. In particular, the hyperparameters of tspDB, which are described in Section 5.3, are selected as follow:

L. We choose L using guidance from the analysis in Chapter 4, which requires $L \leq \bar{P}$. Specifically in tspDB, $L = \bar{P}/10$.

Number of Singular Values (k, k_{var}). This choices is done in a data-driven manner. Specifically, we follow the optimal procedure proposed in [26] for choosing a threshold for HSVT. The procedure depends on the median of the singular values and the shape of the Page matrix. For more details, refer to [26].

"Meta-Algorithm" Parameters (T_0, T', γ). The default parameters we choose are $T_0 = 100,\ T' = 2.5 \times 10^6,\ \gamma = 0.5$. For details on how these parameters are chosen, refer to Section 6.4.2.

6.1.4 Benchmarking Algorithms Parameters and Settings

In this section, we describe the hyper-parameters/implementation used for each method we compare with.

DeepAR. We use the default parameters of "DeepAREstimator" algorithm provided by the GluonTS package. In variance forecasting, we compare with DeepAR as it natively had functionality to produce prediction intervals. It uses a Monte Carlo approach that samples the estimated posterior to produce multiple samples of the time series trajectories. We take the variance of these samples as an estimate of the forecasted variance.

LSTM. Across all datasets, we use a LSTM network with three hidden layers each, with 45 neurons per layer, as is done in [47]. We use the Keras implementation of LSTM [19]. We train the network with a batch size of 128 for 100 epochs.

Prophet. We used Prophet's Python library with the default parameters selected [24].

TRMF. We use the implementation provided by the authors ([58]). We use $k = 60$ for the electricity, $k = 40$ for the traffic, $k = 20$ for the financial and synthetic data and variance experiments (selected via cross-validation); k represents the chosen rank of the $T \times N$ time series matrix. Another hyper-parameter, the lag index is set to include the last day and same weekday in the last week for the traffic and electricity data. For the financial and synthetic data, the lag index include the last 20 points. To perform variance imputation, we adapt TRMF similar to the extension used in tspDB as follows:

1. Impute the time series $X_n(t), X_n^2(t)$ to produce estimates $\hat{f}_n^{\text{TRMF}}(t), \widehat{f_n^2 + \sigma_n^2}^{\text{TRMF}}(t)$ respectively;
2. Output $\hat{\sigma_n^2}^{\text{TRMF}} = \widehat{f_n^2 + \sigma_n^2}^{\text{TRMF}}(t) - \hat{f}_n^{\text{TRMF}}(t)^2$.

6.1.5 Accuracy Score Metrics

In this section, we provide detailed description for the accuracy metrics used throughout the experiments.

Normalized Root Mean Squared Error (NRMSE). Throughout all experiments, NRMSE is used as an accuracy metric. In particular, we rescale each time series to have a zero mean and unit variance before calculating the RMSE. This is done to avoid over-weighting the error in time series with larger magnitudes or greater variance.

Weighted Borda Count (WBC). In Table 6.2, we use WBC to report the algorithms' performance across different experiments. This score is inspired from Borda count – a commonly used method to rank a list of candidates. Specifically, rather than choosing a single candidate, voters rank all candidates in order of preference. Then, each candidate is given a number of points that corresponds to the number of candidates ranked lower.

We use a weighted version of the standard Borda Count, where we rank algorithms (i.e. candidates) based on pairwise comparisons of the normalized root mean squared error (NRMSE) across experiments (i.e. voters). We use this metric as it better captures the relative performance across all experiments and methods, unlike a simple statistic such as the mean or the median. Particularly, let $\mathcal{A}$ represent the set of algorithms used in the experiments. For example, in the mean forecasting experiments, $\mathcal{A} = \{\text{tspDB, DeepAR, LSTM, Prophet, TRMF}\}$. Let $\mathcal{X}$ represent the set of all experiments across different noise levels (σ), and fraction of missing values (ρ). Let $e(a, x)$ represent the NRMSE of algorithm $a \in \mathcal{A}$ in experiment $x \in \mathcal{X}$. Then the Weighted Borda Count of algorithm $a \in \mathcal{A}$ is:

$$\text{WBC}(a) = \frac{1}{|\mathcal{A}| - 1} \sum_{a' \in \mathcal{A}: a' \neq a} \left(\frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \frac{e(a', x)}{e(a, x) + e(a', x)} \right)$$

WBC ranges between 0 and 1 and captures the relative performance against alternative algorithms. A score of 0.5 indicates identical performance to the average of

all other algorithms, a score of 0/1 indicates it performs “infinitely" worse / better, respectively.

6.2 Statistical Accuracy

6.2.1 Mean Estimation Results

Table 6.2: tspDB outperforms state-of-the-art algorithms in mean and variance estimation across different datasets. We use WBC (higher is better) and average NRMSE (lower is better) to summarize the results.

	Mean Imputation (WBC/NRMSE)				Mean Forecasting (WBC/NRMSE)			
	Electricity	Traffic	Synthetic I	Financial	Electricity	Traffic	Synthetic I	Financial
tspDB	**0.61**/**0.39**	**0.50**/**0.49**	**0.53**/**0.25**	**0.65**/**0.28**	**0.53**/ **0.48**	0.50/0.53	**0.70**/**0.20**	**0.66**/**0.36**
LSTM	NA	NA	NA	NA	0.49/0.55	**0.54**/**0.47**	0.45/0.44	0.31/1.20
DeepAR	NA	NA	NA	NA	**0.53**/**0.48**	0.53/**0.47**	0.53/0.33	0.65/0.40
TRMF	0.39/0.69	**0.50**/0.51	0.47/0.33	0.35/0.51	0.50/0.53	0.48/0.57	0.61/0.27	0.58/0.46
Prophet	NA	NA	NA	NA	0.47/0.58	0.45/0.62	0.22/1.01	0.30/1.30

Mean Imputation. We compare the imputation accuracy of tspDB against TRMF, a popular method for imputing (and forecasting) multivariate time series data. We choose it as the single point of comparison as in the work of [33], the authors conduct extensive benchmarking of various time series imputation methods, and find TRMF to have state-of-the-art statistical and computational performance. We note the other time series libraries we compare against, e.g., LSTM, DeepAR, do not have imputation functionality.

We evaluate both tspDB and TRMF using the three real-world datasets and one synthetic dataset (synthetic I) introduced in Section 6.1. To test the algorithms' robustness, we corrupt the various datasets by artificially masking entries with probability $(1 - \rho)$, and by adding zero-mean Gaussian noise to each entry with varying standard deviation σ. As we vary ρ and σ for all four datasets, we find tspDB outperforms TRMF in 80% of experiments. In Table 6.2, we summarize the statistical performance of tspDB against TRMF in terms of WBC and the average NRMSE for each dataset across all experiments. We find that using both metrics, tspDB outperforms TRMF on all datasets. Figure 6-1 gives a visual depiction of the results of the

mean imputation experiments as we vary the fraction of missing data and the noise level.

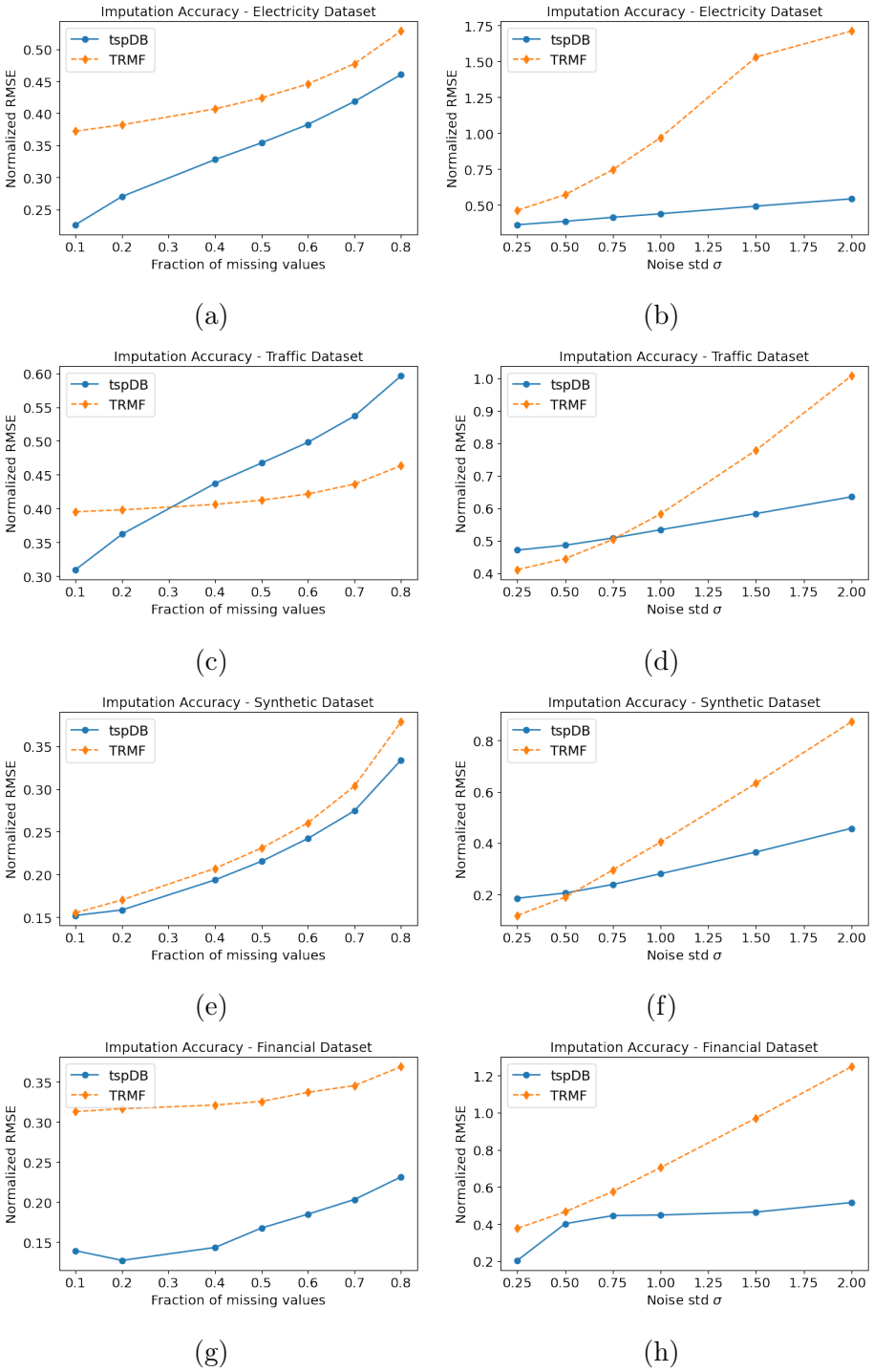


Figure 6-1: tspDB imputation performance surpasses TRMF in 80% of the experiments as we vary the amount of missing data (a,c,e,g), and noise level (b,d,f,h).

Mean Forecasting. We compare the forecasting performance of tspDB against LSTM, DeepAR, TRMF, and Prophet—these are some of the most popular time series libraries used in academia and industry. Using the same datasets used for mean imputation and varying ρ and σ in the same way, we find that tspDB performs competitively with deep-learning algorithms (DeepAR and LSTM), and outperforms both TRMF and Prophet. Specifically, as we vary the fraction of missing values and the noise added, tspDB is the best performing method in 50% of experiments, and at least the second best performing in 80% of experiments. To summarize the statistical performance of all algorithms, we compute their WBC and the average NRMSE for each dataset in Table 6.2; we find tspDB, using both metrics, outperforms all other algorithms in the electricity, financial and synthetic I datasets, while performing competitively with DeepAR and LSTM in the traffic dataset. Figure 6-2 gives a visual depiction of the mean forecasting experiments as we vary the fraction of missing data and the noise level.

6.2.2 Variance Estimation Results

We restrict our analysis to synthetic data as we do not get access to the true underlying time-varying variance in real-world data. We use the dataset synthetic II, which consists of nine sets of multivariate time series each with a different additive combination of time series dynamics and a different noise observation model (Gaussian, Poisson, Bernoulli noise). Refer back to Section 6.1.1 for details on how synthetic II was generated. We again vary the fraction of observed data $\rho \in \{1.0, 0.8, 0.5\}$. For imputation, to the best of our knowledge, there is no algorithm which produces prediction intervals. Hence we use an adaptation of TRMF to benchmark tspDB's performance—note though it does not natively support variance estimation. For forecasting, we compare with DeepAR as it has in-built functionality to produce confidence intervals for its predictions. Refer back to Section 6.1.4 for details of how we use these algorithms to do variance estimation.

We find tspDB has superior performance in all but one experiment ($> 98\%$)

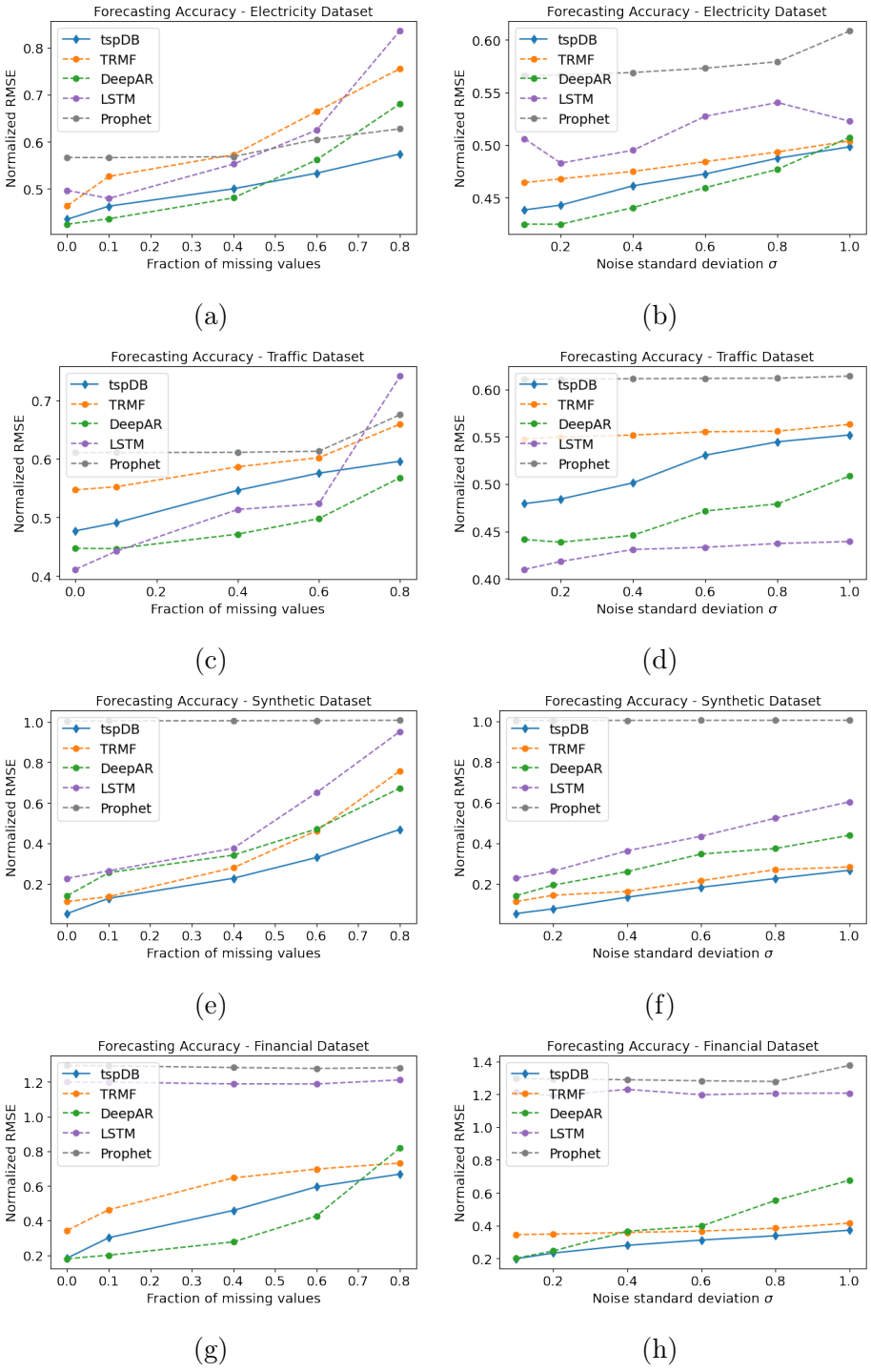


Figure 6-2: tspDB forecasting performance is competitive with/surpasses state-of-the-art algorithms in standard multivariate time series datasets as we vary the amount of missing data (a,c,e,g), and noise level (b,d,f,h).

over both the adapted version of TRMF (for imputation) and DeepAR's in-built

functionality (for forecasting). See Table 6.3 for the NRMSE values for each experiment along with a summary of the performance of each algorithm using WBC and the mean NRMSE across experiments. In summary, we find tspDB outperforms TRMF in variance imputation and DeepAR in variance forecasting across both metrics (WBC, mean NRMSE). Specifically, with respect to imputation, we find tspDB outperforms TRMF in all but one experiment, where the ratio of TRMF's error to tspDB's in the range of 0.97-2.27 (see Table 6.3). With respect to forecasting, we find tspDB outperforms DeepAR in all experiments, where the ratio of DeepAR's error to tspDB's is in the range 1.01-1870.59 (see Table 6.3). tspDB's performance is notably superior when dealing with integer (e.g., Poisson generated) observations. Collectively, these experiments show the robustness of tspDB, in that it is "noise model agnostic" when estimating the mean and variance of a time series.

Table 6.3: tspDB outperforms both TRMF (in variance imputation) and DeepAR (in variance forecasting).

Observation Model	Time Series Dynamics	tspDB (Imputation)			TRMF (Imputation)			tspDB (Forecasting)			DeepAR (Forecasting)		
		$\rho = 1.0$	$\rho = 0.8$	$\rho = 0.5$	$\rho = 1.0$	$\rho = 0.8$	$\rho = 0.5$	$\rho = 1.0$	$\rho = 0.8$	$\rho = 0.5$	$\rho = 1.0$	$\rho = 0.8$	$\rho = 0.5$
Gaussian	Har	**0.076**	**0.099**	**0.118**	0.122	0.125	0.141	**0.106**	**0.144**	**0.156**	0.170	0.184	0.289
	Har + trend	**0.075**	**0.091**	**0.103**	0.133	0.135	0.142	**0.154**	**0.155**	**0.247**	0.286	0.232	0.269
	Har+AR+trend	**0.074**	**0.090**	**0.101**	0.134	0.136	0.146	**0.173**	**0.263**	**0.337**	0.214	0.265	0.388
Poisson	Har	**0.126**	**0.132**	**0.150**	0.137	0.138	0.151	**0.143**	**0.152**	**0.157**	13.20	148.5	90.20
	Har+ trend	**0.086**	**0.087**	**0.101**	0.176	0.187	0.194	**0.093**	**0.182**	**0.199**	0.491	1.403	2.163
	Har+AR+trend	**0.081**	**0.088**	**0.104**	0.184	0.187	0.204	**0.093**	**0.182**	**0.199**	1.386	34.45	162.5
Bernoulli	Har	**0.024**	**0.027**	0.030	0.026	0.027	**0.029**	**0.029**	**0.050**	**0.033**	0.073	0.072	0.077
	Har + trend	**0.022**	**0.025**	**0.025**	0.030	0.030	0.032	**0.029**	**0.048**	**0.059**	0.049	0.068	0.076
	Har+AR+trend	**0.022**	**0.024**	**0.025**	0.030	0.030	0.032	**0.036**	**0.036**	**0.056**	0.070	0.082	0.111
Summary	WBC		**0.597**			0.403			**0.726**			0.264	
	Mean NRMSE		**0.070**			0.112			**0.132**			16.9	

6.2.3 Robustness to Observation Models

In practice, the same latent time series dynamic can lead to very different observations depending on the type of noise that is present (e.g., Gaussian, Poisson, Bernoulli). Thus, as an additional experiment, we showcase tspDB's robustness against different noise models. Pleasingly, we find tspDB is "noise agnostic", i.e., it effectively imputes and forecasts the latent time-varying mean *without knowledge of the underlying noise model*.

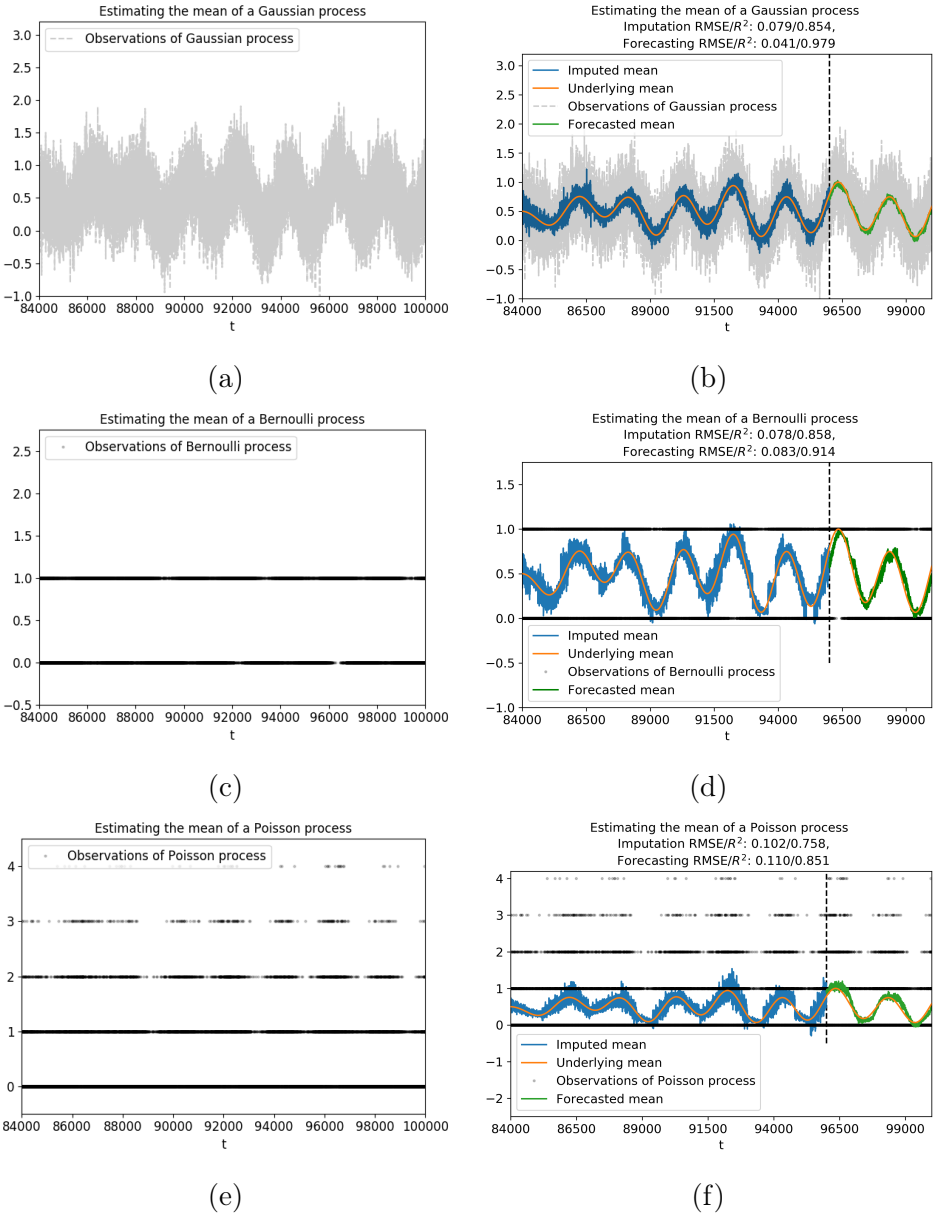


Figure 6-3: Without knowledge of whether the observations come from a time-varying Gaussian process (i.e. floats, see Figure 6-3a), a Bernoulli process (i.e. binary data, see Figure 6-3c), or a Poisson process (i.e. integers, see Figure 6-3e), tspDB successfully imputes and forecasts the underlying mean.

