AN ABSTRACT OF THE THESIS OF

Trevor D. Ruiz for the degree of Doctor of Philosophy inStatistics and Statistics presented on August 26, 2020.

Title: Estimation and Sparse Selection of Conditional Probability Models for Vector Time Series

Abstract approved: _____

Sharmodeep Bhattacharyya Sarah C. Emerson

Diverse scientific fields collect multiple time series data to investigate the dynamical behavior of complex systems: atmospheric and climate science, geophysics, neuroscience, epidemiology, ecology, and environmental science. Identifying patterns of mutual dependence among such data generates valuable knowledge that can be applied either for inferential or forecasting purposes. Vector autoregressive (VAR) processes provide a flexible class of statistical models for multiple time series that are easy to estimate using regression techniques. However, scaling to large data sets and extension to more general processes stretch the framework's capacity: due to the dense parametrization of VAR models, which have one parameter for every possible pairwise relationship between components (*i.e.*, between each univariate time series in a collection), high-dimensional data generate difficulties associated with model selection and parametric regularization; and modeling more general processes requires data trans-

formations that complicate inference, forecasting, and model interpretation. Fields such as epidemiology, neuroscience, and ecology generate high-dimensional time series of count vectors, which incur both sets of challenges at once. Autoregressive conditional probability models — models in which the conditional means of a time series follow an autoregressive structure in the process history — are natural generalizations that preserve ease of estimation and, in conjunction with selection methods in regression, promise to address challenges associated with modeling large multiple time series of count (and other discrete) data. This thesis focuses on developing empirical methodology for sparse selection of nonlinear VAR-type conditional Poisson models. Chapter 1 provides an overview of related existing work. Chapter 2 develops an empirical method for sparse selection in VAR models based on resampling methods. Chapter 3 presents a conditional probability generalization of the VAR model and analyzes the stability properties of Poisson generalized vector autoregressive (GVAR) processes. Chapter 4 combines the work of the preceding two chapters and develops a resampling-based method for sparse estimation of Poisson GVAR models. Finally, Chapter 5 summarizes key findings, challenges, and future work.

Estimation and Sparse Selection of Conditional Probability Models for Vector Time Series

by

Trevor D. Ruiz

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Presented August 26, 2020
Commencement June 2021

Doctor of Philosophy thesis of Trevor D. Ruiz presented on August 26, 2020.

APPROVED:

Co-Major Professor, representing Statistics

Co-Major Professor, representing Statistics

Chair of the Department of Statistics

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Trevor D. Ruiz, Author

ACKNOWLEDGEMENTS

My advisors, Sharmodeep Bhattacharyya and Sarah Emerson, made significant intellectual and material contributions to the work in this thesis and provided an extraordinary amount of personal support during my time as a graduate student. Sharmo introduced me to network estimation problems involving spatiotemporal data and proposed a number of research projects during my first year as a graduate student. He has since skillfully guided my work in this thesis and in several applied projects with interest, flexibility, and vision. Sarah deftly reviewed and refined large volumes of raw research material as I was beginning my thesis in earnest and provided invaluable guidance in developing and clarifying many of the key ideas that now appear in these chapters. She also identified and resolved a number of technical flaws in computer codes and in early chapter drafts. In addition to greatly improving the quality of my work, Sharmo and Sarah have provided encouragment and advice that have helped me navigate this project and graduate school more generally.

I am also grateful to several collaborators. David Gent has provided rich collaborative opportunities throughout my time as a student and supported me on research assistantships one term each year for the past several years and during multiple summers. While our applied work does not appear in the thesis, it has been equally rewarding, and has strongly influenced the direction of my methodological research and facilitated much intellectual growth. I have been fortunate to work with him. Kris Bouchard was an early mentor and collaborator and contributed considerably to the material in Chapter 2, along with several students in his lab, including Mahesh

Balasubramanian, Max Dougherty, and Pratik Sachdeva. Kris generously devoted many hours to reviewing and discussing raw material, and provided editorial revisions on a prior version of the chapter that appears in the 2nd L4DC conference proceedings. Mahesh collaborated closely on the computer science side of the latter work and a complementary paper, developing C++ codes and conducting scaling runtime experiments for the methodology we developed. I often consulted Max and Pratik on datasets, codes, methodology, and a variety of other subjects. Their contributions provided crucial support for my initial research efforts and helped set the course for work that appears in later chapters.

I thank my committee members, Lan Xue, Duo Jiang, and Lori Cramer for reviewing the thesis and providing comments that improved the manuscript. Cathy Lundmark also provided helpful editorial revisions.

Countless others have contributed in one way or another to the work that appears here: all my teachers, peers, and students; and my friends and family. Thank you.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF APPENDIX FIGURES

LIST OF APPENDIX FIGURES (Continued)

Chapter 1: Introduction

Technological advances in the sophistication of scientific devices, personal devices, and information management have produced an abundance of multivariate time series data, underscoring the need for statistical methods for modeling mutual dependence among large collections of time series of diverse data types. Stochastic process models for vector time series provide the basis for such methods. Modern data are both high-dimensional and diverse in type, which presents two major challenges. First, high dimensionality generates both theoretical and computational difficulties for estimation of stochastic process models and inference. Second, diverse data types present difficulties for the distributional assumptions of common (Gaussian) models. To date, there is a limited amount of work that targets both challenges at once: specification of vector time series models with flexible distributional assumptions and high-dimensional estimation of such models. However, many authors have considered high-dimensional estimation in the context of the Gaussian vector autoregressive model, and there is also a rich literature on generalized univariate models for discrete time series. This thesis presents work at the intersection of these two areas that focuses on developing methodology for vector time series of counts.

1.1 Background

This section contextualizes the contributions of the thesis with a brief review of two threads of existing work. The first thread, covered in Section 1.1.1, pertains to sparse estimation of vector autoregressive models, and is motivated by applications of vector autoregressive models to high-dimensional data. The second, discussed in Section 1.1.2, pertains to generalizations of univariate AR models to count data, focusing on so-called 'observation-driven' conditional models. Considered together, this body of literature provides the relevant background for generalizations of time series models to count vectors and sparse estimation methods.

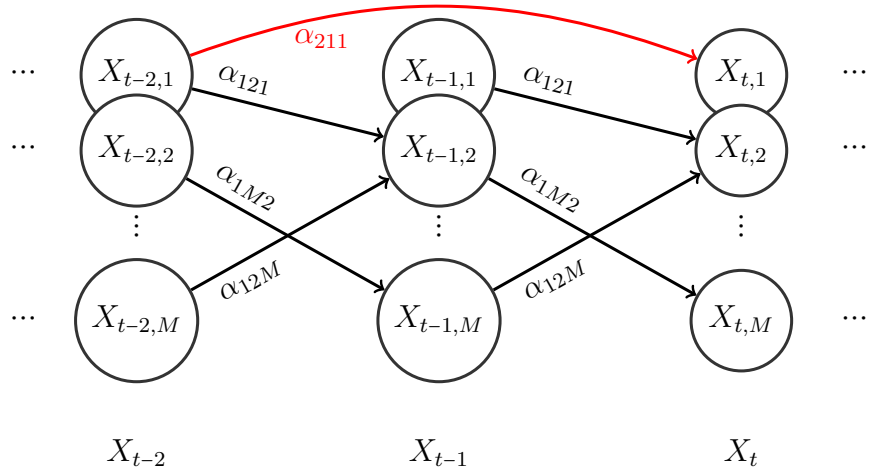
1.1.1 Vector autoregression

The vector autoregressive (VAR) model is a multivariate extension of the univariate autoregressive stochastic process model, $X_t = \sum_{d=1}^{D} \alpha_d X_{t-d} + \epsilon_t$. In univariate autoregression, the fixed parameters $\alpha_1, \ldots, \alpha_D$ relate present values of the random variable X_t to its past values $X_{t-1}, \ldots, X_{t-D}$ up to lag order D and the relationship is modulated by a random noise process $\{\epsilon_t\}_{t\in\mathbb{Z}}$. The multivariate generalization posits the same structure for a vector process. The VAR model is

$$X_t = \sum_{d=1}^{D} A_d X_{t-d} + \epsilon_t$$

where X_t and ϵ_t are random vectors and $A_1, \ldots, A_D$ are fixed parameter matrices. In the VAR model, the diagonal elements of $A_1, \ldots, A_D$ relate present values of each vec-

tor component to their own past values, and the off-diagonal elements relate present values of each vector component to the past values of other components. Figure 1.1 provides a graphical illustration for a sparse set of example parameters. These relationships together with the covariance of the noise process characterize the dependence structure of the vector process $\{X_t\}_{t\in\mathbb{Z}}$.



$$A_1 = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ \alpha_{121} & 0 & \cdots & \alpha_{12M} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \alpha_{1M2} & \cdots & 0 \end{pmatrix} \qquad A_2 = \begin{pmatrix} \alpha_{211} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}$$

Figure 1.1: Illustration of mean structure in an example VAR model with $D = 2$ and four nonzero autoregressive parameters. Above, a graphical representation with nodes representing vector components and edges representing autoregressive parameters. Below, the corresponding parameter matrices A_1 and A_2.

Historical background. The VAR model initially gained popularity in econometrics. Granger (1969) proposed a framework for statistical inference of causal rela-

tionships between time series based on forecasting accuracy that leveraged the VAR model. Subsequently, Sims (1980) advocated the VAR model more broadly as providing a flexible time-domain framework for multiple time series analysis with minimal *a priori* assumptions about the data-generating system. Based largely on the works of these two authors, the VAR model became a popular tool in econometrics for forecasting and macroeconomic structural analysis (Lütkepohl, 2005; Tsay, 2005).

Following the works of Granger and Sims, the VAR model found both broader methodological applications and broader subject-area applications. Based on Granger's notion of causality, the VAR model figured centrally in the development of graphical models for time series in the statistics and machine learning communities (Arnold et al., 2007; Bach and Jordan, 2004; Dahlhaus, 2000; Dahlhaus and Eichler, 2003; Eichler, 1999; Qiu et al., 2016; Songsiri et al., 2010). In parallel, technological advances expanded the range of contexts in which multiple time series data were collected in the sciences, providing novel application domains for these methods and temporal and spatial statistical models in general (Atluri et al., 2018). For example, enhanced recording devices, sensors, and remote sensing technologies enabled simultaneous time-course measurements to be taken at multiple locations for a variety of physical systems, including the brain (Bassett and Sporns, 2017; Buzsáki et al., 2012; Pesaran et al., 2018) and the earth's geophysical and atmospheric climate (Kalnay et al., 1996; Karpatne et al., 2013, 2018).

High-dimensional VAR models. The above developments coupled with advances in computing led to modern use of the VAR model at scales far beyond its ini-

tial applications. A well-known challenge inherent in the VAR model is its dense parametrization — the number of parameters grows quadratically with the number of component time series (*i.e.,* with the vector dimension) — which motivated several relatively early works on subset and order selection methods tailored to VAR models (*e.g.*, Chen et al. (1996); Penm and Terrell (1982)). However, increases in application scales created the need for more computationally efficient alternatives that work well in high-dimensional settings or with large data sets (Hsu et al., 2008; Valdés-Sosa et al., 2005).

Hsu et al. (2008) provided empirical support and asymptotic theory for VAR estimation with a LASSO (Least Absolute Shrinkage and Selection Operator (Tibshirani, 1996)) penalty, which induces sparsity and can be computed efficiently using a variety of fast algorithms (Boyd et al., 2011; Efron et al., 2004; Friedman et al., 2007). More recently Basu et al. (2015) established finite-sample estimation and selection error bounds for the method. However, the method is known to overfit in certain contexts (*e.g.*, Arnold et al. (2007)), and as a result there has been a proliferation of alternative penalization strategies that attain various theoretical or empirical improvements on estimation and selection efficiency relative to the LASSO (Davis et al., 2016; Han et al., 2015; Shojaie and Michailidis, 2010; Song and Bickel, 2011). These methods and other field-specific variations now appear in a wide range of application domains, including climate science (Lozano et al., 2009a,b; Triacca et al., 2013), econometrics (Barigozzi and Brownlees, 2019; Barigozzi and Hallin, 2017; Fan et al., 2011), meteorology (Cavalcante et al., 2017; Dowell and Pinson, 2015), and neuroscience (Wang et al., 2016).

1.1.2 Time series models for count vectors

Many fields generate time series of count vectors: ecological studies collect species counts over time (Ives et al., 2003)); epidemiological studies examine infection case counts observed at multiple locations (Bracher and Held, 2019)); in neuroscience, counts of cellular activations ('neural spikes') are recorded simultaneously across multiple cells (Brown et al., 2004; Buzsáki et al., 2012); and in the social sciences, historical data is generated on events of distinct types (Brandt and Sandler, 2012). In these contexts it is of interest to model the dependence structure between individual time series in order to understand, *e.g.*, ecological interactions, cellular functional connectivity patterns, dynamic event feedback in certain types of historical processes, and epidemic cascade networks. While data transformations might justify the use of VAR models as approximations in the analysis of vector time series in a wide range of contexts, for discrete data in particular such transformations can be artificial and the resulting approach to statistical modeling can lack interpretability. Thus, generalizations of the VAR model that relax distributional assumptions are desirable. There is comparatively much more literature on this subject in the univariate case than in the vector case; a brief overview of univariate time series models for counts is given below before turning to multivariate extensions.

Univariate models. A thread of early work on non-Gaussian time series models focuses on exponential family generalizations of univariate process models; these approaches characterize a stochastic process $\{X_t\}_{t\in\mathbb{Z}}$ with history $\mathcal{H}_t$ by specifying a parametric model for $X_t|\mathcal{H}_t$ in the exponential family. Cox et al. (1981) introduce a

broad distinction between 'parameter-driven' models in which the parameter of interest is a function of a latent random process, and 'observation-driven' models in which the parameter of interest depends on the past observations.

Observation-driven models specified through conditional means later became known as 'generalized autoregressive models' (Fahrmeir and Tutz, 1994; Wong et al., 1986), and bear an analogous relationship to the autoregressive process as generalized linear models do to the normal linear model. A generalized autoregressive model is of the form

$$\mu_t = \mathbb{E}_f(X_t|\mathcal{H}_t) = g\left(z_t(\mathcal{H}_t)'\beta\right) \tag{1.1}$$

where f is an exponential family density, g is a known (inverse link) function, and $z_t(\mathcal{H}_t)$ is a vector of covariates that depends on the process history (*e.g.*, $z_t = (X_{t-1}\ X_{t-2})'$ or $z_t = ((X_{t-1}-\mu_{t-1})\ (X_{t-2}-\mu_{t-2}))'$). These models have the advantage that maximum likelihood estiamtes (MLEs) can be computed using standard methods for generalized linear models and large-sample inference is substantially similar to the GLM case (Fahrmeir and Kaufmann, 1985), only based on conditional observed Fisher information in place of unconditional expected Fisher information (Fahrmeir, 1988, 1987). However, in most contexts the regularity conditions under which asymptotic theory for the MLE is derived include the assumption that the process specified by Equation (1.1) is ergodic (see, *e.g.*, Wong et al. (1986)), which of course depends on g and z_t and is not guaranteed in general.

The Poisson case provides a natural and easy-to-implement framework for modeling count data,[1] but ergodicity can require sometimes restrictive parameter con-

[1]Poisson generalized autoregressive models also have some interesting connections with point

straints (Fahrmeir and Tutz, 1994; Wong et al., 1986; Zeger and Qaqish, 1988). For instance, if $z_t(\mathcal{H}_t)$ are exactly lagged observations, the log-linear model

$$\mu_t = \exp\{\beta_0 + \beta_1 X_{t-1}\}$$

is only ergodic when $\beta_1 < 0$ (Fokianos and Tjøstheim, 2011; Zeger and Qaqish, 1988), and therefore the model excludes positive serial dependence. Alternatively, ergodicity conditions under the identity link exclude negative serial dependence (Fokianos et al., 2009).

This issue has motivated a number of variations on the transformation $z_t(\cdot)$ and the inverse link function g (*e.g.*, Aknouche et al. (2018); Fokianos and Tjøstheim (2012)). Several good reviews are available, including Davis et al. (1999); Fahrmeir and Tutz (1994); Fokianos (2012); Kedem and Fokianos (2005). Specific variations tend to balance parameter flexibility with model interpretability, and many achieve good results. For example, Davis et al. (2003) consider a variation in which μ_t is log-linear in scaled deviations of past observations from their means to allow positive and negative dependence, and the resulting process is ergodic under fairly general conditions. More generally, Neumann et al. (2011) establishes a contraction condition on μ_t that guarantees ergodicity in the Poisson case, and the result is extended to the exponential family in Davis and Liu (2012), providing a simple condition to check for any parametrization.

process models. See, for instance, Brown (2005); Jacobs and Lewis (1977); Lawrance and Lewis (1977); Truccolo et al. (2005).