For these set of experiments, we use dataset synthetic III (see Section 6.1.1 for details); we generate three noise models—Gaussian (i.e., float) , Bernoulli (i.e., boolean), and Poisson (i.e., integer)—all with the same latent time series dynamics (normalized to lie within the interval $[0, 1]$) as shown in Figures 6-3b, 6-3d, 6-3f, respectively. We find that tspDB's imputation error in RMSE/R^2 is (0.079/0.854),

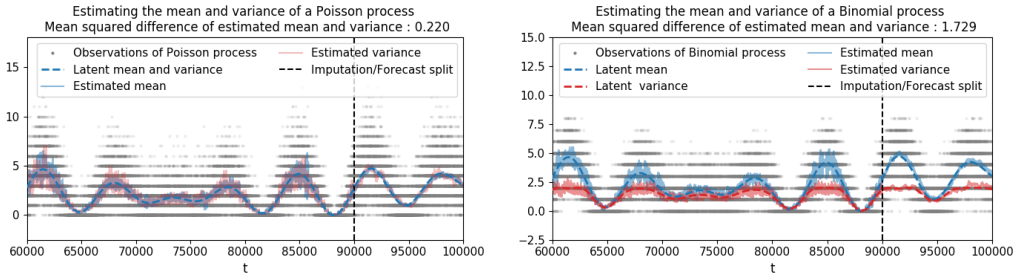
(0.078/0.858) and (0.102/0.758) for the Gaussian, Bernoulli and Poisson observation models respectively. Similarly, the forecasting error in RMSE/R^2 is (0.041/0.979), (0.083/0.914) and (0.110/0.851) for the Gaussian, Bernoulli and Poisson observation models respectively. Figure 6-3 gives a visual depiction of how tspDB effectively imputes and forecasts under Gaussian, Bernoulli and Poisson observation models. In contrast, other methods perform much worse under these other noise models—DeepAR, for example, performs rather poorly with the Bernoulli and Poisson observation models—its forecasting error in RMSE/R^2 is (0.232/0.648), (0.427/0.306), (0.463/$-$1.367) for the Gaussian, Bernoulli and Poisson observation models respectively.

Further, mSSA's (and subsequently tspDB's) ability to accurately estimate the time-varying mean and variance of a time series and its robustness to different observation models lead to more interesting applications. For example, it allows one to verify whether or not a set of integer observations are generated from a Poisson process in a data-driven way as shown in Figure 6-4 – a task that has received significant attention in the literature (see [23, 36, 16, 34]). Specifically, in Figure 6-4, we want to know whether an integer time series follow a Poisson process. One way to check this is to verify whether the latent mean and the latent variances (estimated by tspDB) are equal. In Figure 6-4 we show two examples: one for observations generated from a Poisson process (Figure 6-4a), and one generated from a Binomial process (Figure 6-4b). We further show that tspDB is indeed able to verify which one is likely to be generated from a poisson distribution.

6.3 Computational Performance

6.3.1 Computational Benchmarking of tspDB vs. Other Prediction Algorithms

As stated earlier, we evaluate the computational performance of all algorithms using two metrics: (i) the time it takes to train a prediction model on a given dataset—



(a) Poisson Process. tspDB estimates difference between de-noised mean and variance to be low.

(b) Binomial Process. tspDB estimates difference between de-noised mean and variance to be high.

Figure 6-4: Does the time series follow a Poisson process? With no prior knowledge of the distribution, tspDB can be used to verify if integer observations are generated from a Poisson process (Figure 6-4a) by checking if the latent mean and variances are equal, or if they are different (e.g. Binomial in Figure 6-4b).

each of the methods we benchmark against exposes a ".fit()" interface or equivalent (similar to CREATE PREDICTION_MODEL in tspDB), and we measure the time it takes for a prediction model to be returned from when ".fit()" is executed; (ii) the time required by the model to produce a point forecast for a time series—each of the methods we benchmark against exposes a ".predict()" interface or equivalent (similar to a PREDICT query in tspDB), and we measure the time it takes for a fitted prediction model to return a forecast for a queried future time step (e.g., the time required to do a forward pass in an LSTM network to predict the stock price tomorrow for a certain company). We evaluate all algorithms using the machine configuration detailed in Section 6.1. The hyper-parameters chosen for the various algorithms are detailed in Section 6.1.4; just like for tspDB, we choose the default parameters in the open-source implementations.

Table 6.4: tspDB has significantly less training time, prediction query latency compared to alternatives.

	Training/Insert Time (seconds)				Prediction/Querying Time (milliseconds)			
	Electricity	Traffic	Synthetic I	Financial	Electricity	Traffic	Synthetic I	Financial
tspDB	**11.8**	**14.4**	**6.2**	**2.3**	**3.3**	**3.9**	3.2	**3.6**
LSTM	1357.5	661.2	450.9	97.9	295.6	358.3	278.7	354.8
DeepAR	907.6	572.2	491.2	144.5	366.6	378.5	285.8	207.2
TRMF	34.5	14.8	12.4	3.6	7.3	6.0	**2.5**	4.6
Prophet	6868.8	4726.3	535.9	2165.4	1473.4	1458.8	1480.3	1434.0

In Table 6.4, we report results for these experiments. With respect to time taken to train a model, we find tspDB is 42.4-114.9x faster than LSTM, 39.7-79.2x faster than DeepAR, 1.03-2.92x faster than TRMF, and 86.4-832.7x faster than Prophet. With respect to prediction queries, tspDB's latency is 86.0-99.4x faster than LSTM's, 58.0-110.4x faster than DeepAR's, 0.8-2.2x the latency of TRMF's, and 370.3-456.9x faster than Prophet's.

6.3.2 tspDB vis-a-vis DBMS: comparison with PostgreSQL

On the same datasets we use to benchmark tspDB against these other algorithms, we compare: (i) tspDB's training time (i.e. CREATE PREDICTION_MODEL query) vs. PostgreSQL's bulk insert time; (ii) tspDB's PREDICT query latency against PostgreSQL's SELECT query latency. We find tspDB's model training time ranges from 0.58-1.52x to that of the insert time of PostgreSQL. We evaluate the prediction query latency with and without uncertainty quantification, i.e., producing prediction intervals via variance estimation—see Section 5.4.2 for details on how we do uncertainty quantification. Compared with a SELECT query in PostgreSQL, imputation queries are 1.64-2.67x slower, and forecasting queries are 1.67-2.77x slower. If uncertainty quantification is used, queries are 3.34-5.35x and 3.42-5.48x slower for imputation and forecasting, respectively. This amounts to 1.29-2.36 milliseconds for SELECT queries, 3.22-3.87 milliseconds for imputation queries (5.65-7.88 milliseconds with uncertainty quantification), and 3.24-3.94 milliseconds for forecasting queries (5.67-8.08 milliseconds with uncertainty quantification). Refer to Table 6.5 for a summary of results.

Table 6.5: N and T refer to number of time series and number of observations per time series respectively, and UQ stands for uncertainty quantification.

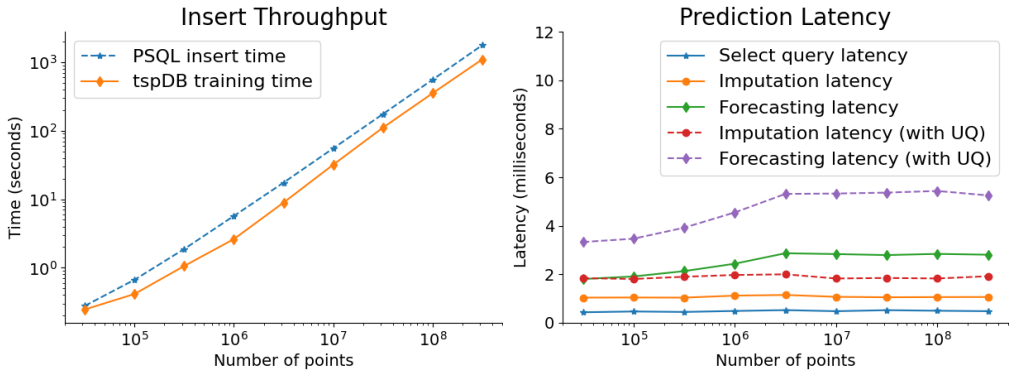
		Training/Insert Time (seconds)		Query Latency (milliseconds)		
	N/T	tspDB	PostgreSQL	tspDB's Forecast /with UQ	tspDB's Imputation /with UQ	PostgreSQL's SELECT
Electricity	370 / 25968	11.81	8.34	3.32/7.02	3.25/6.93	1.34
Traffic	963 / 10392	14.42	9.50	3.94/8.08	3.87/7.88	2.36
Synthetic I	400 / 15000	6.20	4.93	3.24/5.67	3.22/5.65	1.31
Financial	839 / 3993	2.31	4.00	3.57/7.07	3.47/6.90	1.29

Scalability of tspDB. We compare how the computational performance of tspDB relative to PostgreSQL scales with respect to the metrics we consider as we vary the amount of data inserted. We generate a single time series that is a sum of harmonics and then corrupt it with additive Gaussian noise. Specifically, we generate $X(t) = f(t) + \epsilon(t)$, where $\epsilon(t) \sim \mathcal{N}(0, 0.1)$ and $f(t) = \sum_{i=1}^{4} \alpha_i \cos(\omega_i t/T)$, where $t \in [T]$, $\alpha_i \in [-1.5, 1.5]$ and $\omega_i \in [1, 100]$ (randomly selected). We vary the amount of data points (T) between 10^4 to 10^8. Using this time series data, we create a table with the following simple schema: `synthetic(time:timestamp, time_series: float)`, where the column `time` is the primary key, indexed by the default PostgreSQL DB index (B-tree data structure); we note that indexing the time column is a standard practice in time series DBs.

Throughput. As before, we evaluate the time taken to create a prediction model in tspDB as we vary the number of rows inserted from 10^4 to 10^8 and compare it against the time needed to insert the same rows into the aforementioned indexed table. We find that the time required to train tspDB's prediction model is 1.13-2.17x faster than PostgreSQL insert time as we vary the dataset size. In absolute terms, the time to train tspDB's prediction model is 2.62-7.78 microseconds per record. See Figure 6-5a. Since the time required to create the prediction model in tspDB is comparable to PostgreSQL's bulk insert time, and the two operations are independent and so can be effectively run in parallel, integrating tspDB with PostgreSQL will pleasingly not bottleneck PostgreSQL's throughput as new data points are inserted.

Query Latency. As before, we compare the latency of a standard SELECT point query in PostgreSQL against the a PREDICT query in tspDB. We evaluate the latency for both imputation and forecasting predictions with and without uncertainty quantification. As shown in Figure 6-5b, we find that imputation PREDICT queries are 1.77-2.56x slower relative to SELECT queries as we vary the table size, and forecasting PREDICT queries are 3.87-6.39x slower. PREDICT queries with uncertainty quantification are 3.04-4.60x and 6.99-12.80x slower for imputation and forecasting queries respectively. In absolute terms, this amounts to 0.41-0.61 milliseconds for SELECT queries, 0.94-1.23 milliseconds for imputation queries (1.72-2.11 milliseconds

with uncertainty quantification), and 1.73-3.04 milliseconds for forecasting queries (uncertainty quantification).



(a) Training tspDB prediction model is 1.13-2.17x faster than PostgreSQL's bulk insert

(b) Predictions using tspDB are only 1.77x to 6.39x slower than standard SELECT queries

Figure 6-5: Computational performance of tspDB.

Range Prediction Queries In this experiment, we show that tspDB supports efficient PREDICT range queries with latency similar to that of standard SELECT range query. We fix the number of rows in the time series table to 10^7, which is generated as described in the experiment above.

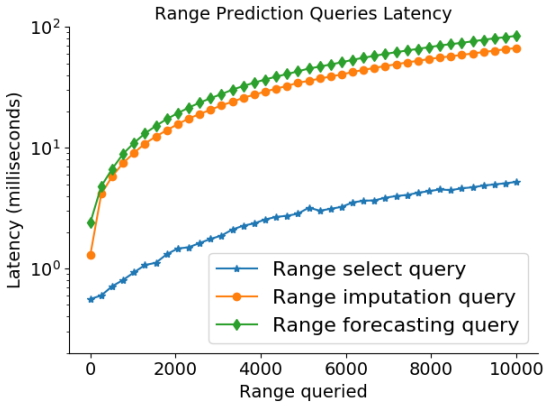


Figure 6-6: Latency of prediction range queries remains comparable to the latency of SELECT queries as the query range increases. Prediction range queries are 2.32x to 16.29x slower.

In Figures 6-6, as we vary the query range between 2 and 10^4 data points, the ratio of the PREDICT range query latency for the two types to that of PostgreSQL's

SELECT range query varies from: 2.32x to 12.86x for imputation; 4.29x to 16.29x for forecasting. Hence, the prediction queries maintain a latency comparable to SELECT as the query range increases.

6.4 Statistical and Computational Tradeoffs in tspDB

We detail two additional set of experiments we conduct to study the statistical and computational tradeoffs in our implementation of tspDB. In Section 6.4.1, we explore this tradeoff between multivariate and univariate time series prediction models – we find using multiple time series can greatly improve accuracy with a moderate slow down in training time and query latency. However, beyond a certain point, the gain in statistical accuracy from adding more time series ($\sim N > 50$) becomes minimal and can add unnecessary overhead. In Section 6.4.2, we study how the three key parameters in the scalable implantation of the prediction algorithm in tspDB, T_0, T', γ (see Section 5.3 for their definition), affect tspDB's statistical and computational performance. We select the default parameters in tspDB based on these experiments.

6.4.1 Multivariate vs. Univariate Prediction Models in tspDB

In our proposed mSSA variant, a prediction model is trained using data from a collection of time series. If this collection of time series are *carefully* selected (i.e., have high "correlation"), this should lead to better statistical accuracy (see Chapter 2 for a more concrete definition of what we mean by "correlated" time series). However, the drawback is that training on multiple time series leads to higher model training time and prediction query latency. We evaluate this tradeoff in tspDB's performance across the three metrics of interest.

Specifically, we use the synthetic I and electricity datasets (detailed in Section 6.1.1). We evaluate tspDB's performance as we vary the number of time series, N, used in training the prediction model. $N \in \{1, 10, 40, 80, 160, 400\}$ and $N \in \{1, 10, 40, 80, 150, 370\}$ for the synthetic and electricity datasets, respectively.

We report performance across the three metrics relative to the univariate case. To reduce variance due to randomness in which time series are selected for the statistical accuracy evaluation, we average results over 50 runs.

Table 6.6: How the three metrics change as we train using more time series (N) relative to the univariate case.

Dataset	Synthetic I						Electricity					
N	1	10	40	80	160	400	1	10	40	80	150	370
Forecasting Accuracy	1.00	0.62	0.11	0.11	0.08	0.06	1.00	0.49	0.44	0.45	0.45	0.46
Training Time	1.00	6.80	16.2	30.28	55.24	250.2	1.00	6.53	20.84	39.63	127.13	369.06
Forecasting Latency	1.00	1.00	1.05	1.15	1.28	1.60	1.00	1.28	1.35	1.60	1.931	2.62

Results. Table 6.6 shows how the three metrics change as we train using more time series. The training time increases almost linearly, as we train on more time series in the synthetic (6.80x-250.2x) and electricity (6.53x-369.06x) datasets. The prediction query latency increases moderately by 1.001x-1.60x and 1.28x-2.62x for the synthetic and electricity datasets respectively. In contrast, the forecasting error decreases by 1.61x-17.59x and 2.04x-2.27x for the synthetic and electricity datasets respectively. In summary, we find that using multiple time series to train the prediction model can greatly improve accuracy with a moderate slow down in the training time and query latency. However, beyond a certain point, the gain in statistical accuracy from adding more time series ($\sim N > 50$) becomes minimal and thus can add unnecessary overhead.

6.4.2 Hyper-parameter Tuning - T_0, T', γ

We discuss how the default parameters for T_0, T', γ are selected in tspDB – we quantify the effect of each of these parameter on three metrics of interest: training time, prediction latency, and statistical accuracy. We find T_0 has minimal effect on these three metrics, and for simplicity of exposition, we simply choose it to be $T_0 = 100$ for all experiments.

Setup. In this experiment, we use a synthetic time series generated from a sum of harmonics with 5×10^7 data points. We initially train the prediction model with all

but the last 10^6 entries and then incrementally add data over a 1000 batches, with a 1000 data points in each batch; note we do so to study the effect of γ on training time, whose effect is limited to such data insertion patterns. We vary the parameter $T' \in [10^4, 10^7]$, and $\gamma \in \{0.01, 0.2, 0.5, 0.7, 1.0\}$.

Results. Figure 6-7 shows tspDB's performance on the three metrics of interest as we vary the parameters. The x-axis reflects the time needed to train the prediction model relative to the time needed to insert the same data into a PostgreSQL table. The y-axis reflects the forecast query latency relative to a PostgreSQL SELECT query. The color of each dot corresponds to the forecasting accuracy in normalized RMSE. We see there is a clear trade-off between query latency and accuracy.

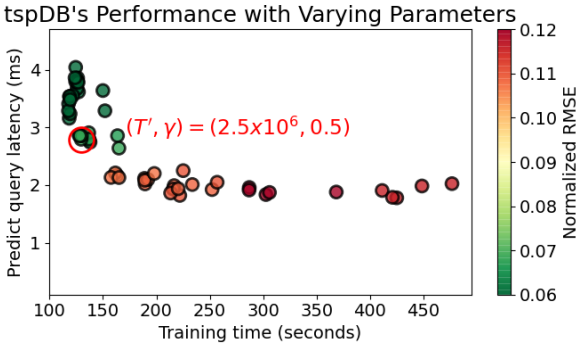


Figure 6-7: T' and γ can significantly affect the three metrics of interest. We choose $T' = 2.5^6$ and $\gamma = 0.5$ which gives the best tradeoff across all three metrics.

Specifically, using small values of $T' \in [10^4, 10^5]$ yields faster but less accurate predictions (3x-4x slower than SELECT queries). Further, it has a negative effect on training time (it is 2x-6x the insert time). On the other hand, using high values of $T' \in [5 \times 10^6, 5 \times 10^7]$ yields more accurate but slower predictions (6x-7x slower than SELECT queries), with relatively low training time (train time is 70% of insert time). We choose the default setting of T' to be 2.5×10^6, gives accurate and relatively fast predictions (5.1x slower than SELECT queries) and relatively low training time (training time is 1.25x quicker compared to PostgreSQL insert time).

We find γ's effect on the accuracy to be minimal but has significant effect on training time for certain range of parameter choice. For example choosing $\gamma = 0.01$ slows training time by $\sim 50\%$ compared to $\gamma = 1.0$. Training time does not vary for

$\gamma \in \{0.5, 0.7, 1.0\}$, with choice of $\gamma = 0.5$ performing best. Hence we choose $\gamma = 0.5$. to be the default setting. Note γ has no effect on predictions latency.

Bibliography

[1] cxoracle. https://oracle.github.io/python-cx_Oracle, 2020. Online; accessed 25 February 2020.

[2] ibmdbr: Ibm in-database analytics for r. https://cran.r-project.org/web/packages/ibmdbR/index.html, 2020. Online; accessed 25 February 2020.

[3] Oracle machine learning for r. https://www.oracle.com/database/technologies/datawarehouse-bigdata/oml4r.html, 2020. Online; accessed 25 February 2020.

[4] Pymysql documenation. https://pymysql.readthedocs.io/en/latest/, 2020. Online; accessed 25 February 2020.

[5] Python support for ibm db2 and ibm informix. https://github.com/ibmdb/python-ibmdb, 2020. Online; accessed 25 February 2020.

[6] Revoscaler package. https://docs.microsoft.com/en-us/machine-learning-server/r-reference/revoscaler/revoscaler, 2020. Online; accessed 25 February 2020.

[7] Anish Agarwal, Abdullah Alomar, and Devavrat Shah. tspdb: Time series predict db. *arXiv e-prints*, pages arXiv–1903, 2019.

[8] Anish Agarwal, Abdullah Alomar, and Devavrat Shah. On multivariate singular spectrum analysis and its variants. *arXiv preprint arXiv:2006.13448*, 2020.

[9] Anish Agarwal, Muhammad Jehangir Amjad, Devavrat Shah, and Dennis Shen. Model agnostic time series analysis via matrix estimation. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 2(3):40, 2018.

[10] Anish Agarwal, Devavrat Shah, Dennis Shen, and Dogyoon Song. On robustness of principal component regression. In *Advances in Neural Information Processing Systems*, pages 9889–9900, 2019.

[11] Mert Akdere, Ugur Cetintemel, Matteo Riondato, Eli Upfal, and Stanley B Zdonik. The case for predictive database systems: Opportunities and challenges. In *CIDR*, pages 167–174, 2011.

[12] Ines Arous, Mourad Khayati, Philippe Cudré-Mauroux, Ying Zhang, Martin Kersten, and Svetlin Stalinlov. Recovdb: accurate and efficient missing blocks recovery for large time series. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 1976–1979. IEEE, 2019.

[13] Sergei Bernstein. *The Theory of Probabilities*. Gastehizdat Publishing House, 1946.

[14] Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327, 1986.

[15] Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327, 1986.

[16] Lawrence Brown, Noah Gans, Avishai Mandelbaum, Anat Sakov, Haipeng Shen, Sergey Zeltyn, and Linda Zhao. Statistical analysis of a telephone call center: A queueing-science perspective. *Journal of the American statistical association*, 100(469):36–50, 2005.

[17] Paul G Brown. Overview of scidb: large scale array storage, processing and analysis. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 963–968, 2010.

[18] José Cambronero, John K Feser, Micah J Smith, and Samuel Madden. Query optimization for dynamic imputation. *Proceedings of the VLDB Endowment*, 10(11):1310–1321, 2017.

[19] François Chollet. keras. https://github.com/fchollet/keras, 2015.

[20] Jeffrey Cohen, Brian Dolan, Mark Dunlap, Joseph M Hellerstein, and Caleb Welton. Mad skills: new analysis practices for big data. *Proceedings of the VLDB Endowment*, 2(2):1481–1492, 2009.

[21] Thomas M Cover. Behavior of sequential predictors of binary sequences. Technical report, DTIC Document, 1966.

[22] Chandler Davis and William Morton Kahan. The rotation of eigenvectors by a perturbation. iii. *SIAM Journal on Numerical Analysis*, 7(1):1–46, 1970.

[23] James Durbin. Some methods of constructing exact tests. *Biometrika*, 48(1-2):41–65, 1961.

[24] Facebook. Prophet. https://facebook.github.io/prophet/. Online; accessed 25 February 2020.

[25] Meir Feder, Neri Merhav, and Michael Gutman. Universal prediction of individual sequences. *Information Theory, IEEE Transactions on*, 38(4):1258–1270, 1992.

[26] Matan Gavish and David L Donoho. The optimal hard threshold for singular values is $4/\sqrt{3}$. *IEEE Transactions on Information Theory*, 60(8):5040–5053, 2014.

[27] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. 1999.

[28] Nina Golyandina, Vladimir Nekrutkin, and Anatoly A Zhigljavsky. *Analysis of time series structure: SSA and related techniques*. Chapman and Hall/CRC, 2001.

[29] Loukas Grafakos. *Classical fourier analysis*, volume 2. Springer, 2008.

[30] Hossein Hassani, Saeed Heravi, and Anatoly Zhigljavsky. Forecasting uk industrial production with multivariate singular spectrum analysis. *Journal of Forecasting*, 32(5):395–408, 2013.

[31] Hossein Hassani and Rahim Mahmoudvand. Multivariate singular spectrum analysis: A general view and new vector forecasting approach. *International Journal of Energy and Statistics*, 1(01):55–83, 2013.

[32] Joe Hellerstein, Christopher Ré, Florian Schoppmann, Daisy Zhe Wang, Eugene Fratkin, Aleksander Gorajek, Kee Siong Ng, Caleb Welton, Xixuan Feng, Kun Li, et al. The madlib analytics library or mad skills, the sql. *arXiv preprint arXiv:1208.4165*, 2012.

[33] Mourad Khayati, Alberto Lerner, Zakhar Tymchenko, and Philippe Cudré-Mauroux. Mind the gap: an experimental evaluation of imputation of missing values techniques in time series. *Proceedings of the VLDB Endowment*, 13(5):768–782, 2020.

[34] Song-Hee Kim and Ward Whitt. Choosing arrival process models for service systems: Tests of a nonhomogeneous poisson process. *Naval Research Logistics (NRL)*, 61(1):66–90, 2014.

[35] Arun Kumar, Jeffrey Naughton, and Jignesh M Patel. Learning generalized linear models over normalized data. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1969–1984, 2015.

[36] Peter AW Lewis. Some results on tests for poisson processes. *Biometrika*, 52(1-2):67–77, 1965.

[37] Chris Mayfield, Jennifer Neville, and Sunil Prabhakar. Eracer: a database approach for statistical inference and data cleaning. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 75–86, 2010.

[38] mldb.ai. Mldb. https://mldb.ai. Online; accessed 25 February 2020.

[39] California Department of Transportation. UCI machine learning repository - pems-sf data set. https://archive.ics.uci.edu/ml/datasets/PEMS-SF, 2011. Online; accessed 25 February 2020.

[40] Vicente Oropeza and Mauricio Sacchi. Simultaneous seismic data denoising and reconstruction via multichannel singular spectrum analysis. *Geophysics*, 76(3):V25–V32, 2011.

[41] Yongjoo Park, Jingyi Qing, Xiaoyang Shen, and Barzan Mozafari. Blinkml: Efficient maximum likelihood estimation with probabilistic guarantees. In *Proceedings of the 2019 International Conference on Management of Data*, pages 1135–1152, 2019.

[42] Nikhil Rao, Hsiang-Fu Yu, Pradeep K Ravikumar, and Inderjit S Dhillon. Collaborative filtering with graph information: Consistency and scalable methods. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2107–2115. Curran Associates, Inc., 2015.

[43] Alexander Ratner, Dan Alistarh, Gustavo Alonso, David G Andersen, Peter Bailis, Sarah Bird, Nicholas Carlini, Bryan Catanzaro, Jennifer Chayes, Eric Chung, et al. Mlsys: The new frontier of machine learning systems. *arXiv preprint arXiv:1904.03257*, 2019.

[44] Jorma Rissanen. Universal coding, information, prediction, and estimation. *Information Theory, IEEE Transactions on*, 30(4):629–636, 1984.

[45] Feras Saad and Vikash K Mansinghka. A probabilistic programming approach to probabilistic data analysis. In *Advances in Neural Information Processing Systems*, pages 2011–2019, 2016.

[46] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 2019.

[47] Rajat Sen, Hsiang-Fu Yu, and Inderjit S Dhillon. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. In *Advances in Neural Information Processing Systems*, pages 4838–4847, 2019.

[48] Devavrat Shah, Sai Burle, Vishal Doshi, Ying-zong Huang, and Balaji Rengarajan. Prediction query language. In *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 611–616. IEEE, 2018.

[49] Zeyuan Shang, Emanuel Zgraggen, Benedetto Buratti, Ferdinand Kossmann, Philipp Eichmann, Yeounoh Chung, Carsten Binnig, Eli Upfal, and Tim Kraska. Democratizing data science through interactive curation of ml pipelines. In *Proceedings of the 2019 International Conference on Management of Data*, pages 1171–1188, 2019.

[50] Paul C Shields. The interactions between ergodic theory and information theory. In *IEEE Transactions on Information Theory*. Citeseer, 1998.

[51] Evan R Sparks, Ameet Talwalkar, Daniel Haas, Michael J Franklin, Michael I Jordan, and Tim Kraska. Automating model search for large scale machine learning. In *Proceedings of the Sixth ACM Symposium on Cloud Computing*, pages 368–380, 2015.

[52] Evan R Sparks, Shivaram Venkataraman, Tomer Kaftan, Michael J Franklin, and Benjamin Recht. Keystoneml: Optimizing pipelines for large-scale advanced analytics. In *2017 IEEE 33rd international conference on data engineering (ICDE)*, pages 535–546. IEEE, 2017.

[53] Artur Trindade. UCI machine learning repository - individual household electric power consumption data set. https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014, 2015. Online; accessed 25 February 2020.

[54] Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

[55] Per-Åke Wedin. Perturbation bounds in connection with singular value decomposition. *BIT Numerical Mathematics*, 12(1):99–111, 1972.

[56] Kevin W Wilson, Bhiksha Raj, and Paris Smaragdis. Regularized non-negative matrix factorization with temporal dependencies for speech denoising. In *Ninth Annual Conference of the International Speech Communication Association*, 2008.

[57] WRDS. The trade and quote (taq) database.

[58] Hsiang-Fu Yu, Nikhil Rao, and Inderjit S Dhillon. Temporal regularized matrix factorization for high-dimensional time series prediction. In *Advances in neural information processing systems*, pages 847–855, 2016.

[59] Hongyuan Zha and Horst D Simon. On updating problems in latent semantic indexing. *SIAM Journal on Scientific Computing*, 21(2):782–791, 1999.

Appendix A

Useful Theorems and Lemmas

A.1 Concentration Inequalities

We recall known concentration inequalities that will be useful throughout.

Theorem A.1.1 (**Bernstein's Inequality [13]**). *Suppose that $X_1, \ldots, X_n$ are independent random variables with zero mean, and M is a constant such that $|X_i| \leq M$ with probability one for each i. Let $S := \sum_{i=1}^{n} X_i$ and $v := Var(S)$. Then for any $t \geq 0$,*

$$\mathbb{P}(|S| \geq t) \leq 2\exp(-\frac{3t^2}{6v + 2Mt}).$$

Theorem A.1.2 (**Spectral Norm of matrices with sub-gaussian entries [54]**). *Let $\boldsymbol{A}$ be an $m \times n$ random matrix whose entries A_{ij} are independent, mean zero, sub-gaussian random variables. Then, for any $t > 0$, we have*

$$\|\boldsymbol{A}\| \leq CK(\sqrt{m} + \sqrt{n} + t)$$

with probability at least $1 - 2\exp(-t^2)$. Here, $K = \max_{i,j} \|A_{ij}\|_{\psi_2}$.

Lemma A.1.1 (**Maximum of sequence of random variables [54]**). *Let X_1, X_2, $\ldots, X_n$ be a sequence of random variables, which are not necessarily independent, and*

satisfy $\mathbb{E}[X_i^{2p}]^{\frac{1}{2p}} \leq Kp^{\frac{\beta}{2}}$ *for some* $K, \beta > 0$ *and all* i*. Then, for every* $n \geq 2$*,*

$$\mathbb{E} \max_{i \leq n} |X_i| \leq CK \log^{\frac{\beta}{2}}(n). \tag{A.1}$$

We note that Lemma A.1.1 implies that if $X_1, \ldots, X_n$ are ψ_α random variables with $\|X_i\|_{\psi_\alpha} \leq K_\alpha$ for all $i \in [n]$, then

$$\mathbb{E} \max_{i \leq n} |X_i| \leq CK_\alpha \log^{\frac{1}{\alpha}}(n).$$

A.2 Matrix Estimation via HSVT

This section describes and analyzes a well-known matrix estimation method, Hard Singular Value Thresholding (HSVT). While the analysis utilizes known arguments from the literature, we need to adapt it for the setting where the underlying 'signal' is only approximately low-rank.

A.2.1 Setup, Notations

Setup. Given a deterministic matrix $\boldsymbol{M} \in \mathbb{R}^{q \times p}$ with $p, q \in \mathbb{N}$ and $q \leq p$, a random matrix $\boldsymbol{Y} \in \mathbb{R}^{q \times p}$ is such that all of its entries, $Y_{ij},\ i \in [q],\ j \in [p]$ are mutually independent and for any given $i \in [q],\ j \in [p]$,

$$Y_{ij} = \begin{cases} M_{ij} + \boldsymbol{\varepsilon}_{ij} & \text{w.p. } \rho, \text{ (i.e. observed)} \\ 0 & \text{w.p. } 1-\rho, \text{ (i.e. not observed)} \end{cases} \tag{A.2}$$

for some $\rho \in (0, 1]$ with $\boldsymbol{\varepsilon}_{ij}$ are independent random variables with $\mathbb{E}[\boldsymbol{\varepsilon}_{ij}] = 0$ and $\|\boldsymbol{\varepsilon}_{ij}\|_{\psi_2} \leq \sigma$. Given this, we have $\mathbb{E}[\boldsymbol{Y}] = \rho \boldsymbol{M}$. Let $\widehat{\rho}$ be the fraction of observed values, defined as

$$\widehat{\rho} = \max\Big(\frac{1}{q \times p}, \frac{\sum_{i=1}^{q}\sum_{j=1}^{p} \mathbb{1}(Y_{ij} \text{ is obs.})}{q \times p}\Big). \tag{A.3}$$

Goal of Matrix Estimation. The goal of matrix estimation is to produce an estimate $\widehat{\boldsymbol{M}}$ from observation $\boldsymbol{Y}$ so that $\widehat{\boldsymbol{M}}$ is close to $\boldsymbol{M}$. In particular, we will be interested in bounding the error between $\widehat{\boldsymbol{M}}$ and $\boldsymbol{M}$ using the following metric: $\|\widehat{\boldsymbol{M}} - \boldsymbol{M}\|_{2,\infty}$.

A.2.2 Matrix Estimation using HSVT

Hard Singular Value Thresholding (HSVT) Map. We define the HSVT map. For any $q, p \in \mathbb{N}$, consider a matrix $\boldsymbol{B} \in \mathbb{R}^{q \times p}$ such that $\boldsymbol{B} = \sum_{i=1}^{q \wedge p} \sigma_i(\boldsymbol{B}) x_i y_i^T$. Here for $i \in [q \wedge p]$, $\sigma_i(\boldsymbol{B})$ is the i-th largest singular value of $\boldsymbol{B}$ and x_i, y_i are the

corresponding left and right singular vectors respectively. Then, for given any $\lambda > 0$, we define the map $\mathrm{HSVT}_\lambda : \mathbb{R}^{q\times p} \to \mathbb{R}^{q\times p}$, which simply shaves off the singular values of the input matrix that are below the threshold λ. Precisely,

$$\mathsf{HSVT}_\lambda(\boldsymbol{B}) = \sum_{i=1}^{q\wedge p} \sigma_i(\boldsymbol{B})\mathbb{1}(\sigma_i(\boldsymbol{B}) \geq \lambda) x_i y_i^T. \tag{A.4}$$

Matrix Estimating using HSVT map. We define a matrix estimation method using the the HSVT map that is utilized by mSSA for imputation. Precisely, we estimate $\boldsymbol{M}$ from $\boldsymbol{Y}$ as follows: given parameter $k \geq 1$,

$$\widehat{\boldsymbol{M}} = \frac{1}{\widehat{\rho}}\mathsf{HSVT}_{\lambda_k}(\boldsymbol{Y}). \tag{A.5}$$

where $\lambda_k = \sigma_k(\boldsymbol{Y})$, i,e. the kth largest singular value of $\boldsymbol{Y}$.

A.2.3 A Useful Linear Operator

We define a linear map associated to HSVT. For a specific choice of $\lambda \geq 0$, define $\varphi_\lambda^{\boldsymbol{B}} : \mathbb{R}^p \to \mathbb{R}^p$ as follows: for any vector $w \in \mathbb{R}^p$ (i.e. $w \in \mathbb{R}^{p\times 1}$),

$$\varphi_\lambda^{\boldsymbol{B}}(w) = \sum_{i=1}^{q\wedge p} \mathbb{1}(\sigma_i(\boldsymbol{B}) \geq \lambda) y_i y_i^T w. \tag{A.6}$$

Note that $\varphi_\lambda^{\boldsymbol{B}}$ is a linear operator and it depends on the tuple $(\boldsymbol{B}, \lambda)$; more precisely, the singular values and the right singular vectors of $\boldsymbol{B}$, as well as the threshold λ. If $\lambda = 0$, then we will adopt the shorthand notation: $\varphi^{\boldsymbol{B}} = \varphi_0^{\boldsymbol{B}}$. The following is a simple, but curious relationship between $\varphi_\lambda^{\boldsymbol{B}}$ and HSVT_λ that will be useful subsequently.

Lemma A.2.1 (Lemma 35 of [10]). *Let $\boldsymbol{B} \in \mathbb{R}^{q\times p}$ and $\lambda \geq 0$ be given. Then for any $j \in [q]$,*

$$\varphi_\lambda^{\boldsymbol{B}}\left(\boldsymbol{B}_{j\cdot}^T\right) = \mathrm{HSVT}_\lambda\left(\boldsymbol{B}\right)_{j\cdot}^T, \tag{A.7}$$

where $\boldsymbol{B}_{j\cdot} \in \mathbb{R}^{1\times p}$ *represents the* j*th row of* $\boldsymbol{B}$*, and* $\mathrm{HSVT}_\lambda\big(\boldsymbol{B}\big)_{j\cdot} \in \mathbb{R}^{1\times p}$ *represents the* j*th row of the matrix obtained after applying HSVT over* $\boldsymbol{B}$ *with threshold* λ*.*

Proof. By (A.6), the orthonormality of the right singular vectors and noting $\boldsymbol{B}_{j\cdot}^T = \boldsymbol{B}^T e_j$ with $e_j \in \mathbb{R}^p$ with jth entry 1 and everything else 0, we have

$$\begin{aligned}
\varphi_\lambda^{\boldsymbol{B}}\big(\boldsymbol{B}_{j\cdot}^T\big) &= \sum_{i=1}^{q\wedge p} \mathbb{1}(\sigma_i(\boldsymbol{B}) \geq \lambda) y_i y_i^T \boldsymbol{B}_{j\cdot}^T = \sum_{i=1}^{q\wedge p} \mathbb{1}(\sigma_i(\boldsymbol{B}) \geq \lambda) y_i y_i^T \boldsymbol{B}^T e_j \\
&= \sum_{i=1}^{q\wedge p} \mathbb{1}(\sigma_i(\boldsymbol{B}) \geq \lambda) y_i y_i^T \big(\sum_{i'=1}^{q\wedge p} \sigma_{i'}(\boldsymbol{B}) x_{i'} y_{i'}^T\big)^T e_j = \sum_{i,i'=1}^{q\wedge p} \sigma_{i'}(\boldsymbol{B}) \mathbb{1}(\sigma_i(\boldsymbol{B}) \geq \lambda) y_i y_i^T y_{i'} x_{i'}^T e_j \\
&= \sum_{i,i'=1}^{q\wedge p} \sigma_{i'}(\boldsymbol{B}) \mathbb{1}(\sigma_i(\boldsymbol{B}) \geq \lambda) y_i \delta_{ii'} x_{i'}^T e_j = \sum_{i=1}^{q\wedge p} \sigma_i(\boldsymbol{B}) \mathbb{1}(\sigma_i(\boldsymbol{B}) \geq \lambda) y_i x_i^T e_j \\
&= HSVT_\lambda\big(\boldsymbol{B}\big)^T e_j = HSVT_\lambda\big(\boldsymbol{B}\big)_{j\cdot}^T.
\end{aligned}$$

□

A.2.4 HSVT based Matrix Estimation: A Deterministic Bound

We state the following result about property of the estimator.

Lemma A.2.2. *For* $k \geq 1$*, let* $\boldsymbol{M} = \boldsymbol{M}_k + \boldsymbol{E}_k$ *with* $rank(\boldsymbol{M}_k) = k$*. Let* $\varepsilon = \max(\widehat{\rho}/\rho, \rho/\widehat{\rho}) \geq 1$*. Then, the HSVT estimate* $\widehat{\boldsymbol{M}}$ *with parameter* k *is such that for all* $j \in [q]$*,*

$$\begin{aligned}
\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T\|_2^2 \leq \frac{2\|\boldsymbol{Y} - \rho\boldsymbol{M}\|_2^2 + 2\rho^2\|\boldsymbol{E}_k\|_2^2}{\big(\sigma_k(\rho\boldsymbol{M}_k)\big)^2}\Big(2\Big\|[\boldsymbol{M}_k]_{j\cdot}^T\Big\|_2^2 + \frac{4\varepsilon^2\big(\|\boldsymbol{Y}_{j\cdot}^T - \rho\boldsymbol{M}_{j\cdot}^T\|_2\big)^2}{\rho^2}\Big) \\
+ \frac{4\varepsilon^2}{\rho^2}\Big\|\varphi^{\boldsymbol{M}_k}(\boldsymbol{Y}_{j\cdot}^T - \rho\boldsymbol{M}_{j\cdot}^T)\Big\|_2^2 + 2(\varepsilon-1)^2\|\boldsymbol{M}_{j\cdot}^T\|_2^2 + 2\Big\|[\boldsymbol{E}_k]_{j\cdot}^T\Big\|_2^2.
\end{aligned} \tag{A.8}$$

Proof. We prove our lemma in four steps.

Step 1. Decomposing $\widehat{\boldsymbol{M}}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T$ in two terms. Fix a row index $j \in [q]$. Let λ_k be the kth largest singular value of $\boldsymbol{Y}$, as used by HSVT algorithm with parameter

$k \geq 1$.

$$\widehat{\boldsymbol{M}}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T = \Big(\widehat{\boldsymbol{M}}_{j\cdot}^T - \varphi_{\lambda_k}^{\boldsymbol{Y}}\big(\boldsymbol{M}_{j\cdot}^T\big)\Big) + \Big(\varphi_{\lambda_k}^{\boldsymbol{Y}}\big(\boldsymbol{M}_{j\cdot}^T\big) - \boldsymbol{M}_{j\cdot}^T\Big). \tag{A.9}$$

By definition per (A.6), $\varphi_{\lambda_k}^{\boldsymbol{Y}} : \mathbb{R}^p \to \mathbb{R}^p$ is the projection operator onto $\text{span}\big\{u_1, \ldots, u_k\big\}$, the span of top k right singular vectors of $\boldsymbol{Y}$, denoted as $u_1, \ldots, u_k$. Therefore,

$$\varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{M}_{j\cdot}^T) - \boldsymbol{M}_{j\cdot}^T \in \text{span}\{u_1, \ldots, u_k\}^{\perp}. \tag{A.10}$$

By design, $\text{rank}(\widehat{\boldsymbol{M}}) = k$. Therefore, by Lemma A.2.1

$$\widehat{\boldsymbol{M}}_{j\cdot} - \varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{M}_{j\cdot}^T) = \frac{1}{\widehat{\rho}}\varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{Y}_{j\cdot}^T) - \varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{M}_{j\cdot}^T) \in \text{span}\{u_1, \ldots, u_k\}. \tag{A.11}$$

Therefore, $\langle \widehat{\boldsymbol{M}}_{j\cdot}^T - \varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{M}_{j\cdot}^T), \varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{M}_{j\cdot}^T) - \boldsymbol{M}_{j\cdot}^T \rangle = 0$, and hence

$$\Big\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T\Big\|_2^2 = \Big\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \varphi_{\lambda_k}^{\boldsymbol{Y}}\big(\boldsymbol{M}_{j\cdot}^T\big)\Big\|_2^2 + \Big\|\varphi_{\lambda_k}^{\boldsymbol{Y}}\big(\boldsymbol{M}_{j\cdot}^T\big) - \boldsymbol{M}_{j\cdot}^T\Big\|_2^2 \tag{A.12}$$

by the Pythagorean theorem.