Multivariate models. There is a relative paucity of literature in statistics and probability on vector extensions of the univariate models discussed above, though a few early examples can be found (McKenzie, 1988; Ord et al., 1993) and the subject is beginning to receive renewed attention (Hall et al., 2016a,b, 2018; Mark et al., 2017). Additionally, multivariate extensions appear in a number of specific subject areas. In finance, for example, Heinen et al. (2003) propose a VARMA-type model (Vector Autoregressive Moving Average) for a vector process $\{X_t\}_{t\in\mathbb{Z}}$ wherein, conditional on the process history $\mathcal{H}_t$, the components $X_{t,m}|\mathcal{H}_t$ are independent Poisson random variables with conditional means given by

$$\mu_t = \mathbb{E}(X_t|\mathcal{H}_t) = \nu + \sum_{d=1}^{D} A_d X_{t-d} + \sum_{d=1}^{D} B_d \mu_{t-d}$$

where ν is a fixed intercept, $A_1, \ldots, A_D$ are autoregressive parameter matrices, and $B_1, \ldots, B_D$ are moving average parameter matrices. This model is extended to the double Poisson distribution in Heinen and Rengifo (2007) to accommodate overdispersion. A similar model is considered in political science methodology by Brandt and Sandler (2012), who propose a purely autoregressive conditional mean structure with covariate and latent effects given by

$$\mu_t = AX_{t-1} + \exp\{Z_t\beta + b_t\}$$

where Z_t are covariates and b_t are latent effects. Finally, Pillow et al. (2008) use an autoregressive model with covariate effects to analyze neural spike counts; in their

model, the mean structure is

$$\mu_t = \exp\left\{\nu + Z_t\beta + \sum_{d=1}^{D} A_d h_d(X_{t-d})\right\}$$

where h_d are known functions and the remaining parameters are as above.

Little attention in the above works is given to the choice of link function or covariate transformations and their implications for parameter constraints. When the linear link is used a sufficient condition for process stationarity can be derived (Heinen et al., 2003) but the parameters must be positive, so such models exclude negative serial dependence. Under the log link, both positive and negative serial dependence are possible, but it is unclear what parameter constraints might be required to ensure that the process is well-behaved in some sense (stable, stationary, or ergodic). Mark et al. (2017) acknowledge that process stability is a problem in this case, but they address the issue by thresholding the effects of X_{t-1} on X_t.

1.2 Contributions and organization of the thesis

Existing work leaves ample room for contributions focusing on vector extensions of generalized autoregressive models and high-dimensional estimation and inference of these and other multivariate time series models. First, sparse estimation methods for VAR models continue to be developed. Second, few generalizations of VAR models are available and the probabilistic behavior of existing generalizations is not well understood. Third, very few estimation methods have been developed for generalized

vector autoregression.

The chapters of this thesis contribute to addressing these gaps by jointly developing an empirical method for sparse estimation of VAR and Poisson generalized VAR models utilizing a log link and providing initial probabilistic analysis of the latter stochastic process model. Chapter 2 presents a resampling-based empirical method for sparse estimation of VAR models. The method achieves good selection performance in simulation and in principle is of comparable computational efficiency to the LASSO technique. Chapter 3 analyzes the stability properties of Poisson generalized vector autoregressive models with a log link. A graphical constraint that ensures the existence of moments when the process has both positive and negative serial dependence is derived, and some additional results helpful in understanding process stability are presented and discussed. Finally, Chapter 4 extends the methodology of the Chapter 2 and applies it to sparse estimation of Poisson generalized vector autoregressive models discussed in Chapter 3. A simulation framework is constructed based on the results of the Chapter 3 and an empirical sparse estimation method is developed based on the results of the Chapter 2 and evaluated using the simulation framework. Chapter 5 concludes the thesis with a discussion of impact and future work.

Chapter 2: Sparse estimation of high dimensional VAR models

2.1 Introduction

This chapter presents a novel empirical method for sparse selection in vector autoregressive (VAR) models.[1] The method extends an existing framework for sparse selection with low-bias estimation in regression (Bouchard et al., 2017) to the vector time series setting under modifications. Most existing methods for sparse selection in VAR models are based on L_1 penalization (Davis et al., 2016; Han et al., 2015; Hsu et al., 2008; Shojaie and Michailidis, 2010; Song and Bickel, 2011; Songsiri et al., 2010), and by far the most widely used approach is to perform both selection and estimation simultaneously using the LASSO (Basu, 2014; Basu et al., 2015). However, it has been frequently observed that the LASSO tends to overfit in general (Bühlmann and Van De Geer, 2011; Meinshausen and Bühlmann, 2010; Meinshausen et al., 2006) and especially in the VAR context (Arnold et al., 2007; Davis et al., 2016; Valdés-Sosa et al., 2005), and if the method is also used for estimation, estimates can be severely biased (as shown in this chapter).

Purpose and contributions. The aim of the method developed in this chapter is to improve both selection and estimation accuracy relative to the LASSO. Empirical support is provided in the form of a simulation study, computational scaling exper-

[1] A version of this chapter appears as Ruiz et al. (2020).

iments (described in Appendix C), and a data application. Together, these results explore the statistical and computational properties of the method and illustrate its use. The simulation study compares the selection, estimation, and forecasting accuracy of models estimated by the novel method with models estimated by the LASSO. The scaling experiments identify implementation bottlenecks that limit runtime for large datasets, and incidentally demonstrate successful estimation of very large VAR models. The data analysis illustrates the utility of the novel method in a causal analysis of financial time series.

The main contributions of the chapter are:

1. novel methodology for sparse VAR estimation;
2. basic and distributed-computing implementations of the method;
3. empirical experiments demonstrating improved selection accuracy and lower estimation bias relative to LASSO-penalized least squares estimates;
4. illustrative example of causal analysis of financial time series data.

Organization. The chapter is organized as follows. Section 2.2 reviews background material to contextualize the main contributions: Section 2.2.1 introduces the VAR model class and several of its key properties; and Section 2.2.2 introduces the LASSO estimator and discusses a technique for computing the estimator. Section 2.3 presents the novel methodology: Section 2.3.1 presents the estimation algorithm with a line-by-line explanation of key steps; and Section 2.3.2 discusses bootstrap methods, which are used in the estimation algorithm, in the context of time series. Section 2.4 presents a

collection of empirical results: Section 2.4.1 summarizes a simulation study exploring the statistical properties of the estimation method; and Section 2.4.2 presents an example application of the method to causal analysis of financial time series. Section 2.5 closes the chapter with a discussion of the main findings, challenges, and possible extensions of this work.

2.2 Background

This section reviews key material leveraged in the development and application of methodology for sparse estimation of vector autoregressive models in this chapter. Section 2.2.1 introduces the VAR model class with a formal definition of vector autoregressive processes, a review of process stability conditions and stationarity properties, and a review of forecasting and causality for VAR processes. Section 2.2.2 introduces existing sparse estimation methodology with a review of least squares estimation of VAR processes, the addition of a sparsity constraint to the estimator, and a 'vectorization' strategy for computation using LASSO regression algorithms.

2.2.1 Vector autoregressive processes

The exposition here closely follows Lütkepohl (2005), and the reader is referred to this work for a comprehensive review of VAR models and time-domain multiple time series models more generally.

VAR(D) process definition. Vector autoregressive processes of order D (VAR(D))

are the family of stochastic processes $\{X_t : \Omega \to \mathbb{R}^M\}_{t\in\mathbb{Z}}$ characterized by:

$$X_t = \nu + \sum_{d=1}^{D} A_d X_{t-d} + \epsilon_t \ , \qquad \text{for all } t \in \mathbb{Z} \tag{2.1}$$

It is assumed that the error process ϵ_t is a white noise process with finite variance: $\mathbb{E}\epsilon_t = 0$ for every t; $\mathbb{E}\epsilon_t\epsilon_t' = \Sigma$ for every t; Σ is finite and positive definite; and $\mathbb{E}\epsilon_t\epsilon_s' = 0$ for every $t \neq s$.

VAR(1) process stability. For any $VAR(1)$ process, recursively applying Equation (2.1) $s+1$ times beginning at any timepoint $t \in \mathbb{Z}$, one can write:

$$X_t = (I + A_1 + A_1^2 + \cdots + A_1^s)\nu + A_1^{s+1} X_{t-s-1} + \sum_{j=0}^{s} A_1^j \epsilon_{t-j}$$

This recursion holds for time point t and number of steps s into the process history. Since $s, t \in \mathbb{Z}$, in order for X_t to be well-defined, one needs that the limits $\lim_{s\to\infty} \sum_{j=0}^{s} A_1^j$ and $\lim_{s\to\infty} \sum_{j=1}^{s} A_1^j \epsilon_{t-j}$ exist and are finite: that the former converges to a finite matrix, and that the latter converges to a non-degenerate random vector. A sufficient condition can be expressed as a constraint on the eigenvalues of A_1. If the eigenvalues of A_1 are all in the interval $(-1, 1)$, then:

(i) $\{A_1^i\}_{i=0}^{\infty}$ is absolutely summable and the sequence converges quickly to 0;

(ii) $\sum_{i=0}^{j} A_1^j \to (I - A_1)^{-1}$ as $j \to \infty$; and

(iii) the sum $\sum_{j=1}^{s} A_1^j \epsilon_{t-j}$ converges in the L^2 sense.

For proof details, see Lütkepohl (2005), Appendix C. The eigenvalues of A_1 are all in

$(-1, 1)$ just in case:

$$\det(I - A_1 z) \neq 0, \quad \text{for all } z \in \mathbb{C} \text{ with } |z| \leq 1$$

This is referred to as a 'stability' condition, and a VAR(1) process is said to be stable if the condition holds.

Stable VAR(1) processes are also stationary. Letting as $s \to \infty$ in the recursion yields the process representation:

$$X_t = (I - A_1)^{-1}\nu + \sum_{j=0}^{\infty} A_1^j \epsilon_{t-j}$$

From this representation it can be seen that the process mean and autocovariances are time-invariant, since independently of t:

$$\begin{aligned}
\mathbb{E}X_t &= (I - A_1)^{-1}\nu \\
\mathbb{E}(X_t - \mathbb{E}X_t)(X_{t-s} - \mathbb{E}X_{t-s})' &= \mathbb{E}\left(\left(\sum_{i=0}^{\infty} A_1^i \epsilon_{t-i}\right)\left(\sum_{j=0}^{\infty} A_1^j \epsilon_{t-s-j}\right)'\right) \\
&= \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} A_1^i \mathbb{E}\epsilon_{t-i}\epsilon_{t-s-j}'(A_1')^j \\
&= \sum_{j=0}^{\infty} A_1^{s+j}\Sigma(A_1')^j
\end{aligned}$$

The autocovariance function has a more tractable representation, though the above derivation is more straightforward and suffices to establish stationarity of the process.

VAR(D) process stability. Any VAR(D) process can be rewritten as a VAR(1)

process, and so it is said that a VAR(D) process is stable just in case its VAR(1) representation is stable. Let X_t be a VAR(D) process; its VAR(1) representation is defined in terms of Y_t, $\tilde{\nu}$, $\tilde{A}$, and ψ_t, given blockwise by:

$$Y_t = \begin{bmatrix} X_t \\ X_{t-1} \\ \vdots \\ X_{t-D+1} \end{bmatrix}, \quad \tilde{\nu} = \begin{bmatrix} \nu \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \tilde{A} = \begin{bmatrix} A_1 & A_2 & \cdots & A_{D-1} & A_D \\ I_M & 0 & \cdots & 0 & 0 \\ 0 & I_M & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & I_M & 0 \end{bmatrix}, \quad \psi_t = \begin{bmatrix} \epsilon_t \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Then by the process definition (Equation (2.1)) one has:

$$Y_t = \tilde{\nu} + \tilde{A}Y_{t-1} + \psi_t \,, \quad \text{for all } t \in \mathbb{Z}$$

Since it is assumed that ϵ_t is a white noise process, it follows immediately that ψ_t is a white noise process.

Therefore Y_t is a VAR(1) process. The process is stable if $\det\left(I - \tilde{A}z\right) \neq 0$ for all $|z| \leq 1$. Since Y_t and X_t are exactly the same process, X_t is stable under the same condition. The stability condition on $\tilde{A}$ rewritten in terms of $A_1, \ldots, A_D$ gives that X_t is stable if:

$$\det\left(I - \sum_{d=1}^{D} A_d z^d\right) \neq 0, \quad \text{for all } z \in \mathbb{C} \text{ with } |z| \leq 1$$

Forecasting. An h-step forecast is a prediction of X_{t+h} based on the history of the process up to time t. The conditional expectation $\mathbb{E}(X_{t+h}|X_s = x_s, s \leq t)$ (wherein

the lowercase x_s denotes a fixed realization of the random vector X_s) satisfies certain optimality properties as an h-step forecast within the VAR(D) model class. Denote this predictor by:

$$\hat{X}_t(h) \stackrel{\text{def}}{=} \mathbb{E}(X_{t+h}|X_s = x_s, s \le t) = \nu + \sum_{d=1}^{D} A_d \mathbb{E}(X_{t+h-d}|X_s = x_s, s \le t)$$

When $h = 1$, one has that $\mathbb{E}(X_{t+h-d}|X_s = x_s, s \le t) = x_{t+h-d}$ for every d in the summand because $X_{t+h-d} \in \{X_s, s \le t\}$ for $d \ge 1$. In this case, the forecast is:

$$\hat{X}_t(1) = \nu + \sum_{d=1}^{D} A_d x_{t+1-d}$$

Otherwise, for $h > 1$, the forecasts $\hat{X}_t(h)$ can be computed sequentially in h starting from $h = t + 1$. With the convention that $\hat{X}_t(h) = x_h$ for any $h \le t$, one has that:

$$\hat{X}_t(h) = \nu + \sum_{d=1}^{D} A_d \hat{X}_t(h-d)$$

The forecast $\hat{X}_t(h)$ is unbiased since $\mathbb{E}\hat{X}_t(h) = \mathbb{E}(\mathbb{E}X_{t+h} \mid X_s = x_s, s \le t) = \mathbb{E}X_{t+h}$, and optimal in the MSE sense. That is, for any h-step forecast $\hat{X}_t^*(h)$, the difference in MSE between $\hat{X}_t^*(h)$ and $\hat{X}_t(h)$ is positive semi-definite:

$$\mathbb{E}(X_{t+h} - \hat{X}_t^*(h))(X_{t+h} - \hat{X}_t^*(h))' - \mathbb{E}(X_{t+h} - \hat{X}_t(h))(X_{t+h} - \hat{X}_t(h))' \ge 0$$

Granger causality. The forecast $\hat{X}_t(h)$ can be used to characterize a notion of causality originally proposed in Granger (1969): one process is causal for another if

conditioning on the former reduces forecasting error for the latter. Here this notion of causality is articulated in the context of VAR processes.

Consider the partition of X_t into subprocesses X_t^A and X_t^B. Denote the h-step conditional expectation forecast of the subprocess X_t^A given the full process up to time t by:

$$\left(\hat{X}_t(h)\right)^A \stackrel{\text{def}}{=} \mathbb{E}(X_{t+h}^A | X_s = x_s, s < t)$$

Then denote the h-step conditional expectation forecast of the subprocess X_t^A given only its own history up to time t by:

$$\hat{X}_t(h)^A \stackrel{\text{def}}{=} \mathbb{E}(X_{t+h}^A | X_s^A = x_s^A, s < t)$$

$\hat{X}_t(h)^A$ and $(\hat{X}_t(h))^A$ distinguish 'partly conditional' forecasts (forecasts conditioned only on the subprocess) from 'fully conditional' forecasts (forecasts conditioned on the full process).

The subprocess X_t^A causes the subprocess X_t^B (in the Granger sense) just in case for at least one h the difference in MSE between the partly conditional forecast $\hat{X}_t(h)^A$ and the fully conditional forecast $(\hat{X}_t(h))^A$ is positive semi-definite:

$$\mathbb{E}\left(X_{t+h}^A - \hat{X}_t^A(h)\right)\left(X_{t+h}^A - \hat{X}_t^A(h)\right)' - \mathbb{E}\left(X_{t+h}^A - (\hat{X}_t(h))^A\right)\left(X_{t+h}^A - (\hat{X}_t(h))^A\right)' \geq 0$$

It can be shown that for any pair of univariate subprocesses X_{ti} and X_{tj}, X_{tj} causes X_{ti} just in case $(A_d)_{ij} \neq 0$ for at least one d. As a result, it is possible to determine causal relationships among the components $X_{t1}, \ldots, X_{tM}$ of a VAR(D) process X_t by

direct examination of the sparsity pattern of $\sum_d A_d$:

$$X_{tj} \text{ causes } X_{ti} \iff \left(\sum_d A_d\right)_{ij} \neq 0$$

2.2.2 Sparse estimation via LASSO

Sparse estimation of VAR processes is particularly useful if the application involves assessing causality among components, since, following the discussion of Granger causality above, the sparsity patterns of $A_1, \ldots, A_D$ directly indicate the presence and absence of pairwise causal relationships. The three most common estimation techniques for VAR processes are moment estimation (solving the Yule-Walker equations), maximum likelihood (under the assumption that the error process is Gaussian), and least squares. Here an overview of the least squares technique with a sparsity constraint is given.

Least squares estimation. Let $\{x_t \in \mathbb{R}^M\}_{t=0}^T$ denote an observed time series of length $T+1$. The VAR(D) model for $\{x_t\}_{t=0}^T$ can be expressed in the form of a multivariate multiple regression $Y = UB + E$ where:

$$Y = \begin{bmatrix} x_T' \\ x_{T-1}' \\ \vdots \\ x_D' \end{bmatrix} \quad U = \begin{bmatrix} 1 & x_{T-1}' & \cdots & x_{T-D}' \\ 1 & x_{T-2}' & \cdots & x_{T-D-1}' \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{D-1}' & \cdots & x_0' \end{bmatrix} \quad B = \begin{bmatrix} \nu' \\ A_1' \\ \vdots \\ A_D' \end{bmatrix} \quad E = \begin{bmatrix} \epsilon_T' \\ \epsilon_{T-1}' \\ \vdots \\ \epsilon_D' \end{bmatrix} \tag{2.2}$$

The ordinary least squares estimator of B is:

$$\hat{B} = \text{argmin}_B \, \|Y - UB\|_F^2 = (U'U)^{-1}U'Y \tag{2.3}$$

Then, an estimate of Σ is given by $\hat{\Sigma} = \frac{1}{T-1}(Y - U\hat{B})(Y - U\hat{B})'$. If the data are centered and the intercept ν is omitted, this estimation technique is identical to maximum likelihood estimation under the assumption that the noise process is Gaussian (for details, see Lütkepohl (2005)).

LASSO penalty. The least squares estimator $\hat{B}$ can be modified to impose a sparsity constraint on $A_1, \ldots, A_D$ by including a LASSO penalty distributed over the elements a_{dij} of each matrix A_d:

$$\tilde{B} = \text{argmin}_B \left\{ \|Y - UB\|_F^2 + \lambda \sum_{d=1}^{D} \sum_{i,j} |a_{dij}| \right\} \tag{2.4}$$

The regularization hyperparameter λ controls the sparsity of the optimum; larger values of λ result in more sparse estimates, and smaller values result in denser estimates.

Computation via vectorization. Computations for the sparse least squares estimate $\tilde{B}$ can be obtained by rewriting the problem as a univariate multiple regression and applying algorithms for LASSO regression. Let vec($\cdot$) denote the column-stacking operator. Applying this term to the response Y and predictor UB produces an equivalent representation of the regression equation:

$$Y = UB \iff \text{vec}Y = \text{vec}(UB)$$

Noting that $\text{vec}(UB) = (U \otimes I_M)\text{vec}B$,[2] define the quantities:

$$\mathcal{Y} \stackrel{\text{def}}{=} \text{vec}Y \ , \quad \mathcal{X} \stackrel{\text{def}}{=} U \otimes I_M \ , \quad \text{and } \beta \stackrel{\text{def}}{=} \text{vec}B$$

Then Equation (2.2.2) becomes:

$$Y = UB \iff \mathcal{Y} = \mathcal{X}\beta$$

This rearrangement 'vectorizes' the multivariate regression equation $Y = UB$ since it re-expresses the regression equation in terms of the vector $\mathcal{Y}$.

The estimator $\tilde{B}$ can be recovered from a LASSO regression estimate of β. Since for any matrix X, $\|X\|_F = \|\text{vec}X\|_2$, and the $\text{vec}(\,\cdot\,)$ operator is linear, it follows that:

$$\|Y - UB\|_F = \|\mathcal{Y} - \mathcal{X}\beta\|_2$$

Now consider the estimator:

$$\tilde{\beta} = \text{argmin}_\beta \left\{ \|\mathcal{Y} - \mathcal{X}\beta\|_2^2 + \lambda \sum_{d=1}^{D} \sum_{i,j} |a_{dij}| \right\}$$

The estimators $\tilde{\beta}$ and $\tilde{B}$ minimize identical loss functions, and by definition, $\beta = \text{vec}B$. Therefore, the same relationship holds between the estimates:

$$\tilde{\beta} = \text{vec}\tilde{B}$$

[2]The notation $\otimes$ denotes the Kronecker product.

This renders a strategy for computing $\tilde{B}$: recovery via the LASSO estimate $\tilde{\beta}$. The strategy is useful because the latter estimate can be computed by direct application of LASSO regression algorithms. Such algorithms are widely available and can be implemented efficiently for the VAR case by leveraging the sparsity pattern of $\mathcal{X}$. Appendix A gives details on implementing pathwise coordinate descent with such modifications and an appropriate penalty distribution strategy to accommodate the intercept terms interspersed throughout the parameter vector β.

2.3 Methods

This section presents novel methodology for sparse estimation of vector autoregressive models by extending an existing algorithmic framework, the Union of Intersections (UoI) (Bouchard et al., 2017). The framework separates estimation of sparse parameter support sets from selection of support sets and subsequent estimation of parameters in a resampling-intensive scheme. This idea is developed in the context of VAR estimation based on the least squares and LASSO estimators $\hat{B}$ (Equation (2.3)) and $\tilde{B}$ (Equation (2.4)) in conjunction with time series resampling methods to produce a 'UoI-VAR' estimator.

The UoI-VAR method is presented in subsection 2.3.1. Following this, subsection 2.3.2 indicates challenges associated with bootstrapping for time series and presents the bootstrap method used in implementing the UoI-VAR method.

2.3.1 UoI-VAR estimation method

The UoI-VAR method separates estimation of VAR parameters $B = (\nu\ A_1\ \cdots\ A_D)'$ into an intersection step utilizing the LASSO estimator $\tilde{B}$ (Equation (2.4)) to identify candidate support sets and a union step utilizing the least squares estimator $\hat{B}$ (Equation (2.3)) to combine estimates on the candidate supports. These steps are given in detail separately as Algorithms 2.1 and 2.2.

Algorithm 2.1 Intersection step for UoI-VAR.

Require:
data $\{x_t \in \mathbb{R}^M\}_{t=0}^T$
regularization path $\lambda \in \mathbb{R}^K$
number of bootstrap samples B_1
for $b = 1$ to B_1 **do**
draw bootstrap sample $\{x_t^*\}_{t=0}^T$
construct Y^*, U^* based on $\{x_t^*\}_{t=0}^T$ according to Equation (2.2)
for $k = 1$ to K **do**
estimate $\tilde{B}_{bk} \leftarrow \text{argmin}_B \left\{ \|Y^* - U^* B\|_F^2 + \lambda_k \sum_{d=1}^D \sum_{i,j} |a_{dij}| \right\}$
compute support $S_{bk} \leftarrow \left\{ (i,j) : \tilde{b}_{bkij} \neq 0 \right\}$
end for
end for
Ensure: aggregated supports $S_k \leftarrow \bigcap_{b=1}^{B_1} S_{bk}\ , \quad k = 1, \ldots, K$

Intersection step. Algorithm 2.1 begins by drawing B_1 bootstrap samples. LASSO estimates are computed on each bootstrap sample for a range of regularization hyper-parameters λ_k. That is, if $\{x_t^*\}_{t=0}^T$ denotes the bth bootstrap sample, the procedure forms the multivariate regression response Y^* and design matrix U^* according to

Equation (2.2) and computes the sparse least squares estimator (Equation (2.4)):

$$\tilde{B}_{bk} = \text{argmin}_B \left\{ \|Y^* - U^* B\|_F^2 + \lambda_k \sum_{d=1}^{D} \sum_{i,j} |a_{dij}| \right\} \quad k = 1, \ldots, K$$

Since the magnitude of the regularization hyperparameter λ_k controls the sparsity level of the estimate, each such estimate $\tilde{B}_{bk}$ is associated with a support set and the sparsity of those support sets varies with the index k. Each such support set is denoted here as the collection of indices (i, j) that correspond to row and column positions in the matrix $\tilde{B}_{bk}$ with nonzero values:

$$S_{bk} = \left\{ (i, j) : \tilde{b}_{bkij} \neq 0 \right\}$$

A final collection of candidate support sets $S_1, \ldots, S_K$ is formed by performing intersection operations on the support sets S_{bk} across bootstrap samples (index b) for each regularization hyperparameter (index k):

$$S_k = \bigcap_{b=1}^{B_1} S_{bk} \ , \quad k = 1, \ldots, K$$

Each resulting support set S_k comprises the parameter positions that are selected under the sparsity constraint corresponding to λ_k with high probability under resampling of the data. This procedure is an extension of the methodology of Bach (2008) to the time series context, and bears close connection to stability methods in high-dimensional regression (Meinshausen and Bühlmann, 2010).

Algorithm 2.2 Union step for UoI-VAR.

Require:
data $\{x_t \in \mathbb{R}^M\}_{t=0}^T$
candidate support sets $S_1, \ldots, S_K$ (from intersection step)
number of bootstrap samples B_2
threshold parameter γ
for $b = 1$ to B_2 **do**
draw bootstrap samples $\left\{x_t^{*(1)}\right\}_{t=0}^T$, $\left\{x_t^{*(2)}\right\}_{t=0}^T$
construct Y^*, U^* based on $\left\{x_t^{*(1)}\right\}_{t=0}^T$ according to Equation (2.2)
for $k = 1$ to K **do**
fix subspace $\mathcal{B}_k \leftarrow \{B \in \mathbb{R}^{M \times D(D+1)} : \{(i,j) : b_{ij} \neq 0\} = S_k\}$
compute constrained OLS estimate $\hat{B}_{bk} \leftarrow \operatorname{argmin}_{B \in \mathcal{B}_k} \{\|Y^* - U^* B\|_F\}$
compute forecast error $f_{bk} \leftarrow \sum_t \|x_t^{*(2)} - \hat{X}_{t-1}^{*(2)}(1;\ \hat{B}_{bk})\|_2$
end for
find minimum error $k^{*b} \leftarrow \operatorname{argmin}_k f_{bk}$
select support $S^{*b} \leftarrow S_{k^{*b}}$
end for
determine sparsest selected supports $Q \leftarrow \{b : |S^{*b}| \leq \gamma \cdot \text{quantile}(|S^{*1}|, \ldots, |S^{*B_2}|)\}$
Ensure: final estimate $\hat{B} \leftarrow \sum_{b \in Q} \tilde{B}_{k^{*b}}$

Union step. Algorithm 2.2 begins by drawing B_2 pairs of bootstrap samples. Now, each of the candidate support sets $S_1, \ldots, S_K$ from the intersection step defines a parameter subspace $\mathcal{B}_k = \{B \in \mathbb{R}^{M \times D(D+1)} : \{(i,j) : b_{ij} \neq 0\} = S_k\}$ in which the positions of B not included in S_k are set to zero. The algorithm computes least squares estimates on each subspace $\mathcal{B}_1, \ldots, \mathcal{B}_K$ from one of the bootstrap samples in each of the B_2 pairs. That is, if $\left\{x_t^{*(1)}\right\}_{t=0}^{T}$ denotes the first bootstrap sample in the bth pair, the procedure computes:

$$\hat{B}_{bk} = \text{argmin}_{B \in \mathcal{B}_k} \left\{ \|Y^* - U^* B\|_F \right\} , \quad b = 1, \ldots, B_2 \text{ and } k = 1, \ldots, K$$

The solution can be computed easily leveraging the vectorization transformations discussed above.[3]

Next, for each of the B_2 pairs, one-step forecast errors are computed on the remaining bootstrap sample in the pair. That is, if $\left\{x_t^{*(2)}\right\}_{t=0}^{T}$ denotes the second bootstrap sample in the bth pair, and $\hat{X}_t^{*(2)}(h;\ \hat{B}_{bk})$ denotes the h-step conditional expectation forecast of $X_{t+h}^{*(2)}$ from time t based on the estimate $\hat{B}_{bk}$, then for each estimate $\hat{B}_{b1}, \ldots, \hat{B}_{bK}$ the following error quantity is computed:

$$f_{bk} = \sum_{t=1}^{T} \|x_t^{*(2)} - \hat{X}_{t-1}^{*(2)}(1;\ \hat{B}_{bk})\|_2$$

[3]The vectorized problem is $\text{argmin}_{B \in \mathcal{B}_k} \|\mathcal{Y}^* - \mathcal{X}^* \beta\|_2^2$, where $\mathcal{Y}^* = \text{vec} Y^*$, $\mathcal{X}^* = U^* \otimes I_M$, and $\beta = \text{vec} B$. The least squares solution constrained to $\mathcal{B}_k$ is found by dropping columns of $\mathcal{X}^*$ in correspondence to the sparsity pattern induced in β by S_k and computing the usual solution: if $\chi_k = \text{vec}\{\mathbb{1}\{(i,j) \in S_k\}$ denotes the vectorized adjacency matrix generated by S_k, then $|\chi_k|$ matches the dimension of the column space of $\mathcal{X}^*$; letting $\mathcal{X}_k^*$ denote the columns of $\mathcal{X}^*$ for which the entry in the corresponding position in χ_k is 1, the constrained solution is $\hat{B}_{bk} = \left(\mathcal{X}_k^{*\prime} \mathcal{X}_k^*\right)^{-1} \mathcal{X}_k^{*\prime} \mathcal{Y}^*$.

The index k that minimizes f_{bk} is then used to find an optimal support set for each pair of bootstrap samples:

$$S^{*b} = S_{k^{*b}} \text{ where } k^{*b} = \text{argmin}_k f_{bk} , \quad b = 1, \ldots, B_2$$

Each support set in the resulting collection $S^{*1}, \ldots, S^{*B_2}$ is exactly one of the candidate supports $S_1, \ldots, S_K$ from the intersection step: S^{*b} is whichever candidate support yielded the constrained least squares estimates that gave the best forecasts for the bth pair of bootstrap samples.

Lastly, the sparsest approximately $100 \times \gamma\%$ support sets are identified:

$$Q = \{b : |S^{*b}| \leq \gamma \cdot \text{quantile}(|S^{*1}|, \ldots, |S^{*B_2}|)\}$$

The final estimate is the average of the estimates $\hat{B}_{bk}$ corresponding to the selected supports S^{*b} for each $b \in Q$:

$$\hat{B} = \sum_{b \in Q} \tilde{B}_{k^{*b}}$$

This final step performs a union operation on the selected support sets and returns an estimate on the resulting union.

2.3.2 A bootstrap method for time series

Modified bootstrap procedures suitable for time series are required to implement the UoI-VAR method. 'Suitable' means that such procedures must largely preserve the

form of dependence implicit in the model being estimated. The classical bootstrap does not meet this criterion; a block bootstrap procedure is adopted in its place.

Classical bootstrap. The classical bootstrap is random sampling with replacement from the observations. Applied to time series observations, random sampling with replacement fails to preserve any dependence between lagged observations, which results in a loss of information about the structure of interest. For example, suppose $\{X_t\}_{t\in\mathbb{Z}}$ is a VAR(1) process, and consider that one obtains by a bootstrap sample the process rearrangement:

$$\{\dots, X_1^*, X_2^*, \dots\} = \{\dots, X_{14}, X_3, \dots\}$$

According to the data-generating process:

$$\mathbb{E}(X_t|X_{t-1} = x_{t-1}) = \nu + Ax_{t-1}\ , \quad t \in \mathbb{Z}$$

However, for the rearrangement, one has that $\mathbb{E}(X_2^*|X_1^* = x_1^*) = \mathbb{E}(X_3|X_{14} = x_{14}) = \mathbb{E}(X_3)$ and $\mathbb{E}X_t = (I-A)^{-1}\nu$ for every t. As a result, whenever $x_1^* \neq A^{-1}((I-A)^{-1}\nu-\nu)$:

$$\mathbb{E}(X_2^*|X_1^* = x_1^*) \neq \nu + Ax_1^*$$

So the bootstrap sample does not in general follow the same data generating process.