Step 2. Bounding Term 1, $\Big\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \varphi_{\lambda_k}^{\boldsymbol{Y}}\big(\boldsymbol{M}_{j\cdot}^T\big)\Big\|_2$. We begin by bounding the first term on the right hand side of (A.12). By Lemma A.2.1,

$$\begin{aligned}
\widehat{\boldsymbol{M}}_{j\cdot} - \varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{M}_{j\cdot}^T) &= \frac{1}{\widehat{\rho}}\varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{Y}_{j\cdot}^T) - \varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{M}_{j\cdot}^T) = \varphi_{\lambda_k}^{\boldsymbol{Y}}\Big(\frac{1}{\widehat{\rho}}\boldsymbol{Y}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T\Big) \\
&= \frac{1}{\widehat{\rho}}\varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{Y}_{j\cdot}^T - \rho\boldsymbol{M}_{j\cdot}^T) + \frac{\rho - \widehat{\rho}}{\widehat{\rho}}\varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{M}_{j\cdot}^T).
\end{aligned}$$

Using the Parallelogram Law (or, equivalently, combining Cauchy-Schwartz and AM-GM inequalities), we obtain

$$\begin{aligned}
\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{M}_{j\cdot})^T\|_2^2 &= \|\frac{1}{\widehat{\rho}}\varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{M}_{j\cdot}^T - \rho\boldsymbol{M}_{j\cdot}^T) + \frac{\rho - \widehat{\rho}}{\widehat{\rho}}\varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{M}_{j\cdot}^T)\|_2^2 \\
&\leq 2\,\|\frac{1}{\widehat{\rho}}\varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{Y}_{j\cdot}^T - \rho\boldsymbol{M}_{j\cdot}^T)\|_2^2 + 2\,\|\frac{\rho - \widehat{\rho}}{\widehat{\rho}}\varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{M}_{j\cdot}^T)\|_2^2 \\
&\leq \frac{2}{\widehat{\rho}^2}\|\varphi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{Y}_{j\cdot}^T - \rho\boldsymbol{M}_{j\cdot}^T)\|_2^2 + 2\Big(\frac{\rho - \widehat{\rho}}{\widehat{\rho}}\Big)^2\|\boldsymbol{M}_{j\cdot}^T\|_2^2
\end{aligned}$$

$$\leq \frac{2\varepsilon^2}{\rho^2}\|\varphi^{\boldsymbol{Y}}_{\lambda_k}(\boldsymbol{Y}^T_{j\cdot} - \rho\boldsymbol{M}^T_{j\cdot})\|_2^2 + 2(\varepsilon-1)^2\|\boldsymbol{M}^T_{j\cdot}\|_2^2. \tag{A.13}$$

From definition of ε, $\frac{1}{\widehat{\rho}} \leq \frac{\varepsilon}{\rho}$ and $\left(\frac{\rho-\widehat{\rho}}{\widehat{\rho}}\right)^2 \leq (\varepsilon-1)^2$. The first term of (A.13) can be decomposed as,

$$\|\varphi^{\boldsymbol{Y}}_{\lambda_k}(\boldsymbol{Y}^T_{j\cdot} - \rho\boldsymbol{M}^T_{j\cdot})\|_2^2 \tag{A.14}$$

$$\leq 2\left\|\varphi^{\boldsymbol{Y}}_{\lambda_k}(\boldsymbol{Y}^T_{j\cdot} - \rho\boldsymbol{M}^T_{j\cdot}) - \varphi^{\boldsymbol{M}_k}(\boldsymbol{Y}^T_{j\cdot} - \rho\boldsymbol{M}^T_{j\cdot})\right\|_2^2 + 2\left\|\varphi^{\boldsymbol{M}_k}(\boldsymbol{Y}^T_{j\cdot} - \rho\boldsymbol{M}^T_{j\cdot})\right\|_2^2. \tag{A.15}$$

In above, we have used notation $\varphi^{\boldsymbol{M}_k} = \varphi_0^{\boldsymbol{M}_k}$. Given that $\boldsymbol{M}_k$ is rank k matrix, $\varphi^{\boldsymbol{M}_k}$: $\mathbb{R}^p \to \mathbb{R}^p$ is the projection operator mapping any element in $\mathbb{R}^p$ to the projection onto the subspace spanned by $\{\mu_1, \ldots, \mu_k\}$, where $\mu_1, \ldots, \mu_k \in \mathbb{R}^p$ are the k non-trivial right singular vectors of $\boldsymbol{M}_k$. Similarly, by definition $\varphi^{\boldsymbol{Y}}_{\lambda_k}$ is a map $\mathbb{R}^p \to \mathbb{R}^p$ mapping any element in $\mathbb{R}^p$ to its projection onto the subspace spanned by $\{u_1, \ldots, u_k\}$, the top k right singular vectors of $\boldsymbol{Y}$–this can be seen by noting $\lambda_k = \sigma_k(\boldsymbol{Y})$ is the k-th top singular value of $\boldsymbol{Y}$. Recall $\sigma_j(\boldsymbol{Y})$, $j \in [q \wedge p]$ is the jth largest singular value of $\boldsymbol{Y}$.

Next, we bound the first term on the right hand side of (A.15). To that end, by Wedin $\sin\Theta$ Theorem (see [22, 55]) and recalling $\text{rank}(\boldsymbol{M}_k) = k$,

$$\begin{aligned}
\left\|\varphi^{\boldsymbol{Y}}_{\lambda_k} - \varphi^{\boldsymbol{M}_k}\right\|_2 &\leq \frac{\|\boldsymbol{Y} - \rho\boldsymbol{M}_k\|_2}{\sigma_k(\rho\boldsymbol{M}_k)} \\
&\leq \frac{\|\boldsymbol{Y} - \rho\boldsymbol{M}\|_2}{\sigma_k(\rho\boldsymbol{M}_k)} + \frac{\rho\|\boldsymbol{M} - \boldsymbol{M}_k\|_2}{\sigma_k(\rho\boldsymbol{M}_k)} \\
&\leq \frac{\|\boldsymbol{Y} - \rho\boldsymbol{M}\|_2}{\sigma_k(\rho\boldsymbol{M}_k)} + \frac{\rho\|\boldsymbol{E}_k\|_2}{\sigma_k(\rho\boldsymbol{M}_k)}.
\end{aligned} \tag{A.16}$$

Then it follows that

$$\begin{aligned}
\left\|\varphi^{\boldsymbol{Y}}_{\lambda_k}(\boldsymbol{Y}^T_{j\cdot} - \rho\boldsymbol{M}^T_{j\cdot}) - \varphi^{\boldsymbol{M}_k}(\boldsymbol{Y}^T_{j\cdot} - \rho\boldsymbol{M}^T_{j\cdot})\right\|_2 &\leq \|\varphi^{\boldsymbol{Y}}_{\lambda_k} - \varphi^{\boldsymbol{M}_k}\|_2\|\boldsymbol{Y}^T_{j\cdot} - \rho\boldsymbol{M}^T_{j\cdot}\|_2 \\
&\leq \frac{\big(\|\boldsymbol{Y} - \rho\boldsymbol{M}\|_2 + \rho\|\boldsymbol{E}_k\|_2\big)\big(\|\boldsymbol{Y}^T_{j\cdot} - \rho\boldsymbol{M}^T_{j\cdot}\|_2\big)}{\sigma_k(\rho\boldsymbol{M}_k)}.
\end{aligned} \tag{A.17}$$

Using (A.15) and (A.17) in (A.13),

$$\|\widehat{\boldsymbol{M}}_{j\cdot} - \varphi^{\boldsymbol{Y}}_{\lambda_k}(\boldsymbol{M}^T_{j\cdot})\|_2^2 \le \frac{4\varepsilon^2}{\rho^2}\frac{\big(\|\boldsymbol{Y}-\rho\boldsymbol{M}\|_2+\rho\|\boldsymbol{E}_k\|_2\big)^2\big(\|\boldsymbol{Y}^T_{j\cdot}-\rho\boldsymbol{M}^T_{j\cdot}\|_2\big)^2}{\big(\sigma_k(\rho\boldsymbol{M}_k)\big)^2} + \frac{4\varepsilon^2}{\rho^2}\Big\|\varphi^{\boldsymbol{M}_k}(\boldsymbol{Y}^T_{j\cdot}-\rho\boldsymbol{M}^T_{j\cdot})\Big\|_2^2 + 2(\varepsilon-1)^2\|\boldsymbol{M}^T_{j\cdot}\|_2^2. \quad \text{(A.18)}$$

Step 3. Bounding Term 2, $\Big\|\varphi^{\boldsymbol{Y}}_{\lambda_k}\big(\boldsymbol{M}^T_{j\cdot}\big)-\boldsymbol{M}^T_{j\cdot}\Big\|_2^2$. Recall $\boldsymbol{M}=\boldsymbol{M}_k+\boldsymbol{E}_k$ and using (A.16),

$$\begin{aligned}
\Big\|\varphi^{\boldsymbol{Y}}_{\lambda_k}\big(\boldsymbol{M}^T_{j\cdot}\big)-\boldsymbol{M}^T_{j\cdot}\Big\|_2^2 &= \Big\|\varphi^{\boldsymbol{Y}}_{\lambda_k}\big([\boldsymbol{M}_k]^T_{j\cdot}+[\boldsymbol{E}_k]^T_{j\cdot}\big)-[\boldsymbol{M}_k]^T_{j\cdot}-[\boldsymbol{E}_k]^T_{j\cdot}\Big\|_2^2\\
&\le 2\Big\|\varphi^{\boldsymbol{Y}}_{\lambda_k}\big([\boldsymbol{M}_k]^T_{j\cdot}\big)-[\boldsymbol{M}_k]^T_{j\cdot}\Big\|_2^2+2\Big\|\varphi^{\boldsymbol{Y}}_{\lambda_k}\big([\boldsymbol{E}_k]^T_{j\cdot}\big)-[\boldsymbol{E}_k]^T_{j\cdot}\Big\|_2^2\\
&= 2\Big\|\varphi^{\boldsymbol{Y}}_{\lambda_k}\big([\boldsymbol{M}_k]^T_{j\cdot}\big)-\varphi^{\boldsymbol{M}_k}_{\lambda_k}\big([\boldsymbol{M}_k]^T_{j\cdot}\big)\Big\|_2^2+2\Big\|\varphi^{\boldsymbol{Y}}_{\lambda_k}\big([\boldsymbol{E}_k]^T_{j\cdot}\big)-[\boldsymbol{E}_k]^T_{j\cdot}\Big\|_2^2\\
&\le 2\Big\|\varphi^{\boldsymbol{Y}}_{\lambda_k}-\varphi^{\boldsymbol{M}_k}_{\lambda_k}\Big\|_2^2\Big\|[\boldsymbol{M}_k]^T_{j\cdot}\Big\|_2^2+2\Big\|[\boldsymbol{E}_k]^T_{j\cdot}\Big\|_2^2\\
&\le 2\frac{\big(\|\boldsymbol{Y}-\rho\boldsymbol{M}\|_2+\rho\|\boldsymbol{E}_k\|\big)^2}{\big(\sigma_k(\rho\boldsymbol{M}_k)\big)^2}\Big\|[\boldsymbol{M}_k]^T_{j\cdot}\Big\|_2^2+2\Big\|[\boldsymbol{E}_k]^T_{j\cdot}\Big\|_2^2. \quad \text{(A.19)}
\end{aligned}$$

Step 4. Putting everything together. Inserting (A.18) and (A.19) back to (A.12), we have that for each $j\in[q]$,

$$\begin{aligned}
\Big\|\widehat{\boldsymbol{M}}^T_{j\cdot}-\boldsymbol{M}^T_{j\cdot}\Big\|_2^2 &\le 2\frac{\big(\|\boldsymbol{Y}-\rho\boldsymbol{M}\|_2+\rho\|\boldsymbol{E}_k\|_2\big)^2}{\big(\sigma_k(\rho\boldsymbol{M}_k)\big)^2}\Big\|[\boldsymbol{M}_k]^T_{j\cdot}\Big\|_2^2+2\Big\|[\boldsymbol{E}_k]^T_{j\cdot}\Big\|_2^2\\
&\quad+\frac{4\varepsilon^2}{\rho^2}\frac{\big(\|\boldsymbol{Y}-\rho\boldsymbol{M}\|_2+\rho\|\boldsymbol{E}_k\|_2\big)^2\big(\|\boldsymbol{Y}^T_{j\cdot}-\rho\boldsymbol{M}^T_{j\cdot}\|_2\big)^2}{\big(\sigma_k(\rho\boldsymbol{M}_k)\big)^2}\\
&\quad+\frac{4\varepsilon^2}{\rho^2}\Big\|\varphi^{\boldsymbol{M}_k}(\boldsymbol{Y}^T_{j\cdot}-\rho\boldsymbol{M}^T_{j\cdot})\Big\|_2^2+2(\varepsilon-1)^2\|\boldsymbol{M}^T_{j\cdot}\|_2^2\\
&\le \frac{2\|\boldsymbol{Y}-\rho\boldsymbol{M}\|_2^2+2\rho^2\|\boldsymbol{E}_k\|_2^2}{\big(\sigma_k(\rho\boldsymbol{M}_k)\big)^2}\Big(2\Big\|[\boldsymbol{M}_k]^T_{j\cdot}\Big\|_2^2+\frac{4\varepsilon^2\big(\|\boldsymbol{Y}^T_{j\cdot}-\rho\boldsymbol{M}^T_{j\cdot}\|_2\big)^2}{\rho^2}\Big)\\
&\quad+\frac{4\varepsilon^2}{\rho^2}\Big\|\varphi^{\boldsymbol{M}_k}(\boldsymbol{Y}^T_{j\cdot}-\rho\boldsymbol{M}^T_{j\cdot})\Big\|_2^2+2(\varepsilon-1)^2\|\boldsymbol{M}^T_{j\cdot}\|_2^2+2\Big\|[\boldsymbol{E}_k]^T_{j\cdot}\Big\|_2^2,
\end{aligned} \quad \text{(A.20)}$$

where we used $(a+b)^2\le 2a^2+2b^2$. This completes the proof. □

A.2.5 HSVT based Matrix Estimation: Deterministic To High-Probability

Next, we convert the bound obtained in Lemma A.2.2 to a bound in expectation (as well as one in high-probability) for our metric of interest: $\|\widehat{\boldsymbol{M}}-\boldsymbol{M}\|_{2,\infty}$. In particular, we establish

Theorem A.2.1. *For $k \geq 1$, let $\boldsymbol{M} = \boldsymbol{M}_k + \boldsymbol{E}_k$ with rank$(\boldsymbol{M}_k) = k$. Let $\epsilon = \|\boldsymbol{E}_k\|_\infty$ and $\Gamma = \|\boldsymbol{M}_k\|_\infty$. Let $\rho \geq C\log(qp)/q$ for C large enough and $q \leq p$. Then, the HSVT estimate $\widehat{\boldsymbol{M}}$ with parameter k is such that*

$$\mathbb{E}\big[\max_{j\in[q]} \frac{1}{p}\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T\|_2^2\big] \leq \frac{p(C\sigma^2+\rho^2\epsilon q)}{\rho^2\sigma_k(\boldsymbol{M}_k)^2}\Big(\Gamma^2+\frac{\sigma^2}{\rho^2}\Big) + \frac{C\sigma^2 k\log p}{p\rho^2} + \frac{C(\Gamma+\epsilon)^2}{p} + 2\epsilon^2 + \frac{C}{(pq)^2}. \tag{A.21}$$

Proof. We start by identifying certain high probability events. Subsequently, using these events and Lemma A.2.2, we shall conclude the proof.

High Probability Events. For some positive absolute constant $C > 0$, define

$$E_1 := \Big\{|\widehat{\rho}-\rho| \leq \rho/20\Big\}, \tag{A.22}$$

$$E_2 := \Big\{\|\boldsymbol{Y}-\rho\boldsymbol{M}\|_2 \leq C\sigma\sqrt{p}\Big\}, \tag{A.23}$$

$$E_3 := \Big\{\|\boldsymbol{Y}-\rho\boldsymbol{M}\|_{\infty,2}, \|\boldsymbol{Y}-\rho\boldsymbol{M}\|_{2,\infty} \leq C\sigma\sqrt{p}\Big\}, \tag{A.24}$$

$$E_4 := \Big\{\max_{j\in[q]}\|\varphi^{\boldsymbol{B}}_{\sigma_k(\boldsymbol{B})}\Big(\boldsymbol{Y}_{j\cdot}^T-\rho\boldsymbol{M}_{j\cdot}^T\Big)\|_2^2 \leq C\sigma^2 k\log(p)\Big\}, \tag{A.25}$$

$$E_5 := \left\{\left(1-\sqrt{\frac{20\log(qp)}{\rho qp}}\right)\rho \leq \widehat{\rho} \leq \frac{1}{1-\sqrt{\frac{20\log(qp)}{\rho qp}}}\rho\right\}. \tag{A.26}$$

In (A.25) above, $\boldsymbol{B} \in \mathbb{R}^{q\times p}$ is a deterministic matrix. Let the singular value decomposition of $\boldsymbol{B}$ be given as $\boldsymbol{B} = \sum_{i=1}^q \sigma_i(\boldsymbol{B})x_i y_i^T$, where $\sigma_i(\boldsymbol{B})$ are the singular vectors of $\boldsymbol{B}$ in decreasing order and x_i, y_i are the left and right singular vectors respectively. Recall the definition of $\varphi_\lambda^{\boldsymbol{B}}$ in (A.6). In particular, we choose $\lambda = \sigma_k(\boldsymbol{B})$, the kth singular value of $\boldsymbol{B}$ in (A.25). As a result, in effect, we are bounding norm of projection

of random vector $\boldsymbol{Y}_{j\cdot} - \rho \boldsymbol{M}_{j\cdot}$ for any given deterministic subspace of $\mathbb{R}^p$ of dimension k.

Lemma A.2.3. *For some positive constant $c_1 > 0$ and $C > 0$ large enough in definitions of $E_1, \ldots, E_5$,*

$$\mathbb{P}(E_1) \geq 1 - 2e^{-c_1 pq\rho} - (1-\rho)^{pq}, \tag{A.27}$$

$$\mathbb{P}(E_2) \geq 1 - 2e^{-p}, \tag{A.28}$$

$$\mathbb{P}(E_3) \geq 1 - 2e^{-p}, \tag{A.29}$$

$$\mathbb{P}(E_4) \geq 1 - \frac{2}{(qp)^{10}}. \tag{A.30}$$

$$\mathbb{P}(E_5) \geq 1 - \frac{2}{(qp)^{10}}. \tag{A.31}$$

Proof. We bound the probability of events $E_1, \ldots, E_5$ in that order.

Bounding $\boldsymbol{E}_1$. Let

$$\widehat{\rho}_0 = \big(\sum_{i=1}^{q}\sum_{j=1}^{p} \mathbb{1}(Y_{ij} \text{ is obs.})\big)/(q\ p). \tag{A.32}$$

That is, $\widehat{\rho} = \max(\widehat{\rho}_0, 1/(pq))$ and $\mathbb{E}[\widehat{\rho}_0] = \rho$. We define the event $E_6 := \{\widehat{\rho}_0 = \widehat{\rho}\}$. Thus, we have that

$$\begin{aligned}
\mathbb{P}(E_1^c) &= \mathbb{P}(E_1^c \cap E_6) + \mathbb{P}(E_1^c \cap E_6^c) \\
&= \mathbb{P}(|\widehat{\rho}_0 - \rho| \geq \rho/20) + \mathbb{P}(E_1^c \cap E_6^c) \\
&\leq \mathbb{P}(|\widehat{\rho}_0 - \rho| \geq \rho/20) + \mathbb{P}(E_6^c) \\
&= \mathbb{P}(|\widehat{\rho}_0 - \rho| \geq \rho/20) + (1-\rho)^{qp},
\end{aligned}$$

where the final equality follows by the independence of observations assumption and the fact that $\widehat{\rho}_0 \neq \widehat{\rho}$ only if we do not have any observations. By Bernstein's Inequality, we have that

$$\mathbb{P}(|\widehat{\rho}_0 - \rho| \geq \rho/20) \geq 1 - 2e^{-c_1 \rho qp}.$$

Bounding $\boldsymbol{E}_2$. To start with, $\mathbb{E}[\boldsymbol{Y}] = \rho \boldsymbol{M}$. For any $i \in [q], j \in [p]$, the Y_{ij} are independent, 0 with probability $1-\rho$ and with probability ρ equal to $M_{ij} + \boldsymbol{\varepsilon}_{ij}$ with $\|\boldsymbol{\varepsilon}_{ij}\|_{\psi_2} \leq \sigma$. Therefore, it follows that $\|Y_{ij} - \rho M_{ij}\|_{\psi_2} \leq C'\sigma$ for a constant $C' > 0$. Since $q \leq p$, using Theorem A.1.2 it follows that for an appropriately large constant $C > 0$,

$$\mathbb{P}(E_2) \geq 1 - 2e^{-p}.$$

Bounding $\boldsymbol{E}_3$. Recall that we assume $q \leq p$. Observe that for any matrix $A \in \mathbb{R}^{q \times p}$, $\|A\|_{\infty,2}$, $\|A\|_{2,\infty} \leq \|A\|_2$. Thus using the argument to bound $\boldsymbol{E}_2$, we have (A.29).

Bounding $\boldsymbol{E}_4$. Consider for $j \in [q]$,

$$\|\varphi^{\boldsymbol{B}}_{\sigma_k(\boldsymbol{B})}\Big(\boldsymbol{Y}^T_{j\cdot} - \rho \boldsymbol{M}^T_{j\cdot}\Big)\|_2^2 = \sum_{i=1}^k \|y_i y_i^T (\boldsymbol{Y}^T_{j\cdot} - \rho \boldsymbol{M}^T_{j\cdot})\|_2^2 \tag{A.33}$$

$$\leq \sum_{i=1}^k \Big(y_i^T (\boldsymbol{Y}^T_{j\cdot} - \rho \boldsymbol{M}^T_{j\cdot})\Big)_2^2 \;=\; \sum_{i=1}^k Z_i^2, \tag{A.34}$$

where $Z_i = y_i^T(\boldsymbol{Y}^T_{j\cdot} - \rho \boldsymbol{M}^T_{j\cdot})$. By definition of the ψ_2 norm of a random variable and since y_i is unit norm vector that is deterministic (and hence independent the of random vector $\boldsymbol{Y}^T_{j\cdot} - p\boldsymbol{M}^T_{j\cdot}$), it follows that

$$\|Z_i\|_{\psi_2} = \|y_i^T(\boldsymbol{Y}_{j\cdot} - p\boldsymbol{M}_{j\cdot})\|_{\psi_2} \leq \|(\boldsymbol{Y}_{j\cdot} - p\boldsymbol{M}_{j\cdot})\|_{\psi_2}.$$

Since the coordinates of $\boldsymbol{Y}^T_{j\cdot} - \rho \boldsymbol{M}^T_{j\cdot}$ are mean-zero and independent, with ψ_2 norm bounded by $\sqrt{C}\sigma$ for some absolute constant $C > 0$, using Lemma H.10 of [10], it follows that

$$\mathbb{P}\Big(\sum_{i=1}^k Z_i^2 > t\Big) \leq 2k \exp\Big(-\frac{t}{kC\sigma^2}\Big). \tag{A.35}$$

Therefore, for choice of $t = C\sigma^2 k \log p$ with large enough constant $C > 0$, $q \leq p$, and

taking a union bound over all $j \in [p]$, we have that

$$\mathbb{P}\Big(E_4^c\Big) \leq \frac{2}{(qp)^{10}}. \tag{A.36}$$

Bounding $\boldsymbol{E}_5$. Recall the definition of $\widehat{\rho}$. By the binomial Chernoff bound, for $\varepsilon > 1$,

$$\mathbb{P}\Big(\widehat{\rho} > \varepsilon\rho\Big) \leq \exp\left(-\frac{(\varepsilon-1)^2}{\varepsilon+1} qp\rho\right), \quad \text{and}$$
$$\mathbb{P}\Big(\widehat{\rho} < \frac{1}{\varepsilon}\rho\Big) \leq \exp\left(-\frac{(\varepsilon-1)^2}{2\varepsilon^2} qp\rho\right).$$

By the union bound,

$$\mathbb{P}\Big(\frac{1}{\varepsilon}\rho \leq \widehat{\rho} \leq \rho\varepsilon\Big) \geq 1 - \mathbb{P}\Big(\widehat{\rho} > \varepsilon\rho\Big) - \mathbb{P}\Big(\widehat{\rho} < \frac{1}{\varepsilon}\rho\Big).$$

Noticing $\varepsilon + 1 < 2\varepsilon < 2\varepsilon^2$ for all $\varepsilon > 1$, and substituting $\varepsilon = \left(1 - \sqrt{\frac{20\log(qp)}{qp\rho}}\right)^{-1}$ completes the proof. □

The following are immediate corollaries of the above stated bounds.