Moving block bootstrap. The moving block bootstrap (Kunsch et al., 1989; Lahiri et al., 1999; Liu and Singh, 1992) resamples blockwise with replacement from obser-

vations, and in doing so largely preserves serial dependence in data (Liu and Singh, 1992). Consider a finite-length stochastic process $\{X_t\}_{t=0}^{T}$. Denote a fixed block length by L and a random starting index by $I \sim \text{uniform}\{0, \ldots, T-L+1\}$. A randomly chosen block from $\{X_t\}_{t=0}^{T}$ of length L is:

$$\{X_I, \ldots, X_{I+L-1}\}$$

The moving block bootstrap is simply a concatenation of B such blocks. Let:

$$I_1, \ldots, I_B \overset{\text{iid}}{\sim} \text{uniform}\{0, \ldots, T-L+1\}$$

Then a moving block bootstrap sample of $\{X_t\}_{t=0}^{T}$ with block length L and B blocks is:

$$\{X_s^*\}_{s=1}^{S} = \{X_{I_1}, \ldots, X_{I_1+L-1}, \ldots, X_{I_B}, \ldots, X_{I_B+L-1}\}\,, \quad \text{where } S = BL$$

The moving block bootstrap does not perfectly preserve VAR data-generating processes — it preserves the data-generating process within each block but not across blocks. That is, if $\{X_t\}_{t=0}^{T}$ is a VAR(1) process and $\{X_s^*\}_{s=1}^{S}$ is a moving block bootstrap sample, then in general between blocks one has:

$$\mathbb{E}(X_s^* | X_{s-1}^* = x_{s-1}^*) \neq \nu + Ax_{s-1}\,, \quad \text{whenver } s = bL \text{ for } b = 1, \ldots, B$$

However, the data-generating process is preserved within blocks, since in the same

example:

$$\mathbb{E}(X_s^*|X_{s-1}^* = x_{s-1}^*) = \nu + Ax_{s-1}\,, \quad \text{whenever } s \neq bL \text{ for } b = 1, \ldots, B$$

In general, the method introduces BD 'discontinuity' points for VAR(D) processes — points at which the data-generating process fails to describe the bootstrapped process.

For VAR(D) estimation, the block length L and number of blocks B should be chosen so that $L \gg B$ and $L \gg D$. Maintaining $L \gg B$ limits the number of discontinuity points that violate the dependence structure assumed in the VAR(D) model, and maintaining $L \gg D$ ensures that the blocks preserve a sufficient order of dependence from the original data to estimate a VAR(D) model using the MBB sample. However, estimation quality may be sensitive to small changes in the choice of L and B. For further discussion of block length selection, see Bühlmann and Künsch (1999); reviews of alternative bootstrap methods for time series are given in Kreiss and Lahiri (2012); Lahiri et al. (1999).

2.4 Results

This section presents results from a simulation study and a data application that together provide empirical support for the UoI-VAR method. The simulation study, presented in Section 2.4.1, explores the selection, estimation, and forecasting performance of the UoI-VAR estimator relative to the LASSO estimator. Section 2.4.2

summarizes a data analysis that illustrates the utility of the UoI-VAR method in causal network analysis.

Method implementations. Two distinct implementations of the method were developed: a MATLAB implementation for the simulation study and data analysis; and a distributed implementation in C++ for conducting scaling experiments (Balasubramanian et al., 2020). While code testing confirmed matching outputs, the two implementations utilize different methods of computing LASSO and OLS estimators on bootstrap samples in the intersection and union steps. The results presented here rely on the MATLAB implementation, which executes a pathwise coordinate descent algorithm based on Friedman et al. (2010) and modified for the VAR setting per Appendix A, setting $\lambda = 0$ for computing least squares estimates. The C++ implementation and scaling experiments are discussed in Appendix C.

2.4.1 Simulation study

The performance of the UoI-VAR estimator on synthetic data was assessed relative to the LASSO estimator for a range of data dimensionalities, with the parametric sparsity of the data-generating processes fixed in proportion to the process dimension. The performance comparisons were made in terms of selection accuracy, model fit, and estimation bias for each method.

Simulation study design. The simulation study consisted in generating a set of VAR(1) parameters for each combination of the process dimensions $M = 5, 10, 20, 40, 80$

and time series lengths $T = 50, 100, 200$; then, 50 datasets were simulated at the appropriate length from each set of parameters, comprising 750 synthetic datasets in total. UoI-VAR and LASSO estimators were computed on each datset.

Parameter generation. VAR(1) process parameters were generated as follows: M nonzero matrix parameters were drawn at random from a distribution increasing exponentially away from zero in either direction, and allocated to random positions in an $M \times M$ parameter matrix A_1. The matrix A_1 was then rescaled by a factor of $(|\Lambda_{\max}(A_1)| + 0.1)^{-1}$, where $\Lambda_{\max}(A_1)$ denotes the maximum eigenvalue of A_1, in order to guarantee process stability. The parameters generated by this process exhibit $1 - 1/M\%$ sparsity. The intercept was fixed at $\nu = 0$, and the error process covariance was fixed at $\Sigma = 0.5 I_M$.

Hyperparameter settings. Both the UoI-VAR and LASSO methods utilized the same regularization path λ, which was adjusted slightly for each distinct combination of M, T to ensure that the LASSO selection spanned the full range of sparsity levels from a null model to a saturated model. For the LASSO, the regularization strength that minimized average forecasting errors over five-fold cross-validation was used for the final estimate. In detail, this cross validation procedure is shown as Algorithm 2.3. For UoI-VAR, the additional hyperparameters $B_1 = 10$, $B_2 = 50$, and $\gamma = 0.3$ were fixed for all combinations of M, T.

Simulation results. Figure 2.1 summarizes the results of the simulation study, reporting empirical distributions of selection accuracy, estimation error, and forecast

Algorithm 2.3 Cross-validation for regularization parameter λ selection.

block length $L \leftarrow \text{floor}(T/5)$
for $j = 1$ to 5 **do**
  starting index $i_0 \leftarrow (j-1)L$
  block $I \leftarrow \{i_0, \ldots, i_0 + L - 1\}$
  training data $\{x_s^*\}_{s=0}^{S} \leftarrow \{x_t\}_{t \notin I}$, where $S = T - L$
  construct Y^*, U^* based on $\{x_s^*\}_{s=0}^{S}$ according to Equation (2.2)
  estimate $\tilde{B}_{jk} \leftarrow \text{argmin}_B \left\{ \|Y^* - U^* B\|_F^2 + \lambda_k \sum_{d=1}^{D} \sum_{i,j} |a_{dij}| \right\}$, for all $\lambda_k \in \lambda$
  forecast error $f_{jk} \leftarrow \sum_{t \in I} \|x_t - \hat{X}_{t-1}(1;\, \tilde{B}_{jk})\|_2$, for all $\lambda_k \in \lambda$
end for
set optimal $\lambda^* \leftarrow \lambda_{k^*}$ where $k^* = \text{argmin}_k \sum_j f_{jk}$

error for each estimator under the combinations of dimension M and time series length T considered in the study. To describe these quantities precisely, denote a parameter estimate of $B = (\nu\ A)'$ by $\hat{B} = (\hat{\nu}\ \hat{A})'$, and denote an estimate of the support set of A, $S = \{(i,j) : a_{ij} \neq 0\}$, by $\hat{S} = \{(i,j) : \hat{a}_{ij} \neq 0\}$. The selection accuracy metric is defined in terms of false positives $FP = |\hat{S} \setminus S|$, the number of positions in the parameter estimates that are in fact zero but estimated as nonzero, and false negatives $FN = |S \setminus \hat{S}|$, the number of positions that are in fact nonzero but estimated as zero. The selection accuracy metric is defined as $1 - \frac{FN+FP}{M+FP}$ (for a more general definition, replace M by $TP = |S|$). The metric equals 1 if selection is perfect ($FN = 0$ and $FP = 0$) and 0 if selection is maximally erroneous ($FP = M^2 - M$ and $FN = M$). Estimation error is computed on the estimated support and defined as $\sum_{i,j \in \hat{S}} \|a_{ij} - \hat{a}_{ij}\|_2^2$. Finally, one-step forecast error is defined as $\frac{1}{T-1} \sum_t \|x_{t+1} - \hat{X}_{t-1}(1; \hat{B})\|_2^2$.

Across all settings, UoI-VAR exhibits improved selection accuracy relative to LASSO; these behaviors are driven predominantly by false positive rates (depicted in Supplementary Figure S.1). The main limitation of UoI-VAR is an increased false

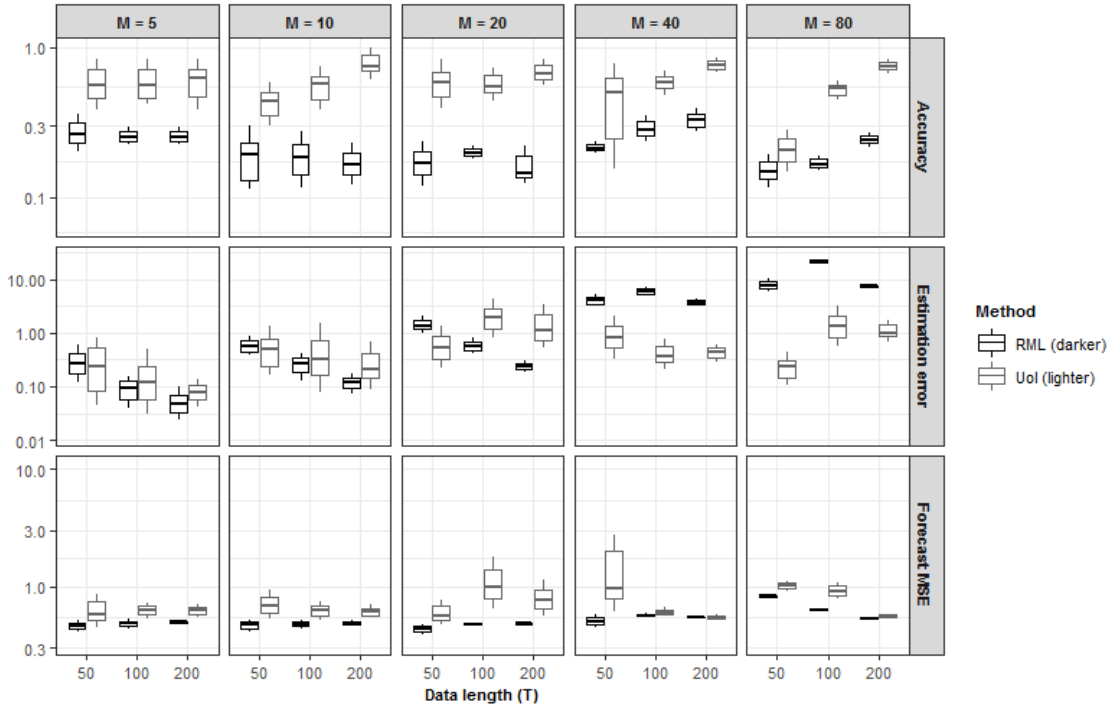


Figure 2.1: Selection, estimation, and forecasting behavior for UoI-VAR and LASSO estimators observed in the simulation study. Panel rows distinguish selection, estimation, and forecasting metrics; panel columns distinguish dimensions M; and in each panel boxplots for the row metric for each estimator are plotted against time series length T on the horizontal axis.

negative rate relative to LASSO when less data are available (shorter T settings). However, this problem diminishes rapidly as time series length T increases, and as a result, for larger dimensions M, the selection performance of the UoI-VAR method improves much faster than LASSO as T increases. Furthermore, as depicted in the second row of the figure, UoI-VAR achieves dramatically lower estimation errors in large-M settings. Finally, it appears that these improvements come at the cost of a slight decrease in forecast accuracy.

2.4.2 Application

To illustrate an application, a VAR(1) model was used to identify putative causal connections between weekly closes of 50 randomly chosen publicly traded companies listed on the S&P 500 index in 2013-2014, and the causal analysis was repeated with each of the UoI-VAR and LASSO methods compared in the simulation study (Section 2.4.1).

S&P data. The dataset analyzed in the application is shown in Figure 2.2, in which each panel depicts weekly closes of share prices for one of 50 randomly chosen companies listed on the S&P 500 index during the years 2013-2014, which saw a steadily climbing index with no major economic disturbances.[4] To obtain an approximately stationary process, first-order differences were calculated from the raw series (shown

[4]Long-term historical data on daily closes for all 500 companies listed in the index are publicly available. A subset of companies was chosen for this illustration to simplify visualization of results, and a subset of years was chosen so that stationarity assumptions would be plausible. The time series was thinned from daily closes to weekly closes to obtain a regular time step, since trading days are Monday through Friday.

in Supplementary Figure S.3); then, VAR(1) model parameters were estimated from these differences using the UoI-VAR method and LASSO.

Causal analysis. Figure 2.3 shows two causal networks among the 50 companies in the dataset: one network inferred from the sparsity pattern of a LASSO estimate of the VAR(1) model for the first-order differences; and another network inferred from the UoI-VAR estimate. Following the discussion of Granger causality in Section 2.2.1, these networks are direct visualizations of the zero or nonzero classification of estimated entries in the parameter matrix A. That is, each network is the graph $G = (V, E)$ where the node set $V = \{1, \ldots, 50\}$ represents the 50 companies in the data and the edge set $E = \{(i, j) \in V \times V : \hat{A}_{ji} \neq 0\}$ is the estimated support set of A.

The UoI-VAR estimate identified 44 pairwise causal relationships that mostly describe influences on Google's share price from other companies. Relatively few causal relationships connect other companies. By comparison, the LASSO estimate identifies 146 causal relationships among a larger collection of companies. The latter contains the network identified using the UoI-VAR estimate, but the network is much less prominent due to the density of the graph. The network based on the UoI-VAR estimate is comparatively easier to examine directly, and highlights a single prominent causal structure.

Forecasts are comparable between the two estimates. One-step forecast root mean square error (RMSE) averaged over all companies using the LASSO estimate is 8.3993; for the UoI-VAR estimate, 8.4525 (an increase of 0.6% relative to LASSO). The scale of share prices varies widely among the companies, and the forecast errors for both

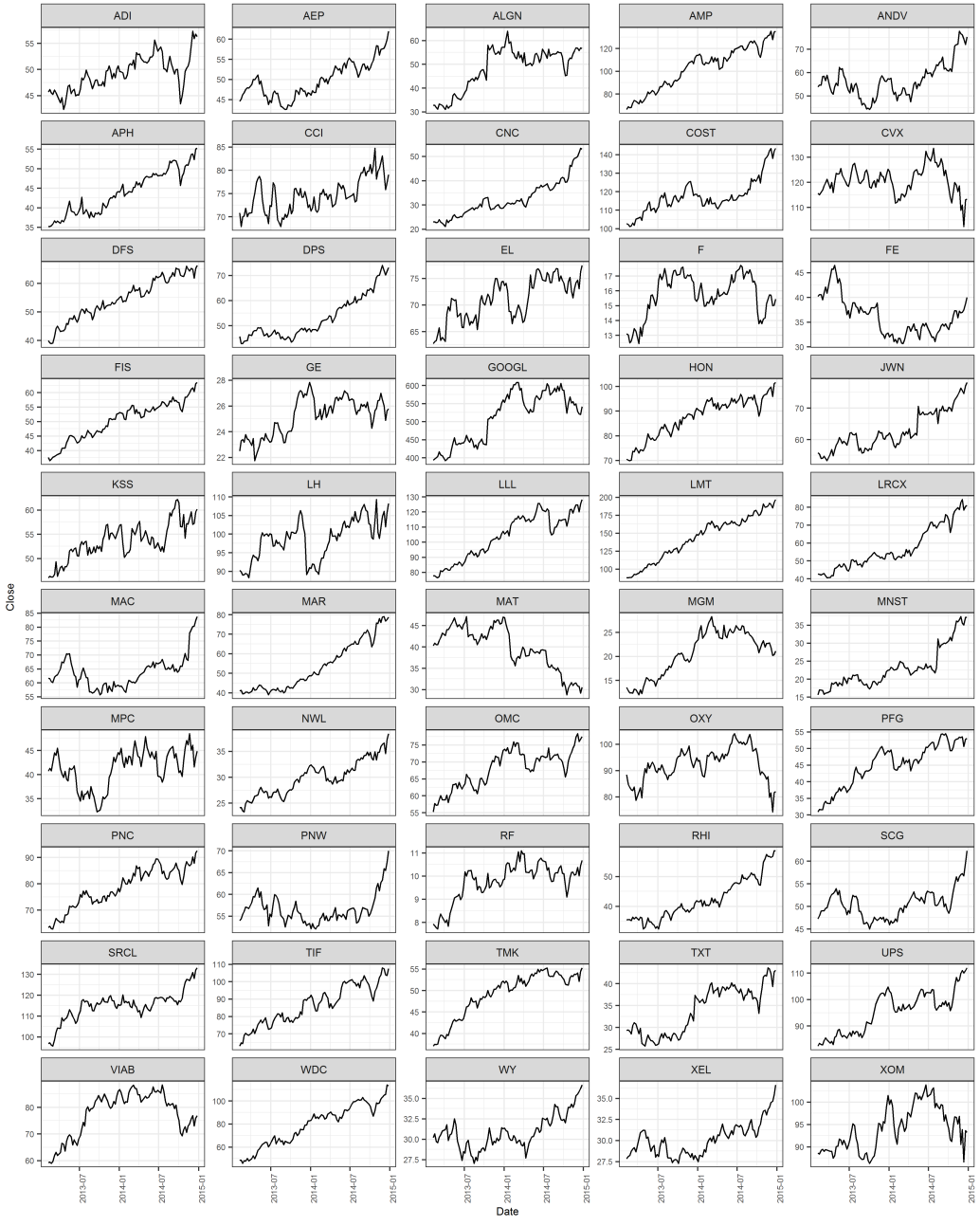


Figure 2.2: S&P dataset analyzed in the example causal analysis application.