Corollary A.2.1. *Let $E := E_1 \cap E_2$. Then, for $\rho \geq C\log(qp)/q$,*

$$\mathbb{P}(E^c) \leq C_1 e^{-c_2 p}, \tag{A.37}$$

where C_1 and c_2 are positive constants.

Corollary A.2.2. *Let $E := E_2 \cap E_3 \cap E_4 \cap E_5$. Then,*

$$\mathbb{P}(E^c) \leq \frac{C_1}{(qp)^{10}}, \tag{A.38}$$

where C_1 is an absolute positive constant.

Probabilistic Bound for HSVT based Matrix Estimation. Recall $\epsilon = \|\boldsymbol{E}_k\|_\infty$. Then $\|\boldsymbol{E}_k\|_F^2 \leq \epsilon qp$. And $\|\boldsymbol{E}_k\|_2^2 \leq \|\boldsymbol{E}_k\|_F^2 \leq \epsilon qp$. Let $\rho \geq C\log(qp)/q$ for C large enough and recall $q \leq p$. Further, recall $\Gamma = \|\boldsymbol{M}_k\|_\infty$; thus, $\|\boldsymbol{M}\|_\infty \leq \Gamma + \epsilon$. Then

$\|[\boldsymbol{M}_k]_{j\cdot}^T\|_2 \le \Gamma\sqrt{p}$ and $\|[\boldsymbol{M}]_{j\cdot}^T\|_2 \le (\Gamma+\epsilon)\sqrt{p}$.

Define $E = E_1 \cap E_2 \cap E_3 \cap E_4 \cap E_5$. Then, from Corollaries A.2.1 and A.2.2, we have that $\mathbb{P}(E^c) \le \frac{C_1}{(qp)^{10}}$ for large enough constant $C_1 > 0$.

Under E_5, we have $\varepsilon = \max(\widehat{\rho}/\rho, \rho/\widehat{\rho}) \le \left(1 - \sqrt{\frac{20\log(qp)}{qp\rho}}\right)^{-1}$. Under this choice of ε and using $\rho \ge C\log(qp)/q$, we have that for C large enough, $\varepsilon \le C$ and $(\varepsilon - 1)^2 \le C/p$.

Given this setup, under event E, Lemma A.2.2 leads to the following: for all $j \in [q]$ and with appropriately (re-defined) large enough constant $C > 0$,

$$\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T\|_2^2 \le C\frac{\sigma^2 p + \rho^2\epsilon qp}{\rho^2\sigma_k(\boldsymbol{M}_k)^2}\left(p\Gamma^2 + \frac{\sigma^2 p}{\rho^2}\right) + \frac{C\sigma^2 k\log p}{\rho^2} + C(\Gamma+\epsilon)^2 + 2p\epsilon^2. \tag{A.39}$$

That is, under event E,

$$\max_{j\in[q]} \frac{1}{p}\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T\|_2^2 \le C\frac{p(\sigma^2 + \rho^2\epsilon q)}{\rho^2\sigma_k(\boldsymbol{M}_k)^2}\left(\Gamma^2 + \frac{\sigma^2}{\rho^2}\right) + \frac{C\sigma^2 k\log p}{p\rho^2} + \frac{C(\Gamma+\epsilon)^2}{p} + 2\epsilon^2. \tag{A.40}$$

For any random variable X and event A, such that under event A, $X \le B$ and $\mathbb{P}(A^c) \le \delta$, we have

$$\begin{aligned}\mathbb{E}[X] &= \mathbb{E}[X\mathbb{1}(A)] + \mathbb{E}[X\mathbb{1}(A^c)] \\ &\le \mathbb{E}[X\mathbb{1}(A)] + \mathbb{E}[X^2]^{\frac{1}{2}}\mathbb{P}(A^c)^{\frac{1}{2}} \\ &\le B + \mathbb{E}[X^2]^{\frac{1}{2}}\delta^{\frac{1}{2}}.\end{aligned} \tag{A.41}$$

We shall use this reasoning above to bound $\mathbb{E}\big[\max_{j\in[q]} \frac{1}{p}\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T\|_2^2\big]$: let $X = \max_{j\in[q]} \frac{1}{p}\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T\|_2^2$ and $A = E$; B is given by right hand side of (A.40), $\delta = \frac{C_1}{(qp)^{10}}$; the only missing quantity that remains to be bounded is $\mathbb{E}[X^2]$. We do that next.

To begin with, for any $j \in [q]$,

$$\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T\|_2 \leq \|\widehat{\boldsymbol{M}}_{j\cdot}^T\|_2 + \|\boldsymbol{M}_{j\cdot}^T\|_2 \tag{A.42}$$

by triangle inequality. As stated earlier, $\|[\boldsymbol{M}]_{j\cdot}^T\|_2 \leq (\Gamma + \epsilon)\sqrt{p}$. Next, we bound $\|\widehat{\boldsymbol{M}}_{j\cdot}\|_2^T$. From (A.5), the fact that $\widehat{\rho} \geq 1/(qp)$, and Lemma A.2.1, we have

$$\begin{aligned}
\|\widehat{\boldsymbol{M}}_{j\cdot}^T\|_2 &= \frac{1}{\widehat{\rho}}\|\mathsf{HSVT}_{\lambda_k}(\boldsymbol{Y})_{j\cdot}^T\|_2 \\
&\leq q\ p\|\phi_{\lambda_k}^{\boldsymbol{Y}}(\boldsymbol{Y}_{j\cdot}^T)\|_2 \\
&\leq q\ p\|\phi_{\lambda_k}^{\boldsymbol{Y}}\|_2\|\boldsymbol{Y}_{j\cdot}^T\|_2 \\
&\leq q\ p\|\boldsymbol{Y}_{j\cdot}^T\|_2,
\end{aligned} \tag{A.43}$$

where we used the fact that $\phi_{\lambda_k}^{\boldsymbol{Y}}$ is a projection operator and hence $\|\phi_{\lambda_k}^{\boldsymbol{Y}}\|_2 = 1$. Note that $Y_{ij} = B_{ij} \times (M_{ij} + \boldsymbol{\varepsilon}_{ij})$, where B_{ij} is an independent Bernoulli variable with $\mathbb{P}(B_{ij} = 1) = \rho$ representing whether $M_{ij} + \boldsymbol{\varepsilon}_{ij}$ is observed or not. Therefore, $|Y_{ij}| = |B_{ij}| \times |M_{ij} + \boldsymbol{\varepsilon}_{ij}| \leq (\Gamma + \epsilon) + |\boldsymbol{\varepsilon}_{ij}|$. Therefore, from (A.42) and (A.43),

$$\begin{aligned}
\max_{j\in[q]}\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T\|_2 &\leq (\Gamma + \epsilon)\sqrt{p} + qp\big(\max_{j\in[q]}\|\boldsymbol{Y}_{j\cdot}^T\|_2\big) \\
&\leq (\Gamma + \epsilon)\sqrt{p} + qp \times \sqrt{p}\big(\max_{i\in[p],j\in[q]}|Y_{ij}|\big) \\
&\leq 2qp^{\frac{3}{2}}\big(\Gamma + \epsilon + \max_{i\in[p],j\in[q]}|\boldsymbol{\varepsilon}_{ij}|\big).
\end{aligned} \tag{A.44}$$

Using $(a+b)^2 \leq 2a^2 + 2b^2$ twice, we have $(a+b)^4 \leq 8(a^4+b^4)$. Therefore, from (A.44)

$$\max_{j\in[q]}\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T\|_2^4 \leq 16q^4p^6\big((\Gamma + \epsilon)^4 + \max_{i\in[p],j\in[q]}|\boldsymbol{\varepsilon}_{ij}|^4\big). \tag{A.45}$$

Recall $\mathbb{E}[\boldsymbol{\varepsilon}_{ij}] = 0$, $\|\boldsymbol{\varepsilon}_{ij}\|_{\psi_2} \leq \sigma$ and $\boldsymbol{\varepsilon}_{ij}$ are independent across i, j. A property of ψ_2-random variables is that $\big|\eta_{ij}\big|^\theta$ is a $\psi_{2/\theta}$-random variable for $\theta \geq 1$. With choice of $\theta = 4$, we have

$$\mathbb{E}\big[\max_{ij}|\boldsymbol{\varepsilon}_{ij}|^4\big] \leq C'\sigma^4\log^2(qp), \tag{A.46}$$

for some $C' > 0$ by Lemma A.1.1. From (A.43), (A.45), and (A.46), we have that

$$\Big(\mathbb{E}\big[\max_{j\in[q]} \frac{1}{p^2}\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T\|_2^4\big]\Big)^{\frac{1}{2}} \leq 4q^2p^2\big((\Gamma+\epsilon)^4 + C'\sigma^4\log^2(qp)\big)^{\frac{1}{2}}. \tag{A.47}$$

Finally, using (A.40), (A.41) and (A.47), we conclude

$$\mathbb{E}\big[\max_{j\in[q]} \frac{1}{p}\|\widehat{\boldsymbol{M}}_{j\cdot}^T - \boldsymbol{M}_{j\cdot}^T\|_2^2\big] \leq \frac{p(C\sigma^2+\rho^2\epsilon q)}{\rho^2\sigma_k(\boldsymbol{M}_k)^2}\Big(\Gamma^2 + \frac{\sigma^2}{\rho^2}\Big) + \frac{C\sigma^2 k\log p}{p\rho^2} + \frac{C(\Gamma+\epsilon)^2}{p} + 2\epsilon^2 + \frac{C}{(pq)^2}. \tag{A.48}$$

This completes the proof of Theorem A.2.1. □

Appendix B

Proofs

B.1 Proofs for Chapter 2

B.1.1 Proof of Propositions 2.2.1 and 2.2.2

Below, we present the proof of Proposition 2.2.2. Recall the definition of the Hankel matrix representation (Definition 2.1.1), we define the *stacked Hankel matrix* of N time series over T time steps as follows. Given N latent time series $f_1, \ldots, f_N$, let $\mathbf{H}_f^{(n)}$ denote the Hankel matrix representation of the latent time series $f_n(t)$. Further, consider the stacked Hankel matrix $\mathbf{H}_f$, obtained by a column-wise concatenation of the Page matrices $\mathbf{H}_f^{(1)}, \ldots, \mathbf{H}_f^{(N)}$. Specifically,

$$\mathbf{H}_f = \begin{bmatrix} \mathbf{H}_f^{(1)} & \mathbf{H}_f^{(2)} & \ldots & \mathbf{H}_f^{(N)} \end{bmatrix}. \tag{B.1}$$

We now establish Proposition B.1.1, which establishes that this stacked Hankel matrix is approximately low-rank. This immediately implies Proposition 2.2.2; since the stacked Page matrix is a sub-matrix of $\mathbf{H}_f$.

Proposition B.1.1. *Let Properties 2.2.1 and 2.2.4 hold for N latent time series of interest, $f_1, \ldots, f_N$. Then for any $T \geq 1$, the stacked Hankel Matrix of these N time series have ϵ'-approximate rank $R \times G$ with $\epsilon' = R\Gamma_1\epsilon$.*

Proof. We have N latent time series $f_1, \ldots, f_n$ satisfying Properties 2.2.1 and 2.2.4.

Consider their stacked Hankel matrix over $[T]$, $\mathbf{H}_f \in \mathbb{R}^{L\times N(T-L+1)}$. By definition for $i \in [L]$ and $j = (n-1)\times(T-L+1)+j'$ for $j' \in [T-L+1]$, we have that the entry in the i-th row and j-th columns is

$$\begin{aligned}[\mathbf{H}_f]_{ij} &= f_n(i+j'-1). \\ &= \sum_{r=1}^{R} U_{nr} W_r(i+j'-1). \end{aligned} \tag{B.2}$$

Where the inequality is direct consequence of Property 2.2.1. Further, let $\mathbf{H}_W^{(r)}$ be the Hankel matrix induced by the r-th fundamental time series $W_r(\cdot)$. Due to Property 2.2.4, there exists a low-rank matrix $\mathbf{M}^{(r)} \in \mathbb{R}^{L\times(T-L+1)}$ such that (a) $\text{rank}(\mathbf{M}^{(r)}) \le G$, (b) $\|\mathbf{H}_W^{(r)} - \mathbf{M}^{(r)}\|_\infty \le \epsilon$. That is, for any $i \in [L], j' \in [T-L+1]$, we have that $\mathbf{M}_{ij'}^{(r)} = \sum_{g=1}^{G} a_{ig}^r b_{j'g}^r$ for some $a_{i\cdot}^r, b_{j'\cdot}^r \in \mathbb{R}^G$. Therefore, for any i, j', we have

$$\begin{aligned} W_r(i+j'-1) = \left[\mathbf{H}_W^{(r)}\right]_{ij'} &= \mathbf{M}_{ij'}^{(r)} + \left(\left[\mathbf{H}_W^{(r)}\right]_{ij'} - \mathbf{M}_{ij'}^{(r)}\right) \\ &= \sum_{g=1}^{G} a_{ig}^r b_{j'g}^r + \left[\mathbf{H}_W^{(r)}\right]_{ij'} - \mathbf{M}_{ij'}^{(r)} \end{aligned} \tag{B.3}$$

From (B.2) and (B.3), we conclude that

$$\begin{aligned}[\mathbf{H}_f]_{ij} &= \sum_{r=1}^{R}\sum_{g=1}^{G} U_{nr} a_{ig}^r b_{j'g}^r + \sum_{r=1}^{R} U_{nr}\left(\left[\mathbf{H}_W^{(r)}\right]_{ij'} - \mathbf{M}_{ij'}^{(r)}\right) \\ &= \sum_{(r,g)\in[R]\times[G]} a_{ig}^r \times (U_{nr} b_{j'g}^r) + \sum_{r=1}^{R} U_{nr}\left(\left[\mathbf{H}_W^{(r)}\right]_{ij'} - \mathbf{M}_{ij'}^{(r)}\right). \end{aligned} \tag{B.4}$$

Define matrix $\mathsf{M} \in \mathbb{R}^{L\times N(T-L+1)}$ with its entry for row $i \in [L]$ and column $j = (n-1)\times(T-L+1)+j'$ for $j' \in [T-L+1]$ given by

$$\begin{aligned} \mathsf{M}_{ij} &= \sum_{(r,g)\in[R]\times[G]} a_{ig}^r \times (U_{nr} b_{j'g}^r) \\ &= \sum_{(r,g)\in[R]\times[G]} \alpha_{i(r,g)} \beta_{j(r,g)}, \end{aligned} \tag{B.5}$$

where $\alpha_{i(r,g)} = a_{ig}^r$ and $\beta_{j(r,g)} = U_{nr} b_{j'g}^r$. Further,

$$
\begin{aligned}
|\left[\mathbf{H}_f\right]_{ij} - \mathsf{M}_{ij}| &\leq \sum_{r=1}^{R} |U_{nr}| \left| \left[\mathbf{H}_W^{(r)}\right]_{ij'} - \mathbf{M}_{ij'}^{(r)} \right| \\
&\leq \sum_{r=1}^{R} \Gamma_1 \left\| \mathbf{H}_W^{(r)} - \mathbf{M}^{(r)} \right\|_\infty \;\leq\; R\Gamma_1\epsilon. \qquad \text{(B.6)}
\end{aligned}
$$

That is, the stacked Hankel matrix $\mathbf{H}_f$ of N time series of $[T]$ observations has ϵ'-approximate rank $G \times R$ with $\epsilon' = R\Gamma_1\epsilon$. This completes the proof. □

Note that Proposition 2.2.1 is a special case of Proposition 2.2.2 where $\epsilon = 0$.

B.1.2 Proof of Proposition 2.2.4

Helper Lemmas for Proposition 2.2.4

We begin by stating some classic results from Fourier Analysis. To do so, we introduce some notation. Throughout, we have $R > 0$.

$C[0, R]$ and $L^2[0, R]$ functions. $C[0, R]$ is the set of real-valued, continuous functions defined on $[0, R]$. $L^2[0, R]$ is the set of square integrable functions defined on $[0, R]$, i.e. $\int_0^R f^2(t)dt \leq \infty$

Inner Product of functions in $L^2[0, R]$. $L^2[0, R]$ is a space endowed with inner product defined as $\langle f, g\rangle := \frac{1}{R}\int_0^R f(t)g(t)dt$, and associated norm as $\|f\| := \sqrt{\frac{1}{R}\int_0^R f^2(t)dt}$.

Fourier Representation of functions in $L^2[0, R]$. For $f \in L^2[0, R]$, define its $G \geq 1$-order Fourier representation, $\mathcal{F}(f, G) \in L^2[0, R]$ as

$$
\mathcal{F}(f, G)(t) = a_0 + \sum_{g=1}^{G} (a_g \cos(2\pi gt/R) + b_g \cos(2\pi gt/R)), \quad t \in [0, R], \qquad \text{(B.7)}
$$

where a_0, a_g, b_g with $g \in [G]$ are called the Fourier coefficients of f, defined as

$$
a_0 := \langle f, 1\rangle = \frac{1}{R}\int_0^R f(t)dt,
$$

$$a_g := \langle f, \cos(2\pi gt/R)\rangle = \frac{1}{R}\int_0^R f(t)\cos(2\pi gt/R)dt,$$
$$b_g := \langle f, \sin(2\pi gt/R)\rangle = \frac{1}{R}\int_0^R f(t)\sin(2\pi gt/R)dt.$$

We now state a classic result from Fourier analysis.

Theorem B.1.1 ([29]). *Given $k \geq 1, R > 0$, let $f \in C^k(R, \mathsf{PER})$. Then, for any $t \in [0, R]$ (or more generally $t \in \mathbb{R}$),*

$$\lim_{G\to\infty} \mathcal{F}(f, G)(t) \to f(t). \tag{B.8}$$

We next argue that if $f \in C^k(R, \mathsf{PER})$, then its Fourier coefficients decay rapidly. Precisely,

Lemma B.1.1. *Given $k \geq 1, R > 0$, let $f \in C^k(R, \mathsf{PER})$. Then, for $j \in [k]$, the G-order Fourier coefficient of $f^{(j)}$, the j-th derivative of f, recursively satisfy the following relationship: for $g \in [G]$,*

$$a_g^{(j)} = -\Big(\frac{2\pi g}{R}\Big) b_g^{(j-1)}, \qquad b_g^{(j)} = \Big(\frac{2\pi g}{R}\Big) a_g^{(j-1)}. \tag{B.9}$$

Proof. We establish (B.9) for $a_g^{(1)}$, $g \in [G]$. Notice that an identical argument applies to establish (B.9) for any $a_g^{(j)}, b_g^{(j)}$ for $j \in [k]$ and $g \in [G]$. To that end, consider

$$\begin{aligned} a_g^{(1)} = \langle f^{(1)}, \cos(2\pi gt/R)\rangle &= \frac{1}{R}\int_0^R f^{(1)}(t)\cos(2\pi gt/R)dt \\ &\overset{(a)}{=} \frac{1}{R}\Big(\Big[f(t)\cos(2\pi gt/R)\Big]_0^R - \frac{2\pi g}{R}\Big[\frac{1}{R}\int_0^R f(t)\sin(2\pi gt/R)dt\Big]\Big) \\ &= -\Big(\frac{2\pi g}{R}\Big) b_g^{(0)}. \end{aligned}$$

Where (a) follows by integration by parts. □

Completing Proof of Proposition 2.2.4

Proof. For $G \in \mathbb{N}$, let $\mathcal{F}(f, G)$ be defined as in (B.7). Then for $t \in \mathbb{R}$

$$
\begin{aligned}
|f(t) - \mathcal{F}(f,G)(t)| &\overset{(a)}{=} \Big| \sum_{g=G+1}^{\infty} \left(a_g \cos(2\pi g t/R) + b_g \cos(2\pi g t/R)\right) \Big| \\
&\leq \sum_{g=G+1}^{\infty} |a_g| + |b_g| \\
&\overset{(b)}{\leq} \sum_{g=G+1}^{\infty} \Big(\frac{R}{2\pi g}\Big)^k \Big(|a_g^{(k)}| + |b_g^{(k)}|\Big) \\
&\overset{(c)}{\leq} \sqrt{2}\Big(\frac{R}{2\pi}\Big)^k \sqrt{\sum_{g=G+1}^{\infty} \Big(\frac{1}{g}\Big)^{2k}} \sqrt{\sum_{g=G+1}^{\infty} \Big(|a_g^{(k)}|^2 + |b_g^{(k)}|^2\Big)} \\
&\overset{(d)}{\leq} \sqrt{2}\Big(\frac{R}{2\pi}\Big)^k \frac{1}{G^{k-0.5}} \sqrt{\sum_{g=G+1}^{\infty} \Big(|a_g^{(k)}|^2 + |b_g^{(k)}|^2\Big)} \\
&\overset{(e)}{\leq} \sqrt{2}\Big(\frac{R}{2\pi}\Big)^k \frac{\|f^{(k)}\|}{G^{k-0.5}} \\
&= C(k,R)\frac{\|f^{(k)}\|}{G^{k-0.5}},
\end{aligned}
$$

where $C(k, R)$ is a constant that depends only on k and R; (a) follows from Theorem B.1.1; (b) follows from Lemma B.1.1; (c) follows from Cauchy-Schwarz inequality and fact that $(\alpha+\beta)^2 \leq 2(\alpha^2+\beta^2)$ for any $\alpha, \beta \in \mathbb{R}$; (d) $\sum_{g=G+1}^{\infty} g^{-2k} \leq \int_G^{\infty} x^{-2k} dx$ which can be bounded as $G^{-2k+1}/(2k-1)$ which is at most G^{-2k+1} since $k \geq 1$; (e) follows from the Bessel's inequality, i.e. $\|f^{(k)}\|^2 \geq \sum_{g=0}^{\infty}(|a_g^{(k)}|^2 + |b_g^{(k)}|^2)$.

Thus, for any $t \in \mathbb{R}$, we have a uniform error bound for f being approximated by $\mathcal{F}(f, G)$ which is a sum of $2G$ harmonics. Noting $2G$ harmonics can be represented by an order-$4G$ LRF (by Proposition 2.2.3),we complete the proof. □

B.1.3 Proof of Proposition 2.2.5

Proof. f_1, f_2 have a (G_1, ϵ_1) and (G_2, ϵ_2)-Hankel representation respectively. For any $T \geq 1$, let $\mathbf{H}_1, \mathbf{H}_2 \in \mathbb{R}^{L \times (T-L+1)}$ be the Hankel matrices of f_1, f_2 respectively over time interval $[T]$. By definition, there exists matrices $\boldsymbol{M}_1, \boldsymbol{M}_2 \in \mathbb{R}^{L \times (T-L+1)}$ such

that $\text{rank}(\boldsymbol{M}_1) \leq G_1$, $\|\boldsymbol{M}_1 - \mathbf{H}_1\|_\infty \leq \epsilon_1$ and $\text{rank}(\boldsymbol{M}_2) \leq G_2$, $\|\boldsymbol{M}_2 - \mathbf{H}_2\|_\infty \leq \epsilon_2$.