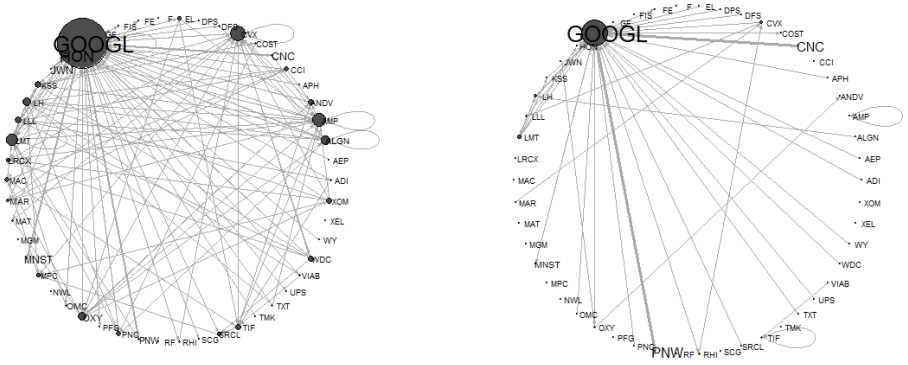


Figure 2.3: Causal networks inferred from S&P 500 data using LASSO estimates of a VAR(1) model for first differences (left) and UoI-VAR estimates of the same model (right). In each network, one node is shown per company, and an edge between two nodes indicates an inferred causal relationship between the corresponding companies. Node and label size are proportional to degree centrality in the network.

methods are approximately ordered in correspondence with average share prices. The median per-company forecast RMSE is 4.7807 using LASSO, and 4.3329 using UoI-VAR (a decrease of 9% relative to LASSO). Thus, forecasts viewed in aggregate reflect slightly worse performance using the UoI-VAR estimate; however, on a company-by-company basis, the forecasts from UoI-VAR are slightly more accurate for most companies.

2.5 Discussion

The UoI-VAR method presented in this chapter addresses the problem of sparse selection and estimation in large vector autoregressive models. Existing methods impose LASSO-type sparsity constraints on least squares or maximum likelihood estimators and provide little practical guidance on how to adjust the strength of the constraint. This chapter presents an empirical estimation method, UoI-VAR, and several experiments exploring its statistical and computational properties.

Findings. The simulation study and data analysis suggest that the UoI-VAR method is most advantageous when model parsimony is a priority. The simulation study indicates that UoI-VAR exhibits much improved selection accuracy, mainly through control of erroneous selection (rather than erroneous omission). This improvement comes at an apparent slight cost in forecasting. A similar tradeoff appears in the data analysis, where the causal network inferred from UoI-VAR parameter estimates is simpler than the network inferred from LASSO estimates but forecasts are slightly worse. However, both the simulation study and data analysis report one-step forecast

errors at each time point in the dataset given the preceding history but after estimating parameters based on the entire dataset; genuine out-of-sample forecasts were not assessed. Therefore, the apparent increase in forecast error for UoI-VAR estimates relative to LASSO estimates may simply reflect that the LASSO tends to overfit.

In developing the methodology, it was found that a thresholding modification to the UoI framework was needed to maintain sparse selection in the method. The threshold γ, which limits the density of the final estimate, is introduced in the union step. It was found that a direct extension of the union step (*i.e.*, setting $\gamma = 1$) tends to produce overly dense estimates due to a small number of dense selected supports S^{*b}. Overall, the selected supports S^{*b} tend to have mutually comparable sparsity levels, yet outliers occur reliably often within the variation exhibited by the bootstrap samples. This behavior does not appear in other UoI methods. The most straightforward strategy for addressing this problem is to draw many supports S^{*b} and utilize only those below a manually chosen sparsity quantile. This strategy avoids imposing an explicit limit on sparsity and captures the overall selection behavior of S^{*b} while eliminating extreme cases. It is possible that the underlying issue is a too-frequent failure of the bootstrap method to capture dependence in the data.

Challenges. The VAR estimation problem exhibits inherent challenges due to the large number of parameters. The VAR estimation problem is superficially similar to estimation of regression parameters due to the use of analogous techniques and algorithms, yet the VAR problem scales quite differently. Estimation of a VAR(1) model from $T \times M$ data (length $\times$ vector dimension) using regression techniques translates

after vectorization to a pseudo-regression problem involving a $(T-1)M \times M(M+1)$ design matrix; so for instance, if $M = 50$ and $T = 1001$, which is a relatively small dataset for certain application domains, the pseudo-design matrix is 50000×2550. As indicated above, this particular data transformation is a poor choice for scalable implementations. However, even if it is avoided, modeling $T \times M$ data with a VAR(D) model requires an $M(DM+1)$ parameter space that scales quadratically in M and linearly in D. This creates a challenge mainly in the sense that it limits the ability to conduct controlled experiments for large data dimensions. The work presented here involves estimation of large models, up to $M = 1000$ in the scaling experiments. However, it was found to be infeasible given available resources to scale simulation experiments much beyond $M \approx 150$ due to the need to repeat estimation on multiple datasets to obtain informative results.

A second challenge encountered was the absence of any obvious efficient methods for tuning hyperparameters for the UoI-VAR method. While the method requires only an appropriate choice for the *range* of regularization hyperparameters λ_k, it eliminates the need to select a single regularization hyperparameter at the cost of introducing B_1, B_2, γ, L: the numbers of bootstrap repetitions, the threshold parameter, and the bootstrap block length. While the method is relatively less sensitive to small changes in these parameters than LASSO is to the choice of λ, and in some sense this is a helpful tradeoff, there are no clear strategies for fixing these parameters. In the work presented here, they were each tuned manually by trial and error experimentation across a range of settings.

Future work. Future work could address either of the challenges identified above: developing scalable implementations of 'core' methods for estimating VAR parameters; or developing systematic methods for tuning hyperparameters in the UoI-VAR method. In addition, several other extensions are possible. In terms of methodology, developing theory for the estimator and investigating the effect of bootstrap methods would add valuable information to the results presented here. Extension of the method to other time series models would broaden the methodology. Finally, applications to a wider range of scientific data could produce valuable domain knowledge.

Chapter 3: Poisson generalized vector autoregression

3.1 Introduction

Chapter 2 developed a sparse estimation method for VAR processes, motivated by applications involving estimation of (Granger) causal networks from continuous multivariate time series. In many application areas it is of interest to estimate networks from multivariate count time series rather than from continuous data (Bracher and Held, 2019; Brandt and Sandler, 2012; Pillow et al., 2008). The VAR process provides possible description of such data only under transformations (*e.g.*, log and square-root or power transformations); one natural alternative is to fit generalized linear models with autoregressive-type predictors. This approach maintains the methodological simplicity of using regression methods for model estimation and avoids the loss of model interpretability associated with data transformations (Cox et al., 1981).

This chapter studies the underlying stochastic process implied by modeling multivariate count time series using a Poisson GLM with a log link function and a vector autoregressive predictor for the conditional distributions of the component time series. The resulting process is approached through the lens of generalizing the Gaussian VAR process: examining the conditional distributions of component univariate time series given the process history and positing an analogous conditional mean structure under different distributional assumptions. The Poisson log-linear case is referred to

here as a Poisson generalized vector autoregressive (GVAR) process. This view situates the resulting process in a much broader class of multivariate stochastic process models.

Contributions. After framing the generalization and introducing the Poisson GVAR process, the chapter presents a number of insights into its probabilistic behavior, focusing on conditions under which first and second moments exist. It is shown that the use of a non-linear link function introduces a need for parametric constraints to avoid unbounded means and variances, a possibility which is well-acknowledged for univariate versions of the model (Davis and Liu, 2012; Fahrmeir and Tutz, 1994; Zeger and Qaqish, 1988) but appears to have been largely overlooked for multivariate extensions. As discussed in Chapter 1, one of the challenges in developing univariate models is finding parametrizations that allow both positive and negative serial dependence and specify stable processes under fairly general conditions. This chapter proposes a graphical constraint under which the Poisson GVAR(1) vector process is stable and allows positive and negative mutual serial dependence among vector components. However, it is further shown that the existence of finite moments under this constraint does not necessarily guarantee that moments are computationally tractable, which suggests that the distinction between stability and instability is perhaps less relevant than that between satisfying and failing specific bounds on process moments (for an interesting general discussion of the distinction between finiteness and computatibilty, see Knuth (1976)). A sufficient condition for attaining prespecified moment bounds is derived. Finally, stability under relaxations of the graphical

constraint is considered. After developing these main results, the chapter turns briefly to likelihood estimation and shows that there are parameter regimes that satisfy both these theoretical and practical constraints that are nonetheless difficult to estimate due to issues related to parameter identifiability.

In short, the chapter explores the probabilistic properties of Poisson GVAR processes and identifies challenges in parametrization and estimation with a focus on order-1 processes. Its main contributions are:

(i) generalization of Poisson log-linear autoregression to the multivariate setting and identification of problematic parameter regimes;

(ii) derivation of a graphical constraint on Poisson GVAR(1) process parameters that guarantees finite process moments or process 'stability';

(iii) derivation of conditions on parameter magnitudes that ensure bounded moments of a specific magnitude for processes satisfying the graphical constraint;

(iv) identification of practical issues in likelihood estimation related to parameter identifiability.

The chapter is organized as follows. Section 3.2 presents the generalization of VAR processes and discusses the log-linear Poisson GVAR process as a special case: the process definition and properties are given in Section 3.2.1; and the need for parametric constraints is introduced in Section 3.2.2. Section 3.3 presents the main results of the chapter on process stability: a sufficient stability condition is established in Section 3.3.1; constraints on parameter magnitudes that ensure specified bounds

on process moments are developed in Section 3.3.2; and a discussion of conditions under which parameters that do not satisfy the graphical constraint produce unstable processes is given in Section 3.3.3. Lastly, Section 3.4 discusses challenges associated with estimation: the maximum likelihood estimator is defined and computation is discussed in Section 3.4.1; the concept of 'pseudo-unidentifiability' is introduced and discussed in Section 3.4.2. Section 3.5 closes the chapter with a discussion of the main findings, challenges, and further work.

3.2 Poisson generalized vector autoregressive processes

This section generalizes the vector autoregressive process (introduced in Chapter 2, Section 2.2.1) to the exponential family and considers Poisson log-linear autoregression as a special case. The basic properties — conditional moments, likelihood and parameter identifiability, independence conditions for subprocesses, and parameter interpretation — of the Poisson case are discussed.

3.2.1 Process definition and properties

Poisson generalized vector autoregression. Generalized vector autoregressive (GVAR) processes are a wide class of stochastic processes $\{X_t : \Omega \to \mathbb{R}^M\}_{t\in\mathbb{Z}}$ characterized by conditionally independent marginal probability distributions of the vector components $X_{t,m}$ given the process history. The class can be viewed as comprising generalizations of the $VAR(D)$ process with uncorrelated Gaussian errors

$\epsilon_t \sim N(0, \Sigma)$ (where Σ is diagonal):

$$X_t = \nu + \sum_{d=1}^{D} A_d X_{t-d} + \epsilon_t \ , \qquad \text{for all } t \in \mathbb{Z}$$

Conditional on the process history, one has that for every time point t:

$$(X_t \mid X_s = x_s, s < t) = \nu + \sum_{d=1}^{D} A_d x_{t-d} + \epsilon_t$$

It is immediate that the distribution of $(X_t \mid X_s, s < t)$ is

$$(X_t \mid X_s = x_s, s < t) \sim N\left(\nu + \sum_{d=1}^{D} A_d x_{t-d}, \Sigma\right)$$

Since Σ is diagonal, the components $X_{t,m}$ are conditionally independent:

$$(X_{t,m} \mid X_s = x_s, s < t) \overset{indep.}{\sim} N\left(\nu_m + \sum_{d=1}^{D} a'_{dm} x_{t-d}, \Sigma_{mm}\right) \ , \quad m = 1, \ldots, M$$

The following definition generalizes this last conditional probability formulation.

Definition 3.1 *(GVAR Process). Let p be a probability density with parameter θ. $\{X_t\}_{t\in\mathbb{Z}}$ is a GVAR process if for every $t \in \mathbb{Z}$:*

$$(X_{t,m} \mid X_s = x_s, s < t) \overset{indep.}{\sim} p\left(\cdot\,; \theta_{t,m} = \theta_m(x_{t-1}, x_{t-2}, \ldots)\right), \ m = 1, \ldots, M$$

For any finite-time process, an initial distribution ν_0 together with the process definition completely characterizes the probabilistic behavior of a GVAR process, since

the joint density of any realization $\{x_t\}_{t=0}^{T}$ of length $T+1$ is then given by

$$\mathbb{P}(X_0 = x_0, \ldots, X_T = x_T) = \nu_0(x_0) \prod_{t=1}^{T} \prod_{m=1}^{M} p(x_{t,m}; \theta_{t,m})$$

Further conditions are required on $\theta_m(\,\cdot\,)$ that depend on the density p to guarantee that the class is well-defined for particular families of distributions. In particular, $\theta_{t,m}$ must map from the support of p to the parameter space for every t and m.

Now, a GVAR process is a Poisson GVAR(D) process if p is the Poisson density and $\theta_{t,m}$ is linear in D lags. To state this exactly, consider the Poisson(λ) density defined on the nonnegative integers $\mathbb{Z}_+$ and parametrized in terms of the canonical parameter $\theta = \log\lambda$:

$$p(x;\theta) = \frac{\exp\{x\theta - e^{\theta}\}}{x!}\,, \quad x \in \mathbb{Z}_+\,, \quad \theta \in \mathbb{R} \tag{3.1}$$

The Poisson GVAR(D) process is defined as a GVAR(D) process with the density above and $\theta_{t,m} = \nu_m + \sum_{d=1}^{D} a'_{dm} x_{t-d}$.

Definition 3.2 *(Poisson GVAR(D) process). Let p be the Poisson density with canonical parameter θ as in Equation* (3.1)*. $\{X_t : \Omega \to \mathbb{R}^M\}_{t\in\mathbb{Z}}$ is a Poisson GVAR(D) process if for every $t \in \mathbb{Z}$, one has that for $m = 1, \ldots, M$:*

$$(X_{t,m} \mid X_s = x_s, s < t) \overset{indep.}{\sim} p\left(x_{t,m}; \theta_{t,m} = \nu_m + \sum_{d=1}^{D} a'_{dm} x_{t-d}\right) \tag{3.2}$$

Poisson GVAR(D) processes are well-defined for any values of the parameters $\nu_m \in \mathbb{R}$ and $a'_{dm} \in \mathbb{R}^M$, since for any $z \in \mathbb{Z}_+^D$, one has that $\theta_m(z_1, \ldots, z_D) \in \mathbb{R}$ for every

$m = 1, \ldots, M$.

Conditional mean and variance. By definition, the component random variables $X_{t,m}$ of a Poisson GVAR(D) process are conditionally Poisson with mean $\mathbb{E}(X_{t,m}|X_s, s < t) = \exp\{\theta_{t,m}\}$ and the components are conditionally independent. As a result, Poisson GVAR(D) processes admit a straightforward characterization of the conditional moments of the vector process $\{X_t\}_{t\in\mathbb{Z}}$. Define the vector and matrix quantities:

$$\theta_t = \begin{bmatrix} \theta_{t,1} \\ \vdots \\ \theta_{t,M} \end{bmatrix}, \quad \nu = \begin{bmatrix} \nu_1 \\ \vdots \\ \nu_M \end{bmatrix}, \quad A_d = \begin{bmatrix} a'_{d1} \\ \vdots \\ a'_{dM} \end{bmatrix}$$

Now it follows immediately that the conditional mean and variance are:

$$\mathbb{E}\left(X_t|X_s = x_s, s < t\right) = \exp\{\theta_t\} = \exp\left\{\nu + \sum_{d=1}^{D} A_d x_{t-1}\right\}$$

Since the components are conditionally independent, $\operatorname{cov}(X_{t,i}, X_{t,j} \mid X_s = x_s, s < t) = 0$. As a result, the conditional variance of the process is simply a diagonalization of the conditional mean: $\operatorname{var}\left(X_t|X_s = x_s, s < t\right) = \operatorname{diag}\left(\exp\{\theta_t\}\right)$. Higher moments can be obtained directly from the Poisson distribution.