Component-wise addition. Note the Hankel matrix of $f_1 + f_2$ over $[T]$ is $\mathbf{H}_1 + \mathbf{H}_2$. Then, matrix $\boldsymbol{M} = \boldsymbol{M}_1 + \boldsymbol{M}_2$ has rank at most $G_1 + G_2$ since for any two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, it is the case that $\text{rank}(\boldsymbol{A} + \boldsymbol{B}) \leq \text{rank}(\boldsymbol{A}) + \text{rank}(\boldsymbol{B})$. Further, $\|\mathbf{H}_1 + \mathbf{H}_2 - (\boldsymbol{M}_1 + \boldsymbol{M}_2)\|_\infty \leq \epsilon_1 + \epsilon_2$. Therefore it follows that $f_1 + f_2$ has $(G_1 + G_2, \epsilon_1 + \epsilon_2)$-Hankel representation.

Component-wise multiplication. For $f_1 \circ f_2$, its Hankel over $[T]$ is given by $\mathbf{H}_1 \circ \mathbf{H}_2$ where we abuse notation of $\circ$ in the context of matrices as the Hadamard product of matrices. Let $\boldsymbol{M} = \boldsymbol{M}_1 \circ \boldsymbol{M}_2$. Then $\text{rank}(\boldsymbol{M}) \leq G_1 \times G_2$ since for any two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, $\text{rank}(\boldsymbol{A} \circ \boldsymbol{B}) \leq \text{rank}(\boldsymbol{A})\text{rank}(\boldsymbol{B})$. Now

$$
\begin{aligned}
\|\mathbf{H}_1 \circ \mathbf{H}_2 - \boldsymbol{M}_1 \circ \boldsymbol{M}_2\|_\infty &\leq \|\mathbf{H}_1 \circ \mathbf{H}_2 - \mathbf{H}_1 \circ \boldsymbol{M}_2\|_\infty + \|\mathbf{H}_1 \circ \boldsymbol{M}_2 - \boldsymbol{M}_1 \circ \boldsymbol{M}_2\|_\infty \\
&\leq \|\mathbf{H}_1\|_\infty \|\mathbf{H}_2 - \boldsymbol{M}_2\|_\infty + \|\boldsymbol{M}_2\|_\infty \|\mathbf{H}_1 - \boldsymbol{M}_1\|_\infty \\
&\leq \|f_1\|_\infty \epsilon_2 + (\|\boldsymbol{M}_2 - \mathbf{H}_2\|_\infty + \|\mathbf{H}_2\|_\infty)\epsilon_1 \\
&\leq \|f_1\|_\infty \epsilon_2 + (\|f_2\|_\infty + \epsilon_2)\epsilon_1 \\
&= \|f_1\|_\infty \epsilon_2 + \|f_2\|_\infty \epsilon_1 + \epsilon_1 \epsilon_2 \qquad \text{(B.10)} \\
&\leq \ 3 \max(\epsilon_1, \epsilon_2) \max(\|f_1\|_\infty, \|f_2\|_\infty). \qquad \text{(B.11)}
\end{aligned}
$$

This completes the proof of Proposition 2.2.5. □

B.2 Proofs for Chapter 4

B.2.1 Proof of Theorem 4.1.1

The proof of Theorem 4.1.1 will utilize Theorem A.2.1. To begin with, given N time series with observations over timesteps $[T]$, the mSSA algorithm as described in Section 3.1 constructs the $L \times (NT/L)$ stacked page matrix $\mathbf{Z}_X$ with $L = \sqrt{\min(N, T)T}$, i.e. $L \leq T$.

As per the model described by (2.1) and Section 2.2, it follows that each entry of

$\mathbf{Z}_X$ is an independent random variable; it is observed with probability $\rho \in (0, 1]$ independently and when it is observed, its equal to value of the latent time series plus zero-mean sub-Gaussian noise. In particular,

$$\mathbb{E}[\mathbf{Z}_X] = \rho \mathbf{Z}_f \tag{B.12}$$

where $\mathbf{Z}_f \in \mathbb{R}^{L \times (NT/L)}$ is the stacked page matrix induce by the latent time series, i.e., entry in row $\ell \in [L]$ and column $(n-1) \times T/L + j$ is equal to $f_n(\ell + (j-1) \times L)$. Further, when entry in row $\ell \in [L]$ and column $(n-1) \times T/L + j$ in $\mathbf{Z}_X$ is observed, i.e. $X_n(\ell + (j-1) \times L) \neq \star$, it is equal to $f_n(\ell + (j-1) \times L) + \eta_n(\ell + (j-1) \times L)$ where $\eta_n(\cdot)$ are independent, zero-mean sub-Gaussian variables with $\|\eta_n(\cdot)\|_{\psi_2} \leq \gamma$ as per the Property 2.2.3.

Under Properties 2.2.1 and 2.2.4, Proposition B.1.1 states that $\mathbf{Z}_f$ has ϵ'-rank at most $R \times G$ with $\epsilon' = R\Gamma_1\epsilon$. That is, there exist rank $k \leq R \times G$ matrix $\boldsymbol{M}_k \in \mathbb{R}^{L \times (NT/L)}$ so that

$$\mathbf{Z}_f = \boldsymbol{M}_k + \boldsymbol{E}_k, \tag{B.13}$$

where $\|\boldsymbol{E}_k\|_\infty \leq \epsilon'$. Due to Property 2.2.1, it follows that $\|\boldsymbol{M}_k\|_\infty \leq R\Gamma_1\Gamma_2 + \epsilon'$. Under Property 4.1.2, we have $\sigma_k(\boldsymbol{M}_k) \geq c\sqrt{NT}/\sqrt{k}$ for some constant $c > 0$. Define

$$\Gamma = R\Gamma_1\Gamma_2 + \epsilon' \;=\; R\Gamma_1(\Gamma_2 + \epsilon). \tag{B.14}$$

Recall from Section 3.1, the elements of the imputed multivariate time series are simply the entries of the matrix $\widehat{\boldsymbol{M}}_f$ where $\widehat{\boldsymbol{M}}_f = \frac{1}{\hat{\rho}}\mathsf{HSVT}_k(\mathbf{Z}_X)$. That is, imputation in mSSA is carried out by applying HSVT to the stacked page matrix $\mathbf{Z}_X$.

All in all, the above description precisely meets the setup of Theorem A.2.1. To apply Theorem A.2.1, we require $\rho \geq C\log(NT)/\sqrt{NT}$ for $C > 0$ large enough. Note that

the number of columns in $\widehat{\boldsymbol{M}}_f$ is equal to NT/L for $L = \sqrt{\min(N,T)T}$ – for this choice of L, note that $NT/L \geq L$. Using $\sigma_k^2(\boldsymbol{M}_k) \geq cNT/k$, for some absolute constant $= c \geq 0$, and using Theorem A.2.1, we obtain

$$\mathbb{E}\big[\frac{1}{(NT/L)}\|\widehat{\boldsymbol{M}}_f - \mathbf{Z}_f\|_{2,\infty}^2\big] \tag{B.15}$$

$$\leq \frac{k(NT/L)(C\gamma^2 + \rho^2\epsilon' L)}{\rho^2 c^2 NT}\Big(\Gamma^2 + \frac{\gamma^2}{\rho^2}\Big) + \frac{C\gamma^2 k \log NT}{(NT/L)\rho^2} + \frac{C(\Gamma+\epsilon')^2}{(NT/L)} + 2(\epsilon')^2 + \frac{C}{(NT)^2} \tag{B.16}$$

Recall that $k \leq R \times G$, $\epsilon' = R\Gamma_1\epsilon$, and $\Gamma = R\Gamma_1(\Gamma_2 + \epsilon)$. Hence, simplifying (B.15), we obtain that

$$\mathbb{E}\big[\frac{1}{(NT/L)}\|\widehat{\boldsymbol{M}}_f - \mathbf{Z}_f\|_{2,\infty}^2\big] \tag{B.17}$$

$$\leq \tilde{C}\Bigg(\frac{RG(1+\rho^2 R\epsilon L)}{\rho^2 L}\Big(R^2(1+\epsilon^2) + \frac{1}{\rho^2}\Big) + \frac{RG\log NT}{(NT/L)\rho^2} + \frac{(R(1+\epsilon))^2}{(NT/L)} + (R\epsilon)^2\Bigg) \tag{B.18}$$

$$\leq \tilde{C}\Bigg(\frac{R^3 G\log NT}{\rho^4 L} + \frac{R^4 G(\epsilon+\epsilon^2)}{\rho^2}\Bigg), \tag{B.19}$$

where $\tilde{C} = C(c, \Gamma_1, \Gamma_2, \gamma)$ is a positive constant dependent on model parameters including $\Gamma_1, \Gamma_2, \gamma$.

It can be easily verified that for any matrix, $\boldsymbol{A} \in \mathbb{R}^{m\times n}$,

$$\frac{1}{mn}\|\boldsymbol{A}\|_F^2 \leq \frac{1}{n}\|\boldsymbol{A}\|_{\infty,2}^2. \tag{B.20}$$

Further, there is a one-to-one mapping of $\hat{f}_n(\cdot)$ (resp. $f_n(\cdot)$) to the entries of $\widehat{\boldsymbol{M}}_f$ (resp. $\mathbf{Z}_f$). Hence,

$$\mathsf{ImpErr}(N,T) = \mathbb{E}\big[\frac{1}{NT}\|\widehat{\boldsymbol{M}}_f - \mathbf{Z}_f\|_F^2\big] \tag{B.21}$$

Therefore, from (B.19), (B.20), and (B.21) it follows that

$$\mathsf{ImpErr}(N,T) \leq C(c,\Gamma_1,\Gamma_2,\gamma)\left(\frac{R^3 G \log NT}{\rho^4 L} + \frac{R^4 G(\epsilon+\epsilon^2)}{\rho^2}\right) \tag{B.22}$$

This completes the proof of Theorem 4.1.1.

B.2.2 Proof of Theorem 4.1.2

The forecasting algorithm, as described in Section 3.1, computes a linear model between the recent past and immediate future to forecast. We shall bound the forecasting error, $\mathsf{ForErr}(N,T,L)$ as defined in (4.2). We start with some setup and notations, followed by a key proposition that establishes the existence of a linear model under the setup of Theorem 4.1.2, and then conclude with detailed analysis of noisy, misspecified least-squares.

Setup, Notations. For $L \geq 1, k \geq 1$, Let

- $\mathbf{Z}_X \in \mathbb{R}^{L\times(NT/L)}$ denotes the stacked Page matrix induced by the observations $X_1(t),\ldots,X_N(t),\ t \in [T]$.
- $\mathbf{Z}_f \in \mathbb{R}^{L\times(NT/L)}$ denotes the stacked Page matrix induced by the latent time series $f_1(t),\ldots,f_N(t),\ t \in [T]$.
- $\bar{\mathbf{Z}}_X \in \mathbb{R}^{(L-1)\times(NT/L)}$ denotes the top $L-1$ rows of $\mathbf{Z}_X$.
- $\bar{\mathbf{Z}}_f \in \mathbb{R}^{(L-1)\times(NT/L)}$ denotes the top $L-1$ rows of $\mathbf{Z}_f$.

It is worth noting that $\mathbb{E}[\mathbf{Z}_X] = \rho\mathbf{Z}_f$ and hence the L-th row of these two matrices are related as

$$[\mathbf{Z}_X]_{L\cdot}^T = \rho[\mathbf{Z}_f]_{L\cdot}^T + \eta, \tag{B.23}$$

where $\eta \in \mathbb{R}^{(NT)/L}$ is a random vector with each component being independent, zero-mean with its distribution given as: it is 0 with probability $1-\rho$ and with probability ρ, due to Property 2.2.3, it equals a zero-mean sub-Gaussian random variable with

$\|\cdot\|_{\psi_2} \leq \gamma$. Therefore, using Lemma G.2 in [10], each component of η is an independent, zero-mean random variable with $\|\cdot\|_{\psi_2}$ bounded above by $C'(\gamma^2 + R\Gamma_1\Gamma_2)$ for some absolute constant $C' > 0$. Let $K = C'(\gamma^2 + R\Gamma_1\Gamma_2)$ and hence each component of η has $\|\cdot\|_{\psi_2}$ bounded by K.

Now, recall that for forecasting, we first apply the imputation algorithm (i.e. HSVT) to $\mathbf{Z}_X$ by replacing $\star$s, i.e. missing observations by 0 as well as setting all the entries in the last row equal to 0. Equivalently, the imputation algorithm is applied to $\bar{\mathbf{Z}}_X$ after setting all missing values to 0. Let $\widehat{\bar{\mathbf{Z}}}_f \in \mathbb{R}^{L-1\times(NT/L)}$ be the estimate produced from the imputation algorithm applied to $\bar{\mathbf{Z}}_X$. Under the setup of Theorem 4.1.1, by following arguments identical to that of Theorems A.2.1 and 4.1.1–in particular, refer to (B.19)–it follows that by selecting the right choice of $k \leq R \times G$, we have

$$\mathbb{E}\Big[\frac{1}{\sqrt{NT}}\|\widehat{\bar{\mathbf{Z}}}_f - \bar{\mathbf{Z}}_f\|_{2,\infty}^2\Big] \leq \tilde{C}\bigg(\frac{R^3 G \log NT}{\rho^4 L} + \frac{R^4 G(\epsilon+\epsilon^2)}{\rho^2}\bigg), \tag{B.24}$$

where $\tilde{C} = C(c, \Gamma_1, \Gamma_2, \gamma) > 0$ is a constant dependent on $c, \Gamma_1, \Gamma_2, \gamma$.

Now, the mSSA forecasting algorithm finds $\widehat{\beta} = \widehat{\beta}((X_1, \ldots, X_N), TL; k)$, by solving the following Ordinary Least Squares (OLS):

$$\widehat{\beta} \in \mathsf{minimize} \quad \|\frac{1}{\widehat{\rho}}[\mathbf{Z}_X]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T\beta\|_2^2 \quad \mathsf{over} \quad \beta \in \mathbb{R}^{L-1}. \tag{B.25}$$

And subsequently, $\widehat{\bar{\mathbf{Z}}}_f^T\widehat{\beta}$ is used as the estimate for $[\mathbf{Z}_f]_{L\cdot} \in \mathbb{R}^{NT/L}$, the Lth row of latent $\mathbf{Z}_f$. The goal is to bound the forecasting error $\mathsf{ForErr}(N, T, L)$, which is given by

$$\mathsf{ForErr}(N, T, L) = \mathbb{E}\Big[\frac{1}{(NT/L)}\|[\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T\widehat{\beta}\|_2^2\Big]. \tag{B.26}$$

Therefore, our interest is in bounding $\mathbb{E}\big[\|[\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T\widehat{\beta}\|_2^2\big]$.

Now, we recall from Proposition 4.1.2 that there exists $\beta^* \in \mathbb{R}^{L-1}$, such that

$$\| [\mathbf{Z}_f]_{L\cdot}^T - \bar{\mathbf{Z}}_f^T \beta^* \|_\infty \leq C_2 \epsilon,$$

where $C_2 \coloneqq R\Gamma_1(1 + \|\beta^*\|_1)$.

Bounding $\mathbb{E}\big[\| [\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\beta}\|_2^2\big]$. By (B.25) and (B.23)

$$\begin{aligned}
\|\frac{1}{\widehat{\rho}} [\mathbf{Z}_X]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\beta}\|_2^2 &\leq \|\frac{1}{\widehat{\rho}} [\mathbf{Z}_X]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \beta^*\|_2^2 \\
&= \|\frac{\rho}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} + \eta - \widehat{\bar{\mathbf{Z}}}_f^T \beta^*\|_2^2 \\
&= \|\frac{\rho}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \beta^*\|_2^2 + \|\eta\|_2^2 + 2\eta^T (\frac{\rho}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \beta^*). \qquad \text{(B.27)}
\end{aligned}$$

Also,

$$\begin{aligned}
\|\frac{1}{\widehat{\rho}} [\mathbf{Z}_X]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\beta}\|_2^2 &= \|\frac{\rho}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} + \eta - \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\beta}\|_2^2 \\
&= \|\frac{\rho}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\beta}\|_2^2 + \|\eta\|_2^2 + 2\eta^T ([\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\beta}). \qquad \text{(B.28)}
\end{aligned}$$

From (B.27) and (B.28)

$$\begin{aligned}
&\mathbb{E}\big[\|\frac{\rho}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\beta}\|_2^2\big] \qquad \text{(B.29)} \\
&\leq \mathbb{E}\big[\|\frac{\rho}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \beta^*\|_2^2\big] + 2\mathbb{E}\big[\eta^T \widehat{\bar{\mathbf{Z}}}_f^T (\beta^* - \widehat{\beta})\big] + 2\mathbb{E}\big[\eta^T \Big(\frac{\rho}{\widehat{\rho}} - 1\Big) [\mathbf{Z}_f]_{L\cdot}\big] \qquad \text{(B.30)}
\end{aligned}$$

η is independent of $\widehat{\bar{\mathbf{Z}}}_f$, β^*, and $\widehat{\rho}$; $\mathbb{E}[\eta] = \mathbf{0}$; thus, we have that

$$\mathbb{E}\big[\eta^T \widehat{\bar{\mathbf{Z}}}_f^T \beta^*\big] = 0, \quad \mathbb{E}\big[\eta^T \Big(\frac{\rho}{\widehat{\rho}} - 1\Big) [\mathbf{Z}_f]_{L\cdot}\big] = 0. \qquad \text{(B.31)}$$

By (B.25), we have $\widehat{\beta} = \widehat{\bar{\mathbf{Z}}}_f^{T,\dagger} \frac{1}{\widehat{\rho}} [\mathbf{Z}_X]_{L\cdot}$, where $\widehat{\bar{\mathbf{Z}}}_f^{T,\dagger}$ is pseudo-inverse of $\widehat{\bar{\mathbf{Z}}}_f^T$. That is,

$$\widehat{\beta} = \widehat{\bar{\mathbf{Z}}}_f^{T,\dagger} \frac{\rho}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} + \frac{1}{\widehat{\rho}} \widehat{\bar{\mathbf{Z}}}_f^{T,\dagger} \eta. \qquad \text{(B.32)}$$

Using cyclic and linearity of Trace operator; the independence properties of η; and

(B.32); we have

$$\begin{aligned}
\mathbb{E}[\eta^T \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\beta}] &= \mathbb{E}[\eta^T \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\bar{\mathbf{Z}}}_f^{T,\dagger} \frac{\rho}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} + \mathbb{E}[\frac{1}{\widehat{\rho}} \eta^T \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\bar{\mathbf{Z}}}_f^{T,\dagger} \eta] \\
&= \mathbb{E}[\eta]^T \mathbb{E}[\widehat{\bar{\mathbf{Z}}}_f^T \widehat{\bar{\mathbf{Z}}}_f^{T,\dagger} \frac{\rho}{\widehat{\rho}}] [\mathbf{Z}_f]_{L\cdot} + \mathbb{E}[\frac{1}{\widehat{\rho}} \mathsf{Tr}(\eta^T \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\bar{\mathbf{Z}}}_f^{T,\dagger} \eta)] \\
&= \mathbb{E}[\frac{1}{\widehat{\rho}} \mathsf{Tr}(\widehat{\bar{\mathbf{Z}}}_f^T \widehat{\bar{\mathbf{Z}}}_f^{T,\dagger} \eta\eta^T)] \\
&= \mathsf{Tr}\big(\mathbb{E}[\frac{1}{\widehat{\rho}} \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\bar{\mathbf{Z}}}_f^{T,\dagger}] \mathbb{E}[\eta\eta^T]\big) \\
&\leq \ C(\gamma) k/\rho, \qquad \text{(B.33)}
\end{aligned}$$

where $C(\gamma)$ is a function only of γ. To see the last inequality, we use various facts. First, by the definition of the HSVT algorithm $\widehat{\bar{\mathbf{Z}}}_f^T$ has rank at most k. Second, let $\widehat{\bar{\mathbf{Z}}}_f^T = \mathbf{U}\mathbf{S}\mathbf{V}^T$ be the singular value decomposition of $\widehat{\bar{\mathbf{Z}}}_f^T$, we have

$$\begin{aligned}
\widehat{\bar{\mathbf{Z}}}_f^T \widehat{\bar{\mathbf{Z}}}_f^{T,\dagger} &= \mathbf{U}\mathbf{S}\mathbf{V}^T \mathbf{V}\mathbf{S}^\dagger \mathbf{U}^T \\
&= \mathbf{U}\tilde{\boldsymbol{I}}\mathbf{U}^T, \qquad \text{(B.34)}
\end{aligned}$$

That is, $\frac{1}{\widehat{\rho}} \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\bar{\mathbf{Z}}}_f^{T,\dagger}$ is a positive semi-definite matrix and $\mathsf{Tr}(\frac{1}{\widehat{\rho}} \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\bar{\mathbf{Z}}}_f^{T,\dagger}) \leq k/\widehat{\rho}$. The matrix $\mathbb{E}[\eta\eta^T]$ is diagonal with all the non-zero entries on diagonal (variance of components of η) bounded above by a constant that depends on K. For a positive semi-definite matrix A and positive semi-definite diagonal matrix B, $\mathsf{Tr}(AB) \leq \|B\|_2 \mathsf{Tr}(A)$. For $\rho \geq C \log(NT)/\sqrt{NT}$ for large enough C, one can verify that $\mathbb{E}[1/\widehat{\rho}] \leq 2/\rho$. This completes the justification of the last step of (B.33).