Likelihood and identifiability. Denote a finite-length M-dimensional Poisson GVAR(D) process by $X = (X_0 \cdots X_T) \in \mathbb{Z}_+^{M\times(T+1)}$, and a process observation by

$x = (x_0 \cdots x_T)$. The joint density of X is:

$$\mathbb{P}(X = x) \propto \exp\left\{\sum_{t=1}^{T}\sum_{m=1}^{M} x_{t,m}\left(\nu_m + \sum_{d=1}^{D} a'_{dm}x_{t-d}\right) - \sum_{t=1}^{T}\sum_{m=1}^{M}\exp\left\{\nu_m + \sum_{d=1}^{D} a'_{dm}x_{t-d}\right\}\right\}$$

Now denote the concatenation of process parameters by $B = (\nu\ A_1 \cdots A_D)$. Then the log-likelihood function is:

$$\ell(B; x) = \sum_{t=1}^{T}\sum_{m=1}^{M} x_{t,m}\left(\nu_m + \sum_{d=1}^{D} a'_{dm}x_{t-d}\right) - \sum_{t=1}^{T}\sum_{m=1}^{M}\exp\left\{\nu_m + \sum_{d=1}^{D} a'_{dm}x_{t-d}\right\} + C \qquad (3.3)$$

The parameters are identifiable just in case any distinct parameters $B \neq B^*$ induce different probability distributions, so that $\mathbb{P}_B(X = x) \neq \mathbb{P}_{B^*}(X = x)$ for some x and some T.

Proposition 3.3 *(Identifiability).* *Let $\{X_t \in \mathbb{Z}_+^M\}_{t=0}^T$ be a Poisson GVAR(D) process with parameter $B = (\nu\ A_1 \cdots A_D)'$. Then B is identifiable for any $T > D$.*

Proof. Let $X = (X_0 \cdots X_T) \in \mathbb{Z}_+^{M\times(T+1)}$ denote the process $\{X_t \in \mathbb{Z}_+^M\}_{t=0}^T$. It suffices to show that $\mathbb{P}_B(X = x) \neq \mathbb{P}_{B^*}(X = x)$.

Consider the first T time points. Either

$$\mathbb{P}_B\left(X_0 = x_0, \ldots, X_{T-1} = x_{T-1}\right) = \mathbb{P}_{B^*}\left(X_0 = x_0, \ldots, X_{T-1} = x_{T-1}\right)$$

for every realization $(x_0 \cdots x_{T-1})$ or not. If not, then B and B^* induce different distributions, so B is identifiable. Suppose, then, that the equality holds, so that B and B^* induce the same distribution on the first T time points.

Let $(x_0 \cdots x_{T-1})$ be an arbitrary realization of the first T time points. One has that in general

$$\mathbb{P}\left(X = x\right) = \mathbb{P}\left(X_s = x_s, s < T\right) \mathbb{P}\left(X_T = x_T | X_s = x_s, s < T\right)$$

It follows that $\mathbb{P}_B(X = x) = \mathbb{P}_{B^*}(X = x)$ just in case

$$\mathbb{P}_B\left(X_T = x_T | X_s = x_s, s < T\right) = \mathbb{P}_{B^*}\left(X_T = x_T | X_s = x_s, s < T\right)$$

Since the realization up to time $T-1$ is arbitrary by hypothesis, the distributions of X induced by B and B^* are the same just in case the conditional distributions of $(X_T | X_s = x_s, s < T)$ are equal under B and B^* for every history $\{x_s \in \mathbb{Z}_+^M\}_{s=0}^{T-1}$.

Now note that by the process definition,

$$\mathbb{P}_B\left(X_T = x_T | X_s = x_s, s < T\right) = \prod_{m=1}^{M} \mathbb{P}_{b_m}\left(X_{T,m} = x_{T,m} | X_s = x_s, s < T\right)$$

where $(X_{T,m} = x_{T,m} | X_s = x_s, s < T)$ are independent Poisson random variables and b_m denote rows of B. With $z_T' = (1\ x_{T-1}' \cdots x_{T-D}')$, the Poisson rates for the conditional distributions are

$$\lambda_{T,m} = \exp\left\{b_m' z_T\right\}$$

so that the vector of Poisson rates is $\lambda_T = \exp\{Bz_T\}$. The rate vectors λ_T and λ_T^* corresponding to B and B^* are equal just in case $Bz_T = B^* z_T$ for some z_T. Let m be the index of any row b_m of B for which $b_m \neq b_m^*$, and let j be any index such

that $b_{mj} \neq b^*_{mj}$. If z_T is equal to a basis vector e_j with a 1 in the jth position, then $b'_m z_T \neq (b^*_m)' z_T$. Consequently, $Bz_T \neq B^* z_T$ and $\lambda_T \neq \lambda^*_T$.

Therefore, for any $B \neq B^*$, one has that $\lambda_T \neq \lambda^*_T$ for at least one $(x_0 \cdots x_{T-1})$. But then one has that

$$\mathbb{P}_B\left(X_T = x_T | X_s = x_s, s < T\right) \neq \mathbb{P}_{B^*}\left(X_T = x_T | X_s = x_s, s < T\right)$$

for at least one x_T.[1] So the conditional distributions of $(X_T | X_s = x_s, s < T)$ differ under B and B^*, which entails that $\mathbb{P}_B(X = x) \neq \mathbb{P}_{B^*}(X = x)$ for at least one x, completing the proof. □

Independent subprocesses. From the likelihood function, it can be shown that the independence of partitions of (finite-length) Poisson GVAR(D) processes reduces to a simple condition on the matrices $A_1, \ldots, A_D$. Define a partition of the process as follows:

$$X^A_t = \left(X_{t,m} : m \in A\right), \quad A \subset \{1, \ldots, M\}$$
$$X^B_t = \left(X_{t,m} : m \in B\right), \quad B \subset \{1, \ldots, M\}$$
$$A \cup B = \{1, \ldots, M\} \quad \text{and} \quad A \cap B = \varnothing$$

[1] To see this, let m be any index for which $\lambda_{Tm} \neq \lambda^*_{Tm}$, and let $x_T = ce_m$, where e_m is a basis vector with a 1 in the mth position and zeroes elsewhere, and where $c \in \mathbb{Z}_+$. Then the ratio of the conditional distributions under B and B^* given z_T as above is

$$\frac{\mathbb{P}_B\left(X_T = x_T | X_{T-D} = x_{T-D}, \ldots, X_{T-1} = x_{T-1}\right)}{\mathbb{P}_{B^*}\left(X_T = x_T | X_{T-D} = x_{T-D}, \ldots, X_{T-1} = x_{T-1}\right)} = \left(\exp\left\{-\sum_{j \neq m}\left(\lambda_{Tj} - \lambda^*_{Tj}\right)\right\}\right)\left(\frac{\lambda_{Tm}}{\lambda^*_{Tm}}\right)^c$$

which is of the form ax^c where $a \neq 0$ and $x \neq 1$. This cannot be equal to 1 for every $c \in \mathbb{Z}_+$.

The subprocesses X_t^A and X_t^B are independent just in case:

$$\mathbb{P}(X_0 = x_0, \ldots, X_T = x_t) = \mathbb{P}(X_0^A = x_0^A, \ldots, X_T^A = x_t^A)\mathbb{P}(X_0^B = x_0^B, \ldots, X_T^B = x_t^B)$$

In other words, the joint density must factor into a function of only the components $x_{t,m}$ for $m \in A$ and a function of only the components $x_{t,m}$ for $m \in B$. Equivalently, the log-likelihood function (Equation (3.3)) must be a sum of such functions. With $x^A = (x_0^A \cdots x_T^A)$ and $x^B = (x_0^B \cdots x_T^B)$ denoting the portions of an arbitrary process observation that correspond to the partitioning, the latter condition is that:

$$X_t^A \perp X_t^B \quad \Longleftrightarrow \quad \ell(\nu, A_1, \ldots, A_D; x) = g(x^A) + h(x^B) \quad \text{for some functions } h, g$$

Assume for the sake of argument and without loss of generality that the partition divides the first K components from the subsequent $M - K$ components. In other words, assume that the component indices m are arranged so that $A = \{1, \ldots, K\}$ and $B = \{K+1, \ldots, M\}$, yielding $X_t = (X_t^A \; X_t^B)$. Then, splitting the sum over m in the likelihood in correspondence to the partitioning yields:

$$\begin{aligned}\ell(\nu, A_1, \ldots, A_D; x) = &\sum_{m=1}^{K}\sum_{t=1}^{T}\left(x_{t,m}\left(\nu_m + \sum_{d=1}^{D} a'_{dm}x_{t-d}\right) - \exp\left\{\nu_m + \sum_{d=1}^{D} a'_{dm}x_{t-d}\right\}\right) \\ &+ \sum_{m=K+1}^{M}\sum_{t=1}^{T}\left(x_{t,m}\left(\nu_m + \sum_{d=1}^{D} a'_{dm}x_{t-d}\right) - \exp\left\{\nu_m + \sum_{d=1}^{D} a'_{dm}x_{t-d}\right\}\right) + C\end{aligned}$$

Since this is the only possible representation of the log-likelihood that separates the $x_{t,m}$ terms in correspondence with the partitioning given by A and B, the indepen-

dence condition can only be satisfied if the first summation includes no terms in x^B and the second summation includes no terms in x^A.

The first summation (from $m = 1$ to K) is a function of only the first K components (those in A) just in case the term $\sum_{d=1}^{D} a'_{dm} x_{t-d}$ does not depend on $x_{t,j}$ for any $j > K$ and any t. Likewise, the second summation is a function of only the last $M - K$ components just in case the same term does not depend on $x_{t,j}$ for any $j \leq K$ and any t. This is true just in case $a_{dij} = 0$ and $a_{dji} = 0$ for every d whenever $i \leq K$ and $j > K$. Therefore:

$$X_t^A \perp X_t^B \quad \Longleftrightarrow \quad a_{dij} = a_{dji} = 0 \text{ for every } i \in A,\ j \in B,\ d = 1, \ldots, D$$

Another way of stating this condition is that the matrices $A_1, \ldots, A_D$ must be block-diagonal. Let $A_{d11} \in \mathbb{R}^{K \times K}$, $A_{d12} \in \mathbb{R}^{K \times (M-K)}$, $A_{d21} \in \mathbb{R}^{(M-K) \times K}$, and $A_{d22} \in \mathbb{R}^{(M-K) \times (M-K)}$ denote partitions of A_d so that:

$$A_d = \left[\begin{array}{c:c} A_{d11} & A_{d12} \\ \hdashline A_{d21} & A_{d22} \end{array}\right]$$

Then, the independence condition for subprocesses X_t^A and X_t^B can be restated as:

$$X_t^A \perp X_t^B \quad \Longleftrightarrow \quad A_{d12} = 0 \text{ and } A_{d21} = 0 \text{ for every } d = 1, \ldots, D$$

This extends directly to arbitrarily many partitions by recursively applying the con-

dition.[2] As a result, collections of independent subprocesses can be identified from direct examination of the matrices $A_1, \ldots, A_D$: if any permutation of the indices yields n block-diagonal matrices, then the process contains n independent subprocesses.

Parameter interpretation and graphical representation. The Poisson distribution is often construed as a probability model for event counts, in which case the parameter is thought of as an event rate. Under this interpretation, the parameters ν_m give baseline event rates for each component. That is, if the process remains at the origin for D or more time steps, each of the m component has a corresponding mean $\exp\{\nu_m\}$.

By contrast, the elements of $A_1, \ldots, A_D$ represent mutual influences between the components' event rates. Each event at time t in the kth component is associated with a change in the mth component's event rate for the later time step $t + d$ of a factor of $\exp(a_{dmk})$, after accounting for other such associations. The sign of the entries indicates the direction of influence. If $a_{dmk} < 0$, then the relationship between components m and k is inhibitory: the occurrence of events in the kth component reduces the event rate for the mth component over d time steps. If on the other hand $a_{dmk} > 0$, then the relationship between components m and k is excitory: the occurrence of events in the kth component increases the event rate for the mth component over d time steps.

It will be convenient in later sections to consider graphical representations of these relationships. It is straightforward to define directed graphs $G_1, \ldots, G_D$ on the set of

[2] For example, say one has $X_t^A \perp X_t^B$. If there exists a partition X_t^C and X_t^D of X_t^A satisfying the independence condition, then one has $X_t^B \perp X_t^C \perp X_t^D$.

vertices $V = \{1, \ldots, M\}$ based on the sparsity patterns of $A_1, \ldots, A_D$ and their entries:

$$G_d = (V, E_d), \quad E_d = \{(i, j) : a_{dji} \neq 0\}, \quad d = 1, \ldots, D \tag{3.4}$$

Additionally, weights can be assigned to each edge in E_d based on the corresponding entries a_{dji}. For example, it can be useful to represent the sign of the entry so that the graph indicates whether the influences displayed are excitory or inhibitory.

3.2.2 Process stability

Although the class of Poisson GVAR(D) processes is well-defined, not all processes in the class are well-behaved. Consider, for example, the univariate case ($M = 1$) with $D = 1$:

$$(X_t \mid X_{t-1} = x_{t-1}) \sim p(x_t; \exp\{\alpha x_{t-1} + \nu\})$$

Since the exponential function is convex, it follows by the law of total expectation and Jensen's inequality that:

$$\mathbb{E}X_t = \mathbb{E}\left(\mathbb{E}\left(X_t | X_{t-1}\right)\right) = \mathbb{E}\left(\exp\{\alpha X_{t-1} + \nu\}\right) \geq \exp\{\alpha \mathbb{E}X_{t-1} + \nu\}$$

Recursive application of Equation (3.2.2) gives that:

$$\mathbb{E}X_t \geq \exp\{\alpha \mathbb{E}X_{t-1} + \nu\} \geq \exp\{\alpha \exp\{\alpha \mathbb{E}X_{t-2} + \nu\} + \nu\} \geq \cdots$$

Therefore, for any k, one has:

$$\mathbb{E}X_t \geq \exp\{\nu + \alpha \exp\{\nu + \alpha \exp\{\cdots \exp\{\nu + \alpha \mathbb{E}X_{t-k}\}\}\}\}$$

Since X_t is nonnegative, if $\alpha > \exp\{-(\nu+1)\}$ then the mean sequence $\{\mathbb{E}X_t\}_{t\in\mathbb{Z}}$ is monotonic and diverges rapidly.[3] On the other hand, if $\alpha \leq 0$ then for all $t \in \mathbb{Z}$:

$$\mathbb{E}X_t = \mathbb{E}\left(\mathbb{E}\left(X_t|X_{t-1}\right)\right) = \mathbb{E}\left(\exp\{aX_{t-1} + \nu\}\right) \leq \mathbb{E}\exp\{\nu\} \leq \exp\{\nu\}$$

Therefore, depending on the value of α, the process means may or may not be finite.

In consideration of the above, it is useful to introduce a distinction between processes with finite moments up to a certain order and all other processes within the class. To that end, define 'first-order stability' and 'second-order stability' as follows.

Definition 3.4 *(Stability).* *A process $\{X_t\}_{t\in\mathbb{Z}}$ is first-order stable if $\sup_t \mathbb{E}X_{t,m} < \infty$ for every $m = 1,\ldots,M$ and second-order stable if $\sup_t \mathbb{E}X_{t,m}^2 < \infty$ for every $m = 1,\ldots,M$.*

Stability in higher-order moments can be defined similarly. For the present purposes, it is said that a process is stable if it is first-order and second-order stable. The above discussion establishes the following lemma.

Lemma 3.5. *Let $\{X_t\}_{t\in\mathbb{Z}}$ be a Poisson autoregressive process with parameter α (that is, GVAR(D) with $D = 1$ and $M = 1$ and $A = \alpha$). $\{X_t\}$ is stable whenever $\alpha \leq 0$ and unstable whenever $\alpha > \exp\{-(\nu+1)\}$.*

[3] Further discussion on this condition is given in Section 3.3.3.

As the lemma pertaining to the univariate case suggests, the values of $\nu, A_1, \ldots, A_D$ jointly determine whether a Poisson GVAR(D) process is first-order or second-order stable. The subject of the following section is establishing certain (sufficient) conditions under which order $D = 1$ processes satisfy these stability properties.

3.3 Stability conditions for Poisson GVAR(1) processes

The specification of stable classes of processes is in general only of interest when the parameters $A_1, \ldots, A_D$ contain positive-signed entries: that is, when $a_{dij} > 0$ for some d, i, j. This is due to the following result.

Lemma 3.6. *Any Poisson GVAR(D) process with $a_{dij} \leq 0$ for every d, i, j is stable.*

Proof. If all parameters a_{dij} are negative, then each of the components $X_{t,1}, \ldots, X_{t,M}$ are stochastically dominated by Poisson random variables with rates $\exp\{\nu_m\}$ and therefore have finite moments. The Poisson distribution is stochastically ordered by rate, and the conditional distributions of $(X_{t,m}|X_s = x_s, s < t)$ are Poisson with rates $e^{\theta_{t,m}}$. If $a_{dij} \leq 0$, then for any $x_{t-1}, \ldots, x_{t-D}$ and every $m = 1, \ldots, M$:

$$\exp\{\theta_{t,m}\} = \exp\left\{\nu_m + \sum_{d=1}^{D} a'_m x_{t-d}\right\} \leq \exp\{\nu_m\}$$

As a result, if $Y_m \sim p(y; \nu_m)$, then for every x and $x_s, s < t$:

$$P(X_{t,m} > x|X_s = x_s, s < t) \leq P(Y_m > x|X_s = x_s, s < t)$$

Taking expectations with respect to X_s yields that Y_m stochastically dominates $X_{t,m}$ for every $t \in \mathbb{Z}$:

$$P(X_{t,m} > x) \leq P(Y_m > x)$$

A consequence of this ordering is that the moments of $X_{t,m}$ are dominated by the moments of Y_m. Since the first two moments of Y_m are finite, so are those of $X_{t,m}$. □

In view of the above, this section discusses the stability of processes with positive-signed parameters in the context of order $D = 1$ processes. First, a stable class of processes is identified in Section 3.3.1 via a graphical constraint. Next, criteria for the parameter magnitudes that ensure practical bounds on process moments are developed in Section 3.3.2. Finally, Section 3.3.3 identifies a class of graphical structures that correspond to unstable processes, and indicates parameter regimes with unknown stability properties.