Now consider the term $\|\frac{\rho}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \beta^*\|_2^2$. Note,

$$\begin{aligned}
\|\frac{\rho}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \beta^*\|_2^2 &= \|\big([\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \beta^*\big) + \big(\frac{\rho - \widehat{\rho}}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot}\big)\|_2^2 \\
&\leq 2\|\big([\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \beta^*\big)\|_2^2 + 2\|\frac{\rho - \widehat{\rho}}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot}\|_2^2. \qquad \text{(B.35)}
\end{aligned}$$

We will bound the two terms on the r.h.s of (B.35) separately. We now consider the

first term.

$$\| [\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \beta^* \|_2^2 \leq 2\| [\mathbf{Z}_f]_{L\cdot} - \bar{\mathbf{Z}}_f^T \beta^* \|_2^2 + 2\|\bar{\mathbf{Z}}_f^T \beta^* - \widehat{\bar{\mathbf{Z}}}_f^T \beta^* \|_2^2. \tag{B.36}$$

By Proposition 4.1.2

$$\| [\mathbf{Z}_f]_{L\cdot} - \bar{\mathbf{Z}}_f^T \beta^* \|_2 \leq \| [\mathbf{Z}_f]_{L\cdot} - \bar{\mathbf{Z}}_f^T \beta^* \|_\infty \sqrt{NT/L} \ \leq \ C_2 \epsilon \sqrt{NT/L}, \tag{B.37}$$

where we used the fact that for any $v \in \mathbb{R}^p$, $\|v\|_2 \leq \|v\|_\infty \sqrt{p}$. And,

$$\|\bar{\mathbf{Z}}_f^T \beta^* - \widehat{\bar{\mathbf{Z}}}_f^T \beta^* \|_2 = \|(\bar{\mathbf{Z}}_f - \widehat{\bar{\mathbf{Z}}}_f)^T \beta^* \|_2 \ \leq \ \|\bar{\mathbf{Z}}_f - \widehat{\bar{\mathbf{Z}}}_f \|_{2,\infty} \|\beta^*\|_1, \tag{B.38}$$

where we used the fact that for any $A \in \mathbb{R}^{q \times p}, v \in \mathbb{R}^p$, $\|Av\|_2 \leq \|A^T\|_{2,\infty} \|v\|_1$. Finally, note that

$$\| [\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\beta} \|_2^2 \leq 2\| \frac{\rho}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\beta} \|_2^2 + 2\| \frac{\rho - \widehat{\rho}}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} \|_2^2. \tag{B.39}$$

Using (B.29), (B.31), (B.33), (B.35),(B.36), (B.37), (B.38), and the bound in (B.39), we obtain

$$\mathbb{E}\big[\| [\mathbf{Z}_f]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T \widehat{\beta} \|_2^2\big] \tag{B.40}$$

$$\leq 4C(\gamma)k/\rho + 6\mathbb{E}\big[\| \frac{\rho - \widehat{\rho}}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} \|_2^2\big] + 2C_2\epsilon^2(NT/L) + 2\|\beta^*\|_1^2 \|\bar{\mathbf{Z}}_f - \widehat{\bar{\mathbf{Z}}}_f\|_{2,\infty}^2. \tag{B.41}$$

Now $L = \sqrt{\min(N,T)T}$, i.e. $NT/L = \sqrt{NT}$. Note that $\|\mathbf{Z}_f\|_\infty \leq R\Gamma_1\Gamma_2$. Hence, $\| [\mathbf{Z}_f]_{L\cdot} \|_2^2 \leq C(\Gamma_1, \Gamma_2) R^2 \sqrt{NT}$, for large enough constant $C(\Gamma_1, \Gamma_2)$ that may depend on Γ_1, Γ_2. Using the bounds derived in Lemma A.2.3, one can verify that $\mathbb{E}[(\frac{\rho - \widehat{\rho}}{\widehat{\rho}})^2] \leq C/\sqrt{NT}$ for large enough positive constant C. Therefore, we have that

$$6\mathbb{E}\big[\| \frac{\rho - \widehat{\rho}}{\widehat{\rho}} [\mathbf{Z}_f]_{L\cdot} \|_2^2\big] \leq C(\Gamma_1, \Gamma_2) R^2 \tag{B.42}$$

Using (B.24), (B.42), and the bound in (B.40); diving by $1/\sqrt{NT}$ on both sides; and noting $k \leq R \times G$, we obtain

$$\mathbb{E}\big[\frac{1}{(NT/L)}\| [\mathbf{Z}_f]_{L\cdot} - \widehat{\widehat{\mathbf{Z}}}_f^T \widehat{\beta}\|_2^2\big] \tag{B.43}$$

$$\leq C(c,\gamma,\Gamma_1,\Gamma_2)\left(\frac{RG}{\rho(NT/L)} + \frac{R^2}{(NT/L)} + R(1+\|\beta^*\|_1)\epsilon^2 + \|\beta^*\|_1^2\left(\frac{R^3 G \log NT}{\rho^4 L} + \frac{R^4 G(\epsilon+\epsilon^2)}{\rho^2}\right)\right) \tag{B.44}$$

$$\leq C(c,\gamma,\Gamma_1,\Gamma_2)\left(\max(1,\|\beta^*\|_1,\|\beta^*\|_1^2)\left(\frac{R^3 G \log NT}{\rho^4 L} + \frac{R^4 G(\epsilon+\epsilon^2)}{\rho^2}\right)\right) \tag{B.45}$$

Using (B.45) and noting that

$$\mathsf{ForErr}(N,T,L=\sqrt{NT}) = \mathbb{E}\big[\frac{1}{(NT/L)}\| [\mathbf{Z}_f]_{L\cdot} - \widehat{\widehat{\mathbf{Z}}}_f^T \widehat{\beta}\|_2^2\big]$$

completes the proof of Theorem 4.1.2.

B.2.3 Proof of Proposition 4.1.2

Recall the definition of the Hankel matrix representation (Definition 2.1.1), we define the *stacked Hankel matrix* of N time series over T time steps as follows. Given N latent time series $f_1, \ldots, f_N$, let $\mathbf{H}_f^{(n)}$ denote the Hankel matrix representation of the latent time series $f_n(t)$. Further, consider the stacked Hankel matrix $\mathbf{H}_f$, obtained by a column-wise concatenation of the Page matrices $\mathbf{H}_f^{(1)}, \ldots, \mathbf{H}_f^{(N)}$. Specifically,

$$\mathbf{H}_f = \begin{bmatrix} \mathbf{H}_f^{(1)} & \mathbf{H}_f^{(2)} & \ldots & \mathbf{H}_f^{(N)} \end{bmatrix}. \tag{B.46}$$

As indicated in Proposition B.1.1, $\mathbf{H}_f$ has ϵ'-rank bounded by $R \times G < L$ with $\epsilon' = R\Gamma_1\epsilon$. That is, there exists a matrix $\mathsf{M} \in \mathbb{R}^{T \times NT}$ such that,

$$\text{rank}(\mathsf{M}) \leq RG, \quad \|\mathbf{H}_f - \mathsf{M}\|_\infty \leq \epsilon' \tag{B.47}$$

Since $\text{rank}(\mathsf{M}) \leq RG$, it must be the case that within the last RG rows of M, there exists at least one row, which we denote as r^*, that can be written as a linear com-

bination of at most RG rows above it, which we denote as $r_1, \ldots, r_{RG}$. Specifically there exists a vector $\theta := (\theta_1, \ldots, \theta_{RG}) \in \mathbb{R}^{RG}$ such that

$$[\mathsf{M}]_{r^*,\cdot} = \sum_{\ell=1}^{RG} \theta_\ell \, [\mathsf{M}]_{r_\ell,\cdot} \tag{B.48}$$

Hence for $j \in [T - L + 1]$,

$$\left| [\mathbf{H}_f]_{r^*,j} - \sum_{\ell=1}^{RG} \theta_\ell \, [\mathbf{H}_f]_{r_\ell,j} \right| \tag{B.49}$$

$$= \left| [\mathbf{H}_f]_{r^*,j} \pm [\mathsf{M}]_{r^*,j} - \sum_{\ell=1}^{RG} \theta_\ell \, [\mathbf{H}_f]_{r_\ell,j} \pm \sum_{\ell=1}^{RG} \theta_\ell \, [\mathsf{M}]_{r_\ell,t} \right| \tag{B.50}$$

$$\tag{B.51}$$

$$\leq \left| [\mathbf{H}_f]_{r^*,j} - [\mathsf{M}]_{r^*,j} \right| + \left| \sum_{\ell=1}^{RG} \theta_\ell \, [\mathbf{H}_f]_{r_\ell,j} - \sum_{\ell=1}^{RG} \theta_\ell \, [\mathsf{M}]_{r_\ell,t} \right| + \left| [\mathsf{M}]_{r^*,j} - \sum_{\ell=1}^{RG} \theta_\ell \, [\mathsf{M}]_{r_\ell,t} \right| \tag{B.52}$$

$$= \left| [\mathbf{H}_f]_{r^*,j} - [\mathsf{M}]_{r^*,j} \right| + \left| \sum_{\ell=1}^{RG} \theta_\ell ([\mathbf{H}_f]_{r_\ell,j} - [\mathsf{M}]_{r_\ell,t}) \right| \tag{B.53}$$

$$\leq \epsilon' + \|\theta\|_1 \| \, [\mathbf{H}_f]_{r_\ell,j} - [\mathsf{M}]_{r_\ell,t} \, \|_\infty \tag{B.54}$$

$$\leq R\Gamma_1(1 + \|\theta\|_1)\epsilon. \tag{B.55}$$

Using (B.55) and observing that every entry of $[\mathbf{Z}_f]_{L^\cdot}$ appears within $[\mathbf{H}_f]_{r^*,\cdot}$, implies that by appropriately selecting entries in $[\mathbf{H}_f]$, there exists $\beta^* \in \mathbb{R}^{L-1}$,

$$\| \, [\mathbf{Z}_f]_{L^\cdot}^T - \bar{\mathbf{Z}}_f^T \beta^* \|_\infty \leq R\Gamma_1(1 + \|\beta\|_1)\epsilon,$$

where the non-zero entries in β^* correspond to the entries of θ. Noting that $\theta \in \mathbb{R}^{RG}$ implies $\|\beta^*\|_0 \leq RG$. This completes the proof.

B.2.4 Proof of Theorem 4.2.1

Setup, Notations. For $L \geq 1, k \geq 1$, Let

- $\mathbf{Z}_X \in \mathbb{R}^{L \times (NT/L)}$ denotes the stacked Page matrix induced by the observations $X_1(t), \ldots, X_N(t),\ t \in [T]$.
- $\mathbf{Z}_{X^2} \in \mathbb{R}^{L \times (NT/L)}$ denotes the stacked Page matrix induced by the squared observations $X_1^2(t), \ldots, X_N^2(t),\ t \in [T]$.
- $\mathbf{Z}_f \in \mathbb{R}^{L \times (NT/L)}$ denotes the stacked Page matrix induced by the latent time series $f_1(t), \ldots, f_N(t),\ t \in [T]$.
- $\mathbf{Z}_{f^2} \in \mathbb{R}^{L \times (NT/L)}$ denotes the stacked Page matrix induced by the squared latent time series $f_1^2(t), \ldots, f_N^2(t),\ t \in [T]$.
- $\mathbf{Z}_{\sigma^2} \in \mathbb{R}^{L \times (NT/L)}$ denotes the stacked Page matrix induced by the time-varying variance $\sigma_1^2(t), \ldots, \sigma_N^2(t),\ t \in [T]$.
- $\mathbf{Z}_{f^2+\sigma^2} = \mathbf{Z}_{\sigma^2} + \mathbf{Z}_{f^2} \in \mathbb{R}^{L \times (NT/L)}$.

Recalling that $\rho = 1$, we note that

$$\mathbb{E}[\mathbf{Z}_X] = \mathbf{Z}_f, \quad \mathbb{E}[\mathbf{Z}_{X^2}] = \mathbf{Z}_{f^2+\sigma^2}. \tag{B.56}$$

Further, from the definition of the variance estimation algorithm, we recall

$$\widehat{\mathbf{Z}}_f \coloneqq \frac{1}{\widehat{\rho}} \mathsf{HSVT}_k(\mathbf{Z}_X) \tag{B.57}$$

$$\widehat{\mathbf{Z}}_{f^2+\sigma^2} \coloneqq \frac{1}{\widehat{\rho}} \mathsf{HSVT}_k(\mathbf{Z}_{X^2}) \tag{B.58}$$

We denote

- $\widehat{\mathbf{Z}}_{f^2} = \widehat{\mathbf{Z}}_f \circ \widehat{\mathbf{Z}}_f$
- $\widehat{\mathbf{Z}}_{\sigma^2} = \max\left(\widehat{\mathbf{Z}}_{f^2+\sigma^2} - \widehat{\mathbf{Z}}_{f^2}, \mathbf{0}\right)$,

where $\mathbf{0} \in \mathbb{R}^{L \times (NT/L)}$ is a matrix of all zeroes, and we apply the $\max(\cdot)$ above entry-wise. We remind the reader that the output of the variance estimation algorithm is

$\widehat{\mathbf{Z}}_{\sigma^2}$. Thus, we have

$$\frac{1}{NT}\sum_{n=1}^{N}\sum_{t=1}^{T}\left(\sigma_n(t)^2-\hat{\sigma}_n^2(t)\right)^2=\frac{1}{NT}\|\mathbf{Z}_{\sigma^2}-\widehat{\mathbf{Z}}_{\sigma^2}\|_F^2.$$

Initial Decomposition. Note that since $\sigma_n^2(t)\geq 0$ for $n\in[N]$ and $t\in[T]$, we have that

$$\begin{aligned}
&\frac{1}{NT}\|\mathbf{Z}_{\sigma^2}-\widehat{\mathbf{Z}}_{\sigma^2}\|_F^2 && \text{(B.59)}\\
&\leq \frac{1}{NT}\|\mathbf{Z}_{\sigma^2}-(\widehat{\mathbf{Z}}_{f^2+\sigma^2}-\widehat{\mathbf{Z}}_{f^2})\|_F^2 && \text{(B.60)}\\
&= \frac{1}{NT}\|\mathbf{Z}_{f^2+\sigma^2}-\mathbf{Z}_{f^2}-(\widehat{\mathbf{Z}}_{f^2+\sigma^2}-\widehat{\mathbf{Z}}_{f^2})\|_F^2 && \text{(B.61)}\\
&\leq \frac{2}{NT}\|\mathbf{Z}_{f^2+\sigma^2}-\widehat{\mathbf{Z}}_{f^2+\sigma^2}\|_F^2+\frac{2}{NT}\|\mathbf{Z}_{f^2}-\widehat{\mathbf{Z}}_{f^2}\|_F^2 && \text{(B.62)}
\end{aligned}$$

We bound the two terms on the r.h.s of (B.62) separately.

Bounding $\mathbb{E}[\|\mathbf{Z}_{f^2}-\widehat{\mathbf{Z}}_{f^2}\|_F^2]$.

$$\begin{aligned}
\|\mathbf{Z}_{f^2}-\widehat{\mathbf{Z}}_{f^2}\|_F^2 &= \sum_{n=1}^{N}\sum_{t=1}^{T}\left(f_n^2(t)-\hat{f}_n^2(t)\right)^2 && \text{(B.63)}\\
&= \sum_{n=1}^{N}\sum_{t=1}^{T}\left(f_n(t)-\hat{f}_n(t)\right)^2\left(f_n(t)+\hat{f}_n(t)\right)^2 && \text{(B.64)}\\
&\leq \left[\max_{n\in[N],t\in[T]}\left(f_n(t)+\hat{f}_n(t)\right)^2\right]\left[\sum_{n=1}^{N}\sum_{t=1}^{T}\left(f_n(t)-\hat{f}_n(t)\right)^2\right] && \text{(B.65)}\\
&\leq C(\Gamma_1,\Gamma_2,\Gamma_3)R^2\left[\sum_{n=1}^{N}\sum_{t=1}^{T}\left(f_n(t)-\hat{f}_n(t)\right)^2\right] && \text{(B.66)}\\
&= C(\Gamma_1,\Gamma_2,\Gamma_3)R^2\|\mathbf{Z}_f-\widehat{\mathbf{Z}}_f\|_F^2 && \text{(B.67)}
\end{aligned}$$

Bounding $\|\mathbf{Z}_{f^2+\sigma^2}-\widehat{\mathbf{Z}}_{f^2+\sigma^2}\|_F^2$. To bound $\|\mathbf{Z}_{f^2+\sigma^2}-\widehat{\mathbf{Z}}_{f^2+\sigma^2}\|_F^2$, we modify the proof of Theorem 4.1.1 in a straightforward manner. The need for the modification is that Theorem 4.1.1 was proven for the case where the coordinate wise noise, $\eta_n(t)=X_n(t)-f_n(t)$ are independent sub-gaussian random variables, and $\|\eta\|_{\psi_2}\leq\gamma$.

However, one can verify that $X_n^2(t) - f_n^2(t) - \sigma_n^2(t)$ is a sub-exponential random variable with $\|\cdot\|_{\psi_1}$ norm bounded as

$$\begin{aligned}
\|X_n^2(t) - f_n^2(t) - \sigma_n^2(t)\|_{\psi_1} &\leq \|X_n^2(t)\|_{\psi_1} && \text{(B.68)}\\
&= \|f_n^2(t) + 2f_n(t)\eta_n(t) + \eta_n^2(t)\|_{\psi_1} && \text{(B.69)}\\
&\leq 2\|f_n^2(t)\|_{\psi_1} + 2\|\eta_n^2(t)\|_{\psi_1} && \text{(B.70)}\\
&= 2\|f_n(t)\|_{\psi_2}^2 + 2\|\eta_n(t)\|_{\psi_2}^2 && \text{(B.71)}\\
&\leq C(\Gamma_1, \Gamma_2)R^2 + 2\gamma^2 && \text{(B.72)}\\
&\leq C(\Gamma_1, \Gamma_2, \gamma)R^2, && \text{(B.73)}
\end{aligned}$$

where we have use the standard facts that for a random variable A, $\|A - \mathbb{E}[A]\|_{\psi_1} \leq \|A\|_{\psi_1}$ and $\|A^2\|_{\psi_1} = \|A\|_{\psi_2}^2$.

Further, note that by using Properties 2.2.1, 2.2.2, 4.2.1, and 4.2.2, and a straightforward modification of Proposition B.1.1, we have

$$\begin{aligned}
\operatorname{rank}(\mathbf{Z}_{f^2+\sigma^2)}) &\leq \operatorname{rank}(\mathbf{Z}_{f^2}) + \operatorname{rank}(\mathbf{Z}_{\sigma^2}) && \text{(B.74)}\\
&\leq (RG)^2 + (R'G'), && \text{(B.75)}
\end{aligned}$$

where we have used that for any two matrices $\boldsymbol{A}, \boldsymbol{B}$, we have $\operatorname{rank}(\boldsymbol{A} \circ \boldsymbol{A}) \leq \operatorname{rank}(\boldsymbol{A})^2$, where $\circ$ denotes Hadamard product, and $\operatorname{rank}(\boldsymbol{A} + \boldsymbol{B}) \leq \operatorname{rank}(\boldsymbol{A}) + \operatorname{rank}(\boldsymbol{B})$. We define $\tilde{k} \coloneqq (RG)^2 + (R'G')$.

Modified Theorem 4.1.1. Below, we state the modified version of Theorem 4.1.1 to get our desired result.

Lemma B.2.1 (Imputation Error)**.** *Let the conditions of Theorem 4.2.1 hold. Then,*

$$\begin{aligned}
&\mathbb{E}\Big[\max_{j\in[L]} \frac{1}{(NT/L)} \| \left[\mathbf{Z}_{f^2+\sigma^2}\right]_{L,\cdot}^T - \left[\widehat{\mathbf{Z}}_{f^2+\sigma^2}\right]_{L,\cdot}^T \|_2^2\Big] && \text{(B.76)}\\
&\leq C(\Gamma_1, \Gamma_2, \Gamma_1', \Gamma_2', \gamma, R, R') \left(\frac{G^2 G' \log^2 NT}{L}\right), && \text{(B.77)}
\end{aligned}$$

where $C(\Gamma_1, \Gamma_2, \Gamma_1', \Gamma_2', \gamma, R, R')$ *is a term that depends only polynomially on* Γ_1, Γ_2, Γ_1', Γ_2', γ, R, R'.

Proof. To reduce redundancy, we provide an overview of the argument needed for this proof, focusing only the parts of the arguments made in Theorem 4.1.1 that need to be modified. For ease of exposition, we let $\tilde{C} = C(\Gamma_1, \Gamma_2, \Gamma_1', \Gamma_2', \gamma, R, R')$. We begin by matching notation with that used in Theorem 4.1.1; in particular with respect to $\rho, k, \epsilon, \Gamma$. Under the setup of Theorem 4.2.1, we have $\rho = 1$, $k = \tilde{k}$, $\epsilon = 0$, $\Gamma \leq \tilde{C}$ Further, recall the definition of $\boldsymbol{Y}, \boldsymbol{M}, p, q, \sigma$ from Appendix A.2.1. We will now use $\boldsymbol{Y} = \mathbf{Z}_{X^2}$, and $\boldsymbol{M} = \mathbf{Z}_{f^2+\sigma^2}, \sigma = \gamma$, $p = (NT/L), q = L$. One can verify that there is only required change to the proof of Theorem 4.1.1; in particular, in the argument made to prove Theorem A.2.1, we need to re-define events E_2, E_3, E_4 in (A.23), (A.24), (A.25) for the case where $(\boldsymbol{Y} - \boldsymbol{M})_{ij}$ is mean-zero sub-exponential. Using the result of Theorem G.1 in [10], which bounds the operator norm of a matrix with sub-exponential mean-zero entries, we have with probability at least $1 - 1/((NT)^{10})$

$$\|\boldsymbol{Y} - \boldsymbol{M}\|_2 \leq \tilde{C}\sqrt{(NT/L)}\log^2 NT \tag{B.78}$$

As a result (B.78), and standard concentration inequalities for sub-exponential random variables, we have the modified events, $\tilde{E}_2, \tilde{E}_3, \tilde{E}_4$.

$$\tilde{E}_2 := \Big\{\|\boldsymbol{Y} - \rho\boldsymbol{M}\|_2 \leq \tilde{C}\sqrt{(NT/L)}\log^2 NT\Big\}, \tag{B.79}$$

$$\tilde{E}_3 := \Big\{\|\boldsymbol{Y} - \rho\boldsymbol{M}\|_{\infty,2}, \|\boldsymbol{Y} - \rho\boldsymbol{M}\|_{2,\infty} \leq \tilde{C}\sqrt{(NT/L)}\log^2 NT\Big\}, \tag{B.80}$$

$$\tilde{E}_4 := \Big\{\max_{j\in[q]}\|\varphi^{\boldsymbol{B}}_{\sigma_k(\boldsymbol{B})}\Big(\boldsymbol{Y}^T_{j\cdot} - \rho\boldsymbol{M}^T_{j\cdot}\Big)\|_2^2 \leq \tilde{C}\tilde{k}\log^2(NT/L)\Big\}, \tag{B.81}$$

Using these modified events in the proofs of Theorem A.2.1 and Theorem 4.1.1, and appropriately simplifying leads to the desired result. □

By Lemma B.2.1 and (B.20), we have that

$$\frac{1}{NT}\mathbb{E}[\|\mathbf{Z}_{f^2+\sigma^2} - \widehat{\mathbf{Z}}_{f^2+\sigma^2}\|_F^2 \leq \mathbb{E}\big[\max_{j\in[L]}\frac{1}{(NT/L)}\|\,[\mathbf{Z}_{f^2+\sigma^2}]^T_{L,\cdot} - \Big[\widehat{\mathbf{Z}}_{f^2+\sigma^2}\Big]^T_{L,\cdot}\,\|_2^2\big] \tag{B.82}$$

$$\leq C(\Gamma_1, \Gamma_2, \Gamma_1', \Gamma_2', \gamma, R, R') \left(\frac{G^2 G' \log^2 NT}{L}. \right). \quad \text{(B.83)}$$

Completing proof. Substituting (B.67) and (B.83) into (B.62) and letting $L = \sqrt{NT}$

$$\frac{1}{NT} \|\mathbf{Z}_{\sigma^2} - \widehat{\mathbf{Z}}_{\sigma^2}\|_F^2 \leq C(\Gamma_1, \Gamma_2, \Gamma_3, \Gamma_1', \Gamma_2', \gamma, R, R') \left(\frac{G^2 G' \log^2 NT}{\sqrt{NT}}. \right). \quad \text{(B.84)}$$

This completes the proof.