3.3.1 Stability of graphically constrained processes

The following establishes the stability of any Poisson GVAR(1) process that satisfies a constraint on the positive-signed entries of the parameter matrix A. The constraint is expressed in terms of graphical structure for ease of interpretation, rather than as a purely technical constraint on A. The result is given immediately below, followed by a proof.

Proposition 3.7. *Let $\{X_t\}_{t\in\mathbb{Z}}$ be an M-dimensional Poisson GVAR(1) process with parameters ν and A, and let G denote the graph of A as defined in Equation* (3.4)*, with*

edge weights indicating the sign of the corresponding entry in A. If G is partitioned into positive-signed and negative-signed edges, then $\{X_t\}_{t\in\mathbb{Z}}$ *is first-order and second-order stable whenever the positive-signed partition of G has no paths exceeding length 1.*

Proof. Consider the partition of A into positive and negative portions:

$$A = A^+ + A^- , \qquad a^+_{ij} = a_{ij}\mathbb{1}\{a_{ij} > 0\} , \quad a^-_{ij} = a_{ij}\mathbb{1}\{a_{ij} < 0\}$$

Let G^+, G^- denote the corresponding partition of the graph G into a positive-signed portion G^+ generated by A^+ and a negative-signed portion G^- generated by A^-. Assume that the maximum path length in G^+ is 1, so that $\|a^+_j\|_0 = 0$ whenever $a^+_{mj} > 0$ (note that this entails $a_{mm} \leq 0$ for every m).

Each component $X_{t,m}$ has either $\|a^+_m\|_0 = 0$ or $\|a^+_m\|_0 > 0$. If $\|a^+_m\|_0 = 0$, then the component has an in-degree of zero in G^+ and in fact $a^+_m = 0$. In this case the same

argument pertaining to negative entries applies:

$$\begin{aligned}
\mathbb{E}X_{t,m} &= \mathbb{E}\left(\exp\{a'_m X_{t-1} + \nu_m\}\right) \\
&\leq \mathbb{E}\left(\exp\{a_m^{+\prime} X_{t-1} + \nu_m\}\right) \\
&= \mathbb{E}\left(\exp\{\nu_m\}\right) \\
&= \exp\{\nu_m\} \\
\mathbb{E}X_{t,m}^2 &= \mathbb{E}\left(\exp\{a'_m X_{t-1} + \nu_m\}\right)\left(\exp\{a'_m X_{t-1} + \nu_m\} + 1\right) \\
&\leq \mathbb{E}\left(\exp\{a_m^{+\prime} X_{t-1} + \nu_m\}\right)\left(\exp\{a_m^{+\prime} X_{t-1} + \nu_m\} + 1\right) \\
&= \mathbb{E}\left(\exp\{\nu_m\}\left(\exp\{\nu_m\} + 1\right)\right) \\
&= \exp\{\nu_m\}\left(\exp\{\nu_m\} + 1\right)
\end{aligned}$$

So if $\|a_m^+\|_0 = 0$, then $X_{t,m}$ has finite first and second moments.

Consider, then, the components $X_{t,m}$ with $\|a_m^+\|_0 > 0$. These components have positive in-degree in G^+ and are the terminal points of each path in G^+. Denoting the set of indices of components that influence $X_{t,m}$ by $I_m = \{i : a_{mi}^+ > 0\}$, so that

$a_m^{+\prime} x = \sum_{i \in I_m} a_{mi} x_i$. Now, one has that:

$$\begin{aligned}
\mathbb{E}X_{t,m} &= \mathbb{E}\left(\exp\left\{a_m' X_{t-1} + \nu_m\right\}\right) \\
&\leq \mathbb{E}\left(\exp\left\{a_m^{+\prime} X_{t-1} + \nu_m\right\}\right) \\
&= \exp\{\nu_m\}\mathbb{E}\left(\exp\left\{\sum_{i \in I_m} a_{mi} X_{t-1,i}\right\}\right) \\
&= \exp\{\nu_m\}\mathbb{E}\left(\mathbb{E}\left(\exp\left\{\sum_{i \in I_m} a_{mi} X_{t-1,i}\right\} | X_{t-2}\right)\right) \\
&= \exp\{\nu_m\}\mathbb{E}\left(\prod_{i \in I_m} \mathbb{E}\left(\exp\left\{a_{mi} X_{t-1,i}\right\} | X_{t-2}\right)\right)
\end{aligned}$$

By hypothesis, $\|a_i^+\|_0 = 0$ for every $i \in I_m$, so each $X_{t-1,i}$ is stochastically dominated by a Poisson random variable with rate e^{ν_i}. Consequently, letting $Y_i \sim p(y; \exp\{\nu_i\})$, the innermost expectation can be calculated explicitly using the moment generating function $M_{Y_i}(t)$:

$$\mathbb{E}\left(\exp\left\{a_{mi} X_{t-1,i}\right\} | X_{t-2}\right) \leq \mathbb{E}\left(\exp\left\{a_{mi} Y_i\right\}\right) = M_{Y_i}\left(a_{mi}\right) = \exp\left\{e^{\nu_i}\left(e^{a_{mi}} - 1\right)\right\}$$

The inequality follows by the stochastic ordering.[4] As a result:

$$\mathbb{E}X_{t,m} \leq \exp\{\nu_m\}\mathbb{E}\left(\prod_{i \in I_m} \exp\left\{e^{\nu_i}\left(e^{a_{mi}} - 1\right)\right\}\right) = \exp\left\{\nu_m + \sum_{i \in I_m} e^{\nu_i}\left(e^{a_{mi}} - 1\right)\right\}$$

[4] If X is stochastically dominated by Y, then for any function ϕ monotonically increasing on the support of X and Y, one has that $\mathbb{E}\phi(X) \leq \mathbb{E}\phi(Y)$. The function $f(x) = \exp\{ax\}$ is monotonically increasing for $a > 0$.

By a similar calculation:

$$
\begin{aligned}
\mathbb{E}X_{t,m}^2 &= \mathbb{E}\left(\exp\{a_m' X_{t-1} + \nu_m\}\right)\left(\exp\{a_m' X_{t-1} + \nu_m\} + 1\right) \\
&\leq \mathbb{E}\left(\exp\{a_m^{+\prime} X_{t-1} + \nu_m\}\right)\left(\exp\{a_m^{+\prime} X_{t-1} + \nu_m\} + 1\right) \\
&= \mathbb{E}\left(\exp\left\{\sum_{i\in I_m} 2a_{mi} X_{t-1,i} + 2\nu_m\right\}\right) + \mathbb{E}\left(\exp\left\{\sum_{i\in I_m} a_{mi} X_{t-1,i} + \nu_m\right\}\right) \\
&= \mathbb{E}\left(\mathbb{E}\left(\exp\left\{\sum_{i\in I_m} 2a_{mi} X_{t-1,i} + 2\nu_m\right\} | X_{t-2}\right)\right) + \mathbb{E}\left(\mathbb{E}\left(\exp\left\{\sum_{i\in I_m} a_{mi} X_{t-1,i} + \nu_m\right\} | X_{t-2}\right)\right) \\
&= \mathbb{E}\left(\prod_{i\in I_m} \mathbb{E}\left(\exp\left\{2a_{mi} X_{t-1,i} + 2\nu_m\right\} | X_{t-2}\right)\right) + \mathbb{E}\left(\prod_{i\in I_m} \mathbb{E}\left(\exp\left\{a_{mi} X_{t-1,i} + \nu_m\right\} | X_{t-2}\right)\right) \\
&\leq \mathbb{E}\left(\exp\left\{2\nu_m + \sum_{i\in I_m} e^{\nu_i}(e^{2a_{mi}} - 1)\right\}\right) + \mathbb{E}\left(\exp\left\{\nu_m + \sum_{i\in I_m} e^{\nu_i}(e^{a_{mi}} - 1)\right\}\right) \\
&= \exp\left\{2\nu_m + \sum_{i\in I_m} e^{\nu_i}(e^{2a_{mi}} - 1)\right\} + \exp\left\{\nu_m + \sum_{i\in I_m} e^{\nu_i}(e^{a_{mi}} - 1)\right\}
\end{aligned}
$$

Together, the above inequalities establish that the process is first-order and second-order stable for any values of the parameters A, ν. Since the calculations hold for every $t \in \mathbb{Z}$, it follows that:

$$
\begin{aligned}
\sup_t \mathbb{E}X_{t,m} &\leq \exp\left\{\nu_m + \sum_{j=1}^{M} e^{\nu_j}(e^{a_{mj}^+} - 1)\right\} \\
\sup_t \mathbb{E}X_{t,m}^2 &\leq \exp\left\{2\nu_m + \sum_{j=1}^{M} e^{\nu_j}(e^{2a_{mj}^+} - 1)\right\} + \exp\left\{\nu_m + \sum_{j=1}^{M} e^{\nu_j}(e^{a_{mj}^+} - 1)\right\}
\end{aligned}
$$

Note that this includes the case where $\|a_m^+\|_0 = 0$, since then $(e^{2a_{mj}^+} - 1) = 0$ for every j. Therefore, any process for which G^+ has a maximum path length of 1 is stable. $\square$

3.3.2 Moment bounds for graphically-constrained processes

Inspection of the bounds used to establish the stability of graphically-constrained Poisson GVAR(1) processes in Section 3.3.1 should suffice to convince that although the process moments can be bounded by finite functions of the parameters, the bounds can be quite large. This motivates distinguishing between theoretically stable processes — processes having finite first and second moments — and practically stable processes in which the moments are not only finite but bounded by a specific constant.

Example. In the arguments of the previous section, the inequalities account for the possibility that $A \neq A^+$; however, if $A = A^+$, then the same calculations hold with equality. The calculations produce functions of the parameters of the form $\sum_j e^{\nu_j}(e^{a^+_{mj}} - 1)$ and $\sum_j e^{\nu_j}(e^{2a^+_{mj}} - 1)$. For compactness, denote these functions by the notation ξ_m, ψ_m:

$$\begin{bmatrix} \xi_1 \\ \vdots \\ \xi_M \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^M e^{\nu_j}(e^{a^+_{1j}} - 1) \\ \vdots \\ \sum_{j=1}^M e^{\nu_j}(e^{a^+_{Mj}} - 1) \end{bmatrix}, \qquad \begin{bmatrix} \psi_1 \\ \vdots \\ \psi_M \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^M e^{\nu_j}(e^{2a^+_{1j}} - 1) \\ \vdots \\ \sum_{j=1}^M e^{\nu_j}(e^{2a^+_{Mj}} - 1) \end{bmatrix}$$

Now, if $A = A^+$, then one has that for every $t \in \mathbb{Z}$:

$$\mathbb{E}X_{t,m} = \exp\{\nu_m + \xi_m\}$$

$$\mathbb{E}X^2_{t,m} = \exp\{\nu_m + \xi_m\} + \exp\{2\nu_m + \psi_m\}$$

In this case, the process is in fact stationary, and moreover, the largest moments are:

$$\sup_m \mathbb{E}X_{t,m} = \max\{\exp\{\nu_m + \xi_m\}\}$$
$$\sup_m \mathbb{E}X_{t,m}^2 = \max\{\exp\{\nu_m + \xi_m\} + \exp\{2\nu_m + \psi_m\}\}$$

These suprema can be very large for parameter magnitudes of seemingly innocuous magnitudes. For instance, if A contains a single nonzero entry a_{21} and ν contains a single nonzero entry ν_2, varying the two nonzero parameters in combinations of seemingly small values between -0.5 and 1.5 can produce a process with means ranging from 2.8 to 311 and variances ranging from 42 to 4.632 $\times 10^{13}$. Several such combinations are shown in Table 3.1. This example indicates that without any further

ν_2	a_{21}	$\sup_m \mathbb{E}X_{t,m}$	$\sup_m \mathrm{Var}X_{t,m}$
-0.5	1	2.835	42.986
-0.5	1.5	8.263	1.064 $\times 10^5$
0	1	5.575	569.789
0	1.5	32.515	1.944 $\times 10^8$
0.5	1	16.996	3.729 $\times 10^4$
0.5	1.5	311.169	4.632 $\times 10^{13}$

Table 3.1: Means and variances of a stable and stationary process with a single nonzero parameter a_{21} and a single nonzero intercept ν_2.

constraints on parameter magnitudes, there are regions in the parameter space where the process moments become so large that it is of little practical importance whether they are finite (Knuth, 1976). The result below develops a heuristic for identifying constraint regions within which process moments are bounded by a specific constant.

Moment bounds. The moment bounds above can be equated with a constant C

to define a constraint on the parameter space that ensures the process moments are bounded by C. However, this strategy produces a somewhat complex constraint region that is a function of all positive parameters in A and ν. By loosening the bounds, a simpler constraint can be obtained in terms of maximum parameter magnitudes that accomplishes the same objective.

Proposition 3.8. *Let $\{X_t\}_{t\in\mathbb{Z}}$ be a Poisson GVAR(1) process with parameters A, ν. Let A^+ denote the positive-signed portion of A and G^+ denote the graph generated by A^+. Denote the maximum in-degree of G^+ and the maximum parameter magnitudes by:*

$$d^* = \max_m\{\|a_m^+\|_0\}\ , \qquad \alpha^* = \max_{i,j}\{a_{ij}^+\}\ , \qquad \nu^* = \max_m\{\nu_m\}$$

Then it follows that for any C:

$$\alpha^* \le \log\left(\frac{\log C - \nu^*}{d^*\exp\{\nu^*\}} + 1\right) \implies \sup_{t,m}\mathbb{E}X_{t,m} \le C$$

$$\alpha^* \le \frac{1}{2}\log\left(\frac{\log C - \log 2 - \nu^* - \nu^*\mathbb{1}\{\nu^* > 0\}}{d^*\exp\{\nu^*\}} + 1\right) \implies \sup_{t,m}\mathbb{E}X_{t,m}^2 \le C$$

Proof. Following the proof of Proposition 3.7, for every t, m, one has:

$$\begin{aligned}
\mathbb{E}X_{t,m} &\leq \exp\left\{\nu_m + \sum_{j=1}^{M} e^{\nu_j}\left(e^{a^+_{mj}} - 1\right)\right\} \\
&\leq \exp\{\nu^* + d^* e^{\nu^*}\left(e^{\alpha^*} - 1\right)\} \\
\mathbb{E}X^2_{t,m} &\leq \exp\left\{2\nu_m + \sum_{j=1}^{M} e^{\nu_j}\left(e^{2a^+_{mj}} - 1\right)\right\} + \exp\left\{\nu_m + \sum_{j=1}^{M} e^{\nu_j}\left(e^{a^+_{mj}} - 1\right)\right\} \\
&\leq \exp\{2\nu^* + d^* e^{\nu^*}\left(e^{2\alpha^*} - 1\right)\} + \exp\{\nu^* + d^* e^{\nu^*}\left(e^{\alpha^*} - 1\right)\}
\end{aligned}$$

Equating the right-hand side of each bound with C completes the proof. □

These constraint regions are plotted in Figure 3.1 for $C = 10^6$ and maximum in-degrees between 1 and 6. The figure indicates the decrease in the maximum entry in A required to accommodate increasing maxima of ν in order to maintain a fixed bound on first and second moments. The changes in α^* are approximately linear in ν^* when $\nu^* < 0$, and exponential when $\nu^* > 0$.

3.3.3 Other graphical structures

A notable feature of the graphical constraint above is that it establishes stability based only on the positive-signed structure G^+. More generally, it can be shown that a Poisson GVAR(1) process is stable if the structure G^+ alone characterizes a stable process.