B.2.5 Proof of Theorem 4.2.2

The variance forecasting algorithm, as described in Section 3.2, computes two linear models: one between the recent past and immediate future for the observations $X_1(\cdot), \ldots, X_N(\cdot)$, and one for the squared observations $X_1^2(\cdot), \ldots, X_N^2(\cdot)$. We shall bound the variance forecasting error, defined as:

$$\mathsf{ForErrVar}(N, T, L) = \frac{L}{NT} \sum_{n=1}^{N} \sum_{m'=1}^{T/L} \mathbb{E}\big[(\sigma_n^2(L \times m') - \bar{\sigma^2}_n(L \times m'))^2\big]. \quad \text{(B.85)}$$

Setup, Notations. For $L \geq 1, k \geq 1$, Let

- $\mathbf{Z}_X \in \mathbb{R}^{L \times (NT/L)}$ denotes the stacked Page matrix induced by the observations $X_1(t), \ldots, X_N(t),\ t \in [T]$.
- $\mathbf{Z}_{X^2} \in \mathbb{R}^{L \times (NT/L)}$ denotes the stacked Page matrix induced by the squared observations $X_1^2(t), \ldots, X_N^2(t),\ t \in [T]$.
- $\mathbf{Z}_f \in \mathbb{R}^{L \times (NT/L)}$ denotes the stacked Page matrix induced by the latent time series $f_1(t), \ldots, f_N(t),\ t \in [T]$.
- $\mathbf{Z}_{f^2} \in \mathbb{R}^{L \times (NT/L)}$ denotes the stacked Page matrix induced by the square of the latent time series $f_1^2(t), \ldots, f_N^2(t),\ t \in [T]$.
- $\mathbf{Z}_{\sigma^2} \in \mathbb{R}^{L \times (NT/L)}$ denotes the stacked Page matrix induced by the latent time-varying variance $\sigma_1^2(t), \ldots, \sigma_N^2(t),\ t \in [T]$.

- $\mathbf{Z}_{f^2+\sigma^2} = \mathbf{Z}_{\sigma^2} + \mathbf{Z}_{f^2} \in \mathbb{R}^{L\times(NT/L)}$.
- $\bar{\mathbf{Z}}_{X^2} \in \mathbb{R}^{(L-1)\times(NT/L)}$ denotes the top $L-1$ rows of $\mathbf{Z}_{X^2}$.
- $\bar{\mathbf{Z}}_{f^2+\sigma^2} \in \mathbb{R}^{(L-1)\times(NT/L)}$ denotes the top $L-1$ rows of $\mathbf{Z}_{f^2+\sigma^2}$.
- $\bar{\mathbf{Z}}_{f^2} \in \mathbb{R}^{(L-1)\times(NT/L)}$ denotes the top $L-1$ rows of $\mathbf{Z}_{f^2}$.

Recall, under the setting of Theorem 4.2.2 and Theorem 4.2.1, that $\mathbb{E}[\mathbf{Z}_X^2] - \mathbb{E}[\mathbf{Z}_X] \circ \mathbb{E}[\mathbf{Z}_X] = \mathbf{Z}_{f^2+\sigma^2} - \mathbf{Z}_{f^2} = \mathbf{Z}_{\sigma^2}$. Let $\zeta_n(t) = X_n^2(t) - f_n^2(t) - \sigma_n^2(t)$ be the sub-exponential noise with $\|\cdot\|_{\psi_1}$ norm bounded by $K := C(\Gamma_1, \Gamma_2, \gamma)R^2$ (see B.73). Then we have that:

$$[\mathbf{Z}_{X^2}]_{L\cdot}^T = [\mathbf{Z}_{\sigma^2}]_{L\cdot}^T + [\mathbf{Z}_f]_{L\cdot}^T + \zeta, \tag{B.86}$$

where we use $\zeta \in \mathbb{R}^{(NT)/L}$ to denote the per-step noise modeled by a zero-mean sub-Exponential random vector where each component is independent. Now, recall that for the variance forecasting algorithm, we first apply the imputation algorithm (i.e. HSVT) twice: once to $\bar{\mathbf{Z}}_X$, and once to $\bar{\mathbf{Z}}_{X^2}$. Let $\widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2} \in \mathbb{R}^{L-1\times(NT/L)}$ be the estimate produced from the imputation algorithm applied to $\bar{\mathbf{Z}}_{X^2}$. Using the results in Lemma B.2.1, follows that by selecting the right choice of $k \le (RG)^2 + (R'G')$, we have

$$\mathbb{E}\Big[\frac{1}{\sqrt{NT}}\|\widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2} - \bar{\mathbf{Z}}_{f^2+\sigma^2}\|_{2,\infty}^2\Big] \le C(\Gamma_1, \Gamma_2, \Gamma_1', \Gamma_2', \gamma, R, R')\left(\frac{G^2G'\log^2 NT}{L}\right), \tag{B.87}$$

where $C(\Gamma_1, \Gamma_2, \Gamma_1', \Gamma_2', \gamma, R, R')$ is a term that depends only polynomially on Γ_1, Γ_2, Γ_1', Γ_2', γ, R, R'.

Now, the mSSA forecasting algorithm finds $\widehat{\beta}_1$ and $\widehat{\beta}_2$ by solving the following two Ordinary Least Squares (OLS):

$$\widehat{\beta}_1 \in \mathsf{minimize} \quad \|\frac{1}{\widehat{\rho}}[\mathbf{Z}_X]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T\beta\|_2^2 \quad \mathsf{over} \quad \beta \in \mathbb{R}^{L-1}, \tag{B.88}$$

$$\widehat{\beta}_2 \in \mathsf{minimize} \quad \|\frac{1}{\widehat{\rho}}[\mathbf{Z}_{X^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2}^T\beta\|_2^2 \quad \mathsf{over} \quad \beta \in \mathbb{R}^{L-1}. \tag{B.89}$$

And use $\widehat{\bar{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\beta}_2 - \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1 \circ \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1$ as an estimate for $[\mathbf{Z}_{\sigma^2}]_{L\cdot} \in \mathbb{R}^{NT/L}$, the Lth row of latent variance $\mathbf{Z}_{\sigma^2}$. The goal is to bound the forecasting error $\mathsf{ForErrVar}(N,T,L)$, which is equivalently given by

$$\mathsf{ForErrVar}(N,T,L) = \mathbb{E}\Big[\frac{1}{(NT/L)}\| [\mathbf{Z}_{\sigma^2}]_{L\cdot} - (\widehat{\bar{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\beta}_2 - \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1 \circ \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1)\|_2^2\Big]. \quad \text{(B.90)}$$

Note that,

$$\begin{aligned}
&\mathbb{E}\left\|[\mathbf{Z}_{\sigma^2}]_{L\cdot} - (\widehat{\bar{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\beta}_2 - \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1 \circ \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1)\right\|_2^2 && \text{(B.91)}\\
=\ &\mathbb{E}\left\|[\mathbf{Z}_{f^2+\sigma^2}]_{L\cdot} - [\mathbf{Z}_{f^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\beta}_2 + \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1 \circ \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1\right\|_2^2 && \text{(B.92)}\\
\leq\ &2\mathbb{E}\left\|[\mathbf{Z}_{f^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1 \circ \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1\right\|_2^2 + 2\mathbb{E}\left\|[\mathbf{Z}_{f^2+\sigma^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\beta}_2\right\|_2^2 && \text{(B.93)}
\end{aligned}$$

Therefore, our interest is in bounding the two terms $\mathbb{E}\left\|[\mathbf{Z}_{f^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1 \circ \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1\right\|_2^2$ and $\mathbb{E}\left\|[\mathbf{Z}_{f^2+\sigma^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\beta}_2\right\|_2^2$.

Bounding $\mathbb{E}\left\|[\mathbf{Z}_{f^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1 \circ \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1\right\|_2^2$.

First, denote the t-th element of $\widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1 \circ \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1$ by $\bar{f}_n(tL)$. That is, for $t \in [P]$ $\bar{f}_n(tL) = \left(\left[\widehat{\bar{\mathbf{Z}}}_f\right]^T_{\cdot,t}\widehat{\beta}_1\right)^2$. Recall that $P \coloneqq \lfloor NT/L \rfloor$

$$\begin{aligned}
\left\|[\mathbf{Z}_{f^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1 \circ \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1\right\|_2^2 &= \sum_{n=1}^N\sum_{t=1}^P \Big(f_n^2(tL) - \bar{f}_n^2(tL)\Big)^2\\
&= \sum_{n=1}^N\sum_{t=1}^P \Big(f_n^2(tL) - \bar{f}_n(tL)\Big)^2\Big(f_n(tL) + \bar{f}_n(tL)\Big)^2\\
&\leq \left[\max_{n\in[N],t\in[P]}\Big(f_n(tL) + \hat{f}_n(tL)\Big)^2\right]\left[\sum_{n=1}^N\sum_{t=1}^P \Big(f_n(tL) - \hat{f}_n(tL)\Big)^2\right]\\
&\leq C(\Gamma_1,\Gamma_2,\Gamma_3)R^2\left[\sum_{n=1}^N\sum_{t=1}^P \Big(f_n(tL) - \bar{f}_n(tL)\Big)^2\right]\\
&= C(\Gamma_1,\Gamma_2,\Gamma_3)R^2\|[\mathbf{Z}_f]_{L,\cdot} - \widehat{\bar{\mathbf{Z}}}^T_f\widehat{\beta}_1\|_2^2 \qquad \text{(B.94)}
\end{aligned}$$

Thus, this term is related to the bound of the mean forecasting error in Theorem 4.1.2; specifically, we have,

$$\frac{L}{NT}\mathbb{E}\left\|[\mathbf{Z}_{f^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T\widehat{\beta}_1 \circ \widehat{\bar{\mathbf{Z}}}_f^T\widehat{\beta}_1\right\|_2^2 \leq C(\Gamma_1,\Gamma_2,\Gamma_3)R^2\frac{L}{NT}\mathbb{E}\|[\mathbf{Z}_f]_{L,\cdot} - \widehat{\bar{\mathbf{Z}}}_f^T\widehat{\beta}_1\|_2^2$$
$$\leq C_1(c,\gamma,\Gamma_1,\Gamma_2,\Gamma_3,R)\max(1,\|\beta_1^*\|_1,\|\beta_1^*\|_1^2)\Big(\frac{G\log NT}{\sqrt{NT}}\Big).$$

Where $C_1(\Gamma_1,\Gamma_2,\Gamma_1',\Gamma_2',\gamma,R,R')$ is a term that depends only polynomially on Γ_1, Γ_2, Γ_1', Γ_2', γ, R, R', and β_1^* is the β^* defined in Proposition 4.1.2.

Bounding $\mathbb{E}\left\|[\mathbf{Z}_{f^2+\sigma^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2}^T\widehat{\beta}_2\right\|_2^2$. To bound $\mathbb{E}\left\|[\mathbf{Z}_{f^2+\sigma^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2}^T\widehat{\beta}_2\right\|_2^2$, we use a very similar proof to the one used to prove Theorem 4.1.2. The main difference in the setup of this theorem is the subexponential noise vector ζ with $\|\zeta_i\|_{\psi_1} < C(\Gamma_1,\Gamma_2,\gamma)R^2$ and that $\text{rank}(\mathbf{Z}_{f^2+\sigma^2}) \leq (RG)^2 + (R'G')$. We include the full proof for completeness. Finally, note that Property 4.2.1, which is a slightly modified version of Proposition 4.1.1 for the setup of Theorem 4.2.2 shows that there exists $\beta_2^* \in \mathbb{R}^{L-1}$, such that

$$\|\,[\mathbf{Z}_{f^2+\sigma^2}]_{L\cdot}^T - \bar{\mathbf{Z}}_{f^2+\sigma^2}^T\beta_2^*\|_\infty = 0, \tag{B.95}$$

By (B.88) and (B.86)

$$\begin{aligned}\|\,[\mathbf{Z}_{X^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2}^T\widehat{\beta}_2\|_2^2 &\leq \|\,[\mathbf{Z}_{X^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2}^T\beta_2^*\|_2^2 \\ &= \|\,[\mathbf{Z}_{f^2+\sigma^2}]_{L\cdot} + \zeta - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2}^T\beta_2^*\|_2^2 \\ &= \|\,[\mathbf{Z}_{f^2+\sigma^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2}^T\beta_2^*\|_2^2 + \|\zeta\|_2^2 + 2\zeta^T([\mathbf{Z}_{f^2+\sigma^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2}^T\beta_2^*).\end{aligned} \tag{B.96}$$

Also,

$$\begin{aligned}\|\,[\mathbf{Z}_{X^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2}^T\widehat{\beta}_2\|_2^2 &= \|\,[\mathbf{Z}_{f^2+\sigma^2}]_{L\cdot} + \zeta - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2}^T\widehat{\beta}_2\|_2^2 \\ &= \|\,[\mathbf{Z}_{f^2+\sigma^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2}^T\widehat{\beta}_2\|_2^2 + \|\zeta\|_2^2 + 2\zeta^T([\mathbf{Z}_{f^2+\sigma^2}]_{L\cdot} - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2}^T\widehat{\beta}_2).\end{aligned} \tag{B.97}$$

From (B.96) and (B.97)

$$\mathbb{E}\big[\| \left[\mathbf{Z}_{f^2+\sigma^2}\right]_{L\cdot} - \widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\beta}_2\|_2^2\big] \tag{B.98}$$

$$\leq \mathbb{E}\big[\| \left[\mathbf{Z}_{f^2+\sigma^2}\right]_{L\cdot} - \widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}\beta_2^*\|_2^2\big] + 2\mathbb{E}\big[\zeta^T\widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}(\beta_2^* - \widehat{\beta}_2)\big] \tag{B.99}$$

ζ is independent of $\widehat{\widehat{\mathbf{Z}}}_{f^2+\sigma^2}$, β_2^*, and $\widehat{\rho}$; $\mathbb{E}[\eta] = \mathbf{0}$; thus, we have that

$$\mathbb{E}\big[\zeta^T\widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}\beta_2^*\big] = 0 \tag{B.100}$$

By (B.88), we have $\widehat{\beta}_2 = \widehat{\widehat{\mathbf{Z}}}^{T,\dagger}_{f^2+\sigma^2}\left[\mathbf{Z}_{X^2}\right]_{L\cdot}$, where $\widehat{\widehat{\mathbf{Z}}}^{T,\dagger}_{f^2+\sigma^2}$ is pseudo-inverse of $\widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}$. That is,

$$\widehat{\beta}_2 = \widehat{\widehat{\mathbf{Z}}}^{T,\dagger}_{f^2+\sigma^2}\left[\mathbf{Z}_{f^2+\sigma^2}\right]_{L\cdot} + \widehat{\widehat{\mathbf{Z}}}^{T,\dagger}_{f^2+\sigma^2}\zeta. \tag{B.101}$$

Using cyclic and linearity of Trace operator; the independence properties of ζ; and (B.101); we have

$$\begin{aligned}
\mathbb{E}[\zeta^T\widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\beta}_2] &= \mathbb{E}[\zeta^T\widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\widehat{\mathbf{Z}}}^{T,\dagger}_{f^2+\sigma^2}\left[\mathbf{Z}_{f^2+\sigma^2}\right]_{L\cdot}] + \mathbb{E}[\zeta^T\widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\widehat{\mathbf{Z}}}^{T,\dagger}_{f^2+\sigma^2}\zeta] \\
&= \mathbb{E}[\zeta]^T\mathbb{E}[\widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\widehat{\mathbf{Z}}}^{T,\dagger}_{f^2+\sigma^2}]\left[\mathbf{Z}_{f^2+\sigma^2}\right]_{L\cdot}] + \mathbb{E}[\mathsf{Tr}(\zeta^T\widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\widehat{\mathbf{Z}}}^{T,\dagger}_{f^2+\sigma^2}\zeta)] \\
&= \mathbb{E}[\mathsf{Tr}(\widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\widehat{\mathbf{Z}}}^{T,\dagger}_{f^2+\sigma^2}\eta\eta^T)] \\
&= \mathsf{Tr}\big(\mathbb{E}[\widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\widehat{\mathbf{Z}}}^{T,\dagger}_{f^2+\sigma^2}]\mathbb{E}[\eta\eta^T]\big) \\
&\leq\ C(\Gamma_1,\Gamma_2,\gamma)R^2k,
\end{aligned} \tag{B.102}$$

where $C(\gamma,\Gamma_1,\Gamma_2)$ is a polynomial function of Γ_1,Γ_2,γ only. To see the last inequality, we use various facts. First, by the definition of the HSVT algorithm $\widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}$ has rank at most k. Second, let $\widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2} = \mathbf{USV}^T$ be the singular value decomposition of $\widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}$, we have

$$\begin{aligned}
\widehat{\widehat{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\widehat{\mathbf{Z}}}^{T,\dagger}_{f^2+\sigma^2} &= \mathbf{USV}^T\mathbf{VS}^\dagger\mathbf{U}^T \\
&= \mathbf{U}\tilde{\boldsymbol{I}}\mathbf{U}^T,
\end{aligned} \tag{B.103}$$

That is, $\widehat{\bar{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\bar{\mathbf{Z}}}^{T,\dagger}_{f^2+\sigma^2}$ is a positive semi-definite matrix and $\mathsf{Tr}(\widehat{\bar{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\bar{\mathbf{Z}}}^{T,\dagger}_{f^2+\sigma^2}) \leq k$. The matrix $\mathbb{E}[\zeta\zeta^T]$ is diagonal with all the non-zero entries on diagonal (variance of components of ζ) bounded above by a constant that depends on $\gamma, \Gamma_1, \Gamma_2$. For a positive semi-definite matrix A and positive semi-definite diagonal matrix B, $\mathsf{Tr}(AB) \leq \|B\|_2 \mathsf{Tr}(A)$. This completes the justification of the last step of (B.102).

Now consider the term $\mathbb{E}\big[\| \left[\mathbf{Z}_{f^2+\sigma^2}\right]_{L.} - \widehat{\bar{\mathbf{Z}}}^T_{f^2+\sigma^2}\beta_2^*\|_2^2\big]$.

$$\| \left[\mathbf{Z}_{f^2+\sigma^2}\right]_{L.} - \widehat{\bar{\mathbf{Z}}}^T_{f^2+\sigma^2}\beta_2^*\|_2^2 \leq 2\| \left[\mathbf{Z}_{f^2+\sigma^2}\right]_{L.} - \bar{\mathbf{Z}}^T_{f^2+\sigma^2}\beta_2^*\|_2^2 + 2\|\bar{\mathbf{Z}}^T_{f^2+\sigma^2}\beta_2^* - \widehat{\bar{\mathbf{Z}}}^T_{f^2+\sigma^2}\beta_2^*\|_2^2. \tag{B.104}$$

By equation B.95, we have that $2\| \left[\mathbf{Z}_{f^2+\sigma^2}\right]_{L.} - \bar{\mathbf{Z}}^T_{f^2+\sigma^2}\beta_2^*\|_2^2 = 0$. And,

$$\|\bar{\mathbf{Z}}^T_{f^2+\sigma^2}\beta_2^* - \widehat{\bar{\mathbf{Z}}}^T_{f^2+\sigma^2}\beta_2^*\|_2^2. = \|(\bar{\mathbf{Z}}_{f^2+\sigma^2} - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2})^T\beta_2^*\|_2^2 \;\leq\; \|\bar{\mathbf{Z}}_{f^2+\sigma^2} - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2}\|_{2,\infty}^2\|\beta_2^*\|_1^2, \tag{B.105}$$

where we used the fact that for any $A \in \mathbb{R}^{q\times p}, v \in \mathbb{R}^p$, $\|Av\|_2 \leq \|A^T\|_{2,\infty}\|v\|_1$. Using (B.98), (B.100), (B.102), (B.104), (B.105) we obtain

$$\mathbb{E}\big[\| \left[\mathbf{Z}_{f^2+\sigma^2}\right]_{L.} - \widehat{\bar{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\beta}_2\|_2^2\big] \tag{B.106}$$

$$\leq 2C(\Gamma_1, \Gamma_2, \gamma)R^2k + 2\|\beta_2^*\|_1^2\|\bar{\mathbf{Z}}_{f^2+\sigma^2} - \widehat{\bar{\mathbf{Z}}}_{f^2+\sigma^2}\|_{2,\infty}^2 \tag{B.107}$$

Now $L = \sqrt{\min(N,T)T}$, i.e. $NT/L = \sqrt{NT}$. Using (B.87), and the bound in (B.106); diving by $1/\sqrt{NT}$ on both sides; and noting $k \leq (RG)^2 + (R'G')$, we obtain

$$\mathbb{E}\big[\frac{1}{(NT/L)}\| \left[\mathbf{Z}_{f^2+\sigma^2}\right]_{L.} - \widehat{\bar{\mathbf{Z}}}^T_{f^2+\sigma^2}\widehat{\beta}_2\|_2^2\big] \tag{B.108}$$

$$\leq C(\Gamma_1, \Gamma_2, \Gamma_3, \Gamma_1', \Gamma_2', \gamma, R, R')\left(\frac{G^2+G'}{NT/L} + \|\beta^*\|_2^2\left(\frac{G^2G'\log^2 NT}{L}\right)\right) \tag{B.109}$$

$$\leq C(\Gamma_1, \Gamma_2, \Gamma_3, \Gamma_1', \Gamma_2', \gamma, R, R')\max(1, \|\beta_2^*\|_2^1)\left(\frac{G^2G'\log^2 NT}{L}\right) \tag{B.110}$$

Using (B.110) and (B.94), let $\tilde{C}_1$ and $\tilde{C}_2$ be constants that depend polynomially on

$\Gamma_1, \Gamma_2, \Gamma_3, \Gamma_1', \Gamma_2', \gamma, R, R'$, theb we have

$$\begin{aligned}
&\mathsf{ForErrVar}(N, T, L = \sqrt{NT}) && \text{(B.111)}\\
&\leq \tilde{C}_1 \Big(\max(1, \|\beta_1^*\|_1^2) \left(\frac{G \log NT}{\sqrt{NT}} \right) + \max(1, \|\beta_2^*\|_1^2) \left(\frac{G^2 G'^3 \log^2 NT}{\sqrt{NT}} \right) \Big) \\
&\leq \tilde{C}_2 \Big(\max(1, \|\beta_2^*\|_1^2, \|\beta_1^*\|_1^2) \left(\frac{G^2 G'^3 \log^2 NT}{\sqrt{NT}} \right) \Big) && \text{(B.112)}
\end{aligned}$$

which completes the proof of Theorem 4.2.2.