Proposition 3.9. *Let $\{X_t\}_{t\in\mathbb{Z}}$ be a Poisson GVAR(1) process with parameters ν, A, and let $\{X^+_t\}_{t\in\mathbb{Z}}$ be a Poisson GVAR(1) process with parameters ν, A^+, where $a^+_{ij} =$*

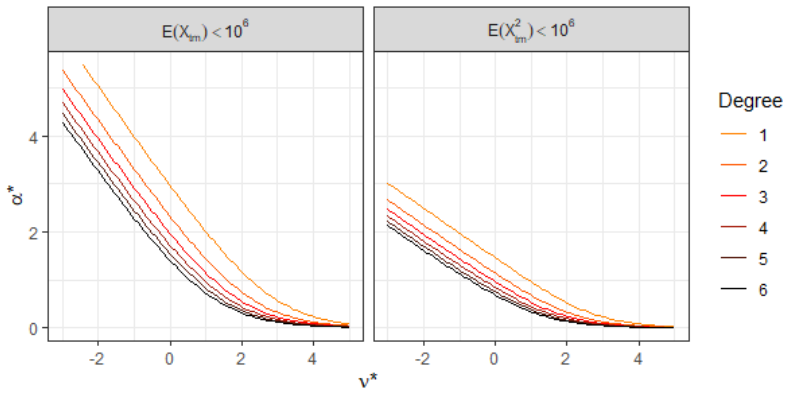


Figure 3.1: Contours indicating the maximum magnitude of entries in A ($\alpha^* = \max_{i,j} a_{ij}$) as a function of the maximum intercept term ($\nu^* = \max_m \nu_m$) for which a Poisson GVAR(1) process with parameters ν, A has a first moment bounded by $C = 10^6$ (left) and a second moment bounded by $C = 10^6$ (right), under the graphical constraint on path length and for various maximum in-degrees.

$a_{ij}\mathbb{1}\{a_{ij} \geq 0\}$. *Then if* $\{X_t^+\}$ *is stable, so is* $\{X_t\}$.

Proof. It suffices to establish the stochastic ordering $X_{t,m} \leq_{st} X_{t,m}^+$. Two random variables X and Y are stochastically ordered as $X \leq_{st} Y$ just in case $P(X \leq x) \geq P(Y \leq x)$ for every x. A useful property of stochastically ordered random variables is that for any function f that is monotonically increasing on the support of X and Y, $X \leq_{st} Y$ implies $f(X) \leq_{st} (Y)$ and $\mathbb{E}f(X) \leq \mathbb{E}f(Y)$. Therefore, since X_t and X_t^+ are nonnegative random vectors, if $X_{t,m} \leq_{st} X_{t,m}^+$ then $\mathbb{E}X_{t,m}^q \leq \mathbb{E}(X_{t,m}^+)^q$ for every q, so the stability of X_t^+ ensures the stability of X_t.

For ease of notation, let $X_t \leq_{st} X_t^+$ indicate that $X_{t,m} \leq_{st} X_{t,m}^+$ elementwise for each $m = 1, \ldots, M$. The strategy will be to consider a common starting point $t = 0$, establish $X_1 \leq_{st} X_1^+$, and then show that if $X_{t-1} \leq_{st} X_{t-1}^+$ then $X_t \leq_{st} X_t^+$, establishing the ordering by induction. Let $\lambda_t = \exp\{\nu + AX_{t-1}\}$ and $\lambda_t^+ = \exp\{\nu + A^+X_{t-1}^+\}$ indicate

the conditional rates of X_t and X_t^+ given their respective histories.

First let $X_0 = X_0^+ = 0$. Then $\lambda_1 = \lambda_1^+ = \exp\{\nu\}$, so $X_1 \stackrel{d}{=} X_1^+$ and it follows trivially that $X_1 \leq_{st} X_1^+$.

Next suppose that for an arbitrary t, $X_{t-1} \leq_{st} X_{t-1}^+$. Since elementwise $\exp\{\nu + A^+x\}$ is a monotonically increasing function of x on $[0,\infty)^M$, and $\exp\{\nu + Ax\} \leq \exp\{\nu + A^+x\}$ elementwise, it follows that:

$$\lambda_t = \exp\{\nu + AX_{t-1}\} \leq_{st} \exp\{\nu + A^+X_{t-1}^+\} = \lambda_t^+$$

Now let $G_{t,m}(\lambda) = P(\lambda_{t,m} \leq \lambda)$ and similarly $G_{t,m}^+(\lambda) = P(\lambda_{t,m}^+ \leq \lambda)$. If $f(x;\lambda)$ denotes the Poisson density, then following results discussed in Karlis and Xekalaki (2005) one has:

$$P(X_{t,m} \leq x) = \int_0^\infty f(x;\lambda)G_{t,m}(\lambda)d\lambda$$
$$P(X_{t,m}^+ \leq x) = \int_0^\infty f(x;\lambda)G_{t,m}^+(\lambda)d\lambda$$

Since $\lambda_t \leq_{st} \lambda_t^+$, $G_{t,m}(\lambda) - G_{t,m}^+(\lambda) \geq 0$ for each m, so it follows that

$$P(X_{t,m} \leq x) - P(X_{t,m}^+ \leq x) = \int_0^\infty F(x;\lambda)\left(G_{t,m}(\lambda) - G_{t,m}^+(\lambda)\right)d\lambda \geq 0$$

Therefore $P(X_{t,m} \leq x) \geq P(X_{t,m}^+ \leq x)$ for every m, *i.e.*, $X_t \leq_{st} X_t^+$, concluding the proof. □

In view of the above, a key question is whether and under what conditions longer

paths in G^+ characterize stable processes. Loops can be examined as a limiting case, and are in fact unstable for relatively large regimes of the parameter space, as the following result shows.

Proposition 3.10. *Let $\{X_t\}_{t\in\mathbb{Z}}$ be an M-dimensional Poisson GVAR(1) process with parameters ν and A, and let G denote the graph of A as defined in Equation* (3.4)*, with edge weights indicating the sign and magnitude of the corresponding entry in A. If the positive-signed portion of G contains a loop, and no component in the loop is the terminus of any edge in the negative-signed portion of G, then $\{X_t\}_{t\in\mathbb{Z}}$ is not stable whenever the minimum edge weight α^* and intercept parameter ν^* in the loop satisfy $\nu^* > -\log\alpha^* - 1$.*

Proof. Let A^+, A^-, G^+, and G^- be defined as in Section 3.3.1. Suppose G^+ contains a loop of length $L \leq M$, and assume without loss of generality that the indices m are ordered so that the loop connects components $1, \ldots, L$ in order. That is, assume $\{a_{21}, a_{32}, \ldots, a_{L(L-1)}, a_{1L}\} \subseteq \{a_{ij} : a_{ij} \neq 0\}$. Finally, assume that $\|a_m^-\|_0 = 0$ for every $m \leq L$, so that no components involved in the loop are terminal points for any path in G^-.

Denote $a_{(l+1)l}$ by α_l for $l = 1, \ldots, L-1$, and denote a_{1L} by α_L. The α_l terms parametrize the edge weights in the loop. For example, if no other nonzero entries are present and $\{a_{ij} : a_{ij} \neq 0\} = \{a_{21}, a_{32}, \ldots, a_{L(L-1)}, a_{1L}\}$, then the upper $L \times L$ block

of the parameter matrix is:

$$A = \begin{bmatrix} 0 & \alpha_1 & 0 & \cdots & 0 & \cdots \\ 0 & 0 & \alpha_2 & \cdots & 0 & \cdots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \\ 0 & 0 & 0 & \cdots & \alpha_{L-1} & \cdots \\ \alpha_L & 0 & 0 & \cdots & 0 & \cdots \\ \vdots & \vdots & \vdots & & \vdots & \ddots \end{bmatrix}$$

The conditions of the proof allow that other entries in this block are nonnegative, but not less than zero.

Now, by nonnegativity of X_t and a_m and Jensen's inequality, one has that for every $m = 2, \ldots, L$ and any $t \in \mathbb{Z}$:

$$\begin{aligned} \mathbb{E}X_{t,m} &= \mathbb{E}\left(\mathbb{E}(X_{t,m}|X_{t-1})\right) \\ &= \mathbb{E}\exp\left\{a'_m X_{t-1} + \nu_m\right\} \\ &\geq \mathbb{E}\exp\left\{\alpha_m X_{t-1,m-1} + \nu_m\right\} \\ &\geq \exp\left\{\alpha_m \mathbb{E}X_{t-1,m-1} + \nu_m\right\} \end{aligned}$$

By the same argument, for $m = 1$ one has:

$$\mathbb{E}X_{t,1} \geq \exp\left\{\alpha_1 \mathbb{E}X_{t-1,L} + \nu_1\right\}$$

Then, it follows that:

$$\begin{aligned}
\mathbb{E}X_{t,L} &\geq \exp\{\nu_L + \alpha_L \mathbb{E}X_{t-1,L-1}\} \\
&\geq \exp\{\nu_L + \alpha_L \exp\{\nu_{L-1} + \alpha_{L-1}\mathbb{E}X_{t-2,L-2}\}\} \\
&\vdots \\
&\geq \exp\{\nu_L + \alpha_L \exp\{\cdots \exp\{\nu_1 + \alpha_1 \mathbb{E}X_{t-L,L}\}\cdots\}\}
\end{aligned}$$

Now, replacing each α_l and ν_l by the minima $\alpha^* = \min_l\{\alpha_l\}$ and $\nu^* = \min_{m\leq L}\{\nu_m\}$, one has:

$$\mathbb{E}X_{t,L} \geq \exp\{\nu^* + \alpha^* \exp\{\cdots \exp\{\nu^* + \alpha^* \mathbb{E}X_{t-L,L}\}\cdots\}\}$$

More generally, this inequality holds if L is replaced by any $m = 1, \ldots, L$.

Now consider for a fixed t and any $m \leq L$ the sequence of expectations $\{\mathbb{E}X_{t+kL,m}\}_{k=0}^{\infty}$. The above inequality establishes that this sequence is bounded below by the recursive sequence $\{s_{kL}\}_{k=0}^{\infty}$ defined by:

$$\begin{aligned}
s_0 &= \mathbb{E}X_{t,m} \\
s_{n+1} &= \exp\{\nu^* + \alpha^* s_n\}
\end{aligned}$$

In general, if $\lim_{n\to\infty} s_n < \infty$, then the limit must be a fixed point of the sequence, *i.e.*, a point s satisfying $s = \exp\{\nu^* + \alpha^* s\}$. It is straightforward to show that no such point exists whenever $\nu^* > -\log\alpha^* - 1$.[5] Furthermore, in this case the sequence is

[5]The function used to define the recursion is $f(x) = \exp\{\nu^* + \alpha^* x\}$. Limits can exist for sequences defined by recursive exponentiation under a variety of interesting conditions (Bromer, 1987; Wassell, 2000). In this particular case, it can be shown by a calculus argument that $f(x) > x$ whenever ν^* is

monotonic. Therefore, if $\nu^* > -\log\alpha^* - 1$ then $\{s_n\}$ diverges, from which it follows that for every t and any $m \le L$:

$$\nu^* > -\log\alpha^* - 1 \qquad \Longrightarrow \qquad \mathbb{E}X_{t+kL,m} \to \infty \text{ as } k \to \infty$$

Consequently, the process is not first-order stable. □

Note that these results leave open the possibility that structure in G^- can mitigate instability in G^+.

3.4 Estimation

This section provides a brief overview of unconstrained likelihood estimation for Poisson GVAR(D) processes, deferring to Appendix A for computational details, and develops a characterization of parameter regimes where estimation is difficult. Section 3.4.1 presents the likelihood estimation technique and computations, and Section 3.4.2 defines pseudo-unidentifiability and characterizes circumstances where pseudo-unidentifiable parameters pose estimation challenges.

---

sufficiently large relative to α^*, in which case no points satisfy $f(x) = x$.

Let $g(x) = f(x) - x$, and consider finding the minimum of g by differentiation:

$$g'(x) = \alpha^* \exp\{\nu^* + \alpha^* x\} - 1 = 0 \quad \Longleftrightarrow \quad x = \frac{-\log\alpha^* - \nu^*}{\alpha^*}$$

Since f is convex, so is g, so this is a minimum. At the minimum, the difference is:

$$\min_{x\in\mathbb{R}} g(x) = g\left(\frac{-\log\alpha^* - \nu^*}{\alpha^*}\right) = \frac{\log\alpha^* + 1 + \nu^*}{\alpha^*}$$

Therefore, whenever $\nu^* > -\log\alpha^* - 1$, $g(x) > 0$ for every x, and in this case, $f(x) > x$.

3.4.1 Likelihood estimation

Poisson GVAR(D) models can be estimated efficiently by maximum likelihood using the classical technique for generalized linear models. The log-likelihood of a process observation $x = (x_0 \cdots x_T) \in \mathbb{R}^{M\times(T+1)}$ given parameters $B = (\nu\ A_1 \cdots A_D)'$, restated from Section 3.2.1, is:

$$\ell(B;x) = \sum_{t=1}^{T}\sum_{m=1}^{M}\left(x_{t,m}\left(\nu_m + \sum_{d=1}^{D} a'_{dm}x_{t-d}\right) - \exp\left\{\nu_m + \sum_{d=1}^{D} a'_{dm}x_{t-d}\right\}\right) + C$$

The maximum likelihood estimator is defined as:

$$\hat{B} = \operatorname{argmin}_B \{-\ell(B;x)\}$$

Computation by IRLS. Exchanging the order of summation in the log-likelihood yields a partitioning according to component. That is, if $b_m = (\nu_m\ a'_{1m} \cdots a'_{Dm})$ denotes the mth column of B, the log-likelihood can be written in the form $\ell(B;x) = \sum_{m=1}^{M} \ell_m(b_m;x)$ where the mth likelihood summand ℓ_m is:

$$\ell_m(b_m;x) = \sum_{T=1}^{T}\left(x_{t,m}\left(\nu_m + \sum_{d=1}^{D} a'_{dm}x_{t-d}\right) - \exp\left\{\nu_m + \sum_{d=1}^{D} a'_{dm}x_{t-d}\right\}\right)$$

The full likelihood ℓ is maximized in B exactly when each partition ℓ_m is maximized in b_m. In other words, the MLE $\hat{B}$ is equivalent to the concatenation of the MLEs $\hat{b}_m$, defined as:

$$\hat{b}_m = \operatorname{argmin}_{b_m} \{-\ell_m(b_m;x)\}$$

Each $\hat{b}_m$ can be computed by iteratively reweighted least squares (IRLS), which is a standard and efficient algorithm for computing MLEs for genreralized linear models. To see this, it suffices to express the estimation problem as a generalized linear model. Define the mth pseudo-response y_m and correpsonding pseudo-covariates Z_m as:

$$y_m = \begin{bmatrix} x_{T,m} \\ x_{T-1,m} \\ \vdots \\ x_{D,m} \end{bmatrix}, \quad Z_m = \left[\begin{array}{c|ccc|c|ccc} 1 & x_{T-1,1} & \cdots & x_{T-1,M} & \cdots & x_{T-D,1} & \cdots & x_{T-D,M} \\ 1 & x_{T-2,1} & \cdots & x_{T-2,M} & \cdots & x_{T-D-1,1} & \cdots & x_{T-D-1,M} \\ \vdots & \vdots & & \vdots & & \vdots & & \vdots \\ 1 & x_{D-1,1} & \cdots & x_{D-1,M} & \cdots & x_{0,1} & \cdots & x_{0,M} \end{array}\right]$$

Now let the mth parameter vector β_m be simply b_m:

$$\beta_m = \begin{bmatrix} \nu_m \\ \hdashline a_{1m1} \\ \vdots \\ a_{1mM} \\ \hdashline \vdots \\ \hdashline a_{Dm1} \\ \vdots \\ a_{DmM} \end{bmatrix}$$

Define the model:

$$y_{mj} \overset{indep.}{\sim} \text{Poisson}(\lambda_{mj}) \quad \text{where} \quad \log \lambda_{mj} = z'_{mj}\beta_m \,, \quad j = 1, \ldots, T - D$$

This is a Poisson GLM with a log link function. By simply exchanging notations it follows that:

$$\ell(\beta_m; y_m, Z_m) = \sum_{j=1}^{T-D} \left(y_{mj}(z'_{mj}\beta_m) - \exp\left\{ z'_{mj}\beta_m \right\} \right) = \ell(b'_m; x)$$

Consequently, $\hat{B}$ can be obtained by computing each column $\hat{b}_m$ separately via IRLS, *i.e.*, regressing y_m on Z_m. Details on the IRLS algorithm are provided in Appendix A.

3.4.2 Pseudo-unidentifiability

The parameters of any Poisson GVAR(D) process are identifiable, as shown in Section 3.2.1. However, there are circumstances under which the likelihood varies either minimally or not at all with the parameters. Under either of these two circumstances, parameters are difficult to estimate. This potential problem is not exactly an identifiability issue, but it is similar in nature. The parameters are identifiabile if $B \neq B^*$ implies that $\ell(B; x) \neq \ell(B^*; x)$ for some x. One problem that arises for certain parameter regimes of Poisson GVAR processes is that for some x that occur with at least moderate probability, $\ell(B; x) = \ell(B^*; x)$ but $B \neq B^*$. This issue is referred to as 'pseudo-unidentifiability'.

Consider the possibility that for some x, the log-likelihood ℓ is not a one-to-one function of B:

$$\ell(B; x) = \ell(B^*; x) \quad \text{and} \quad B \neq B^*$$

In this case, B is 'pseudo-unidentifiable' from x in the sense that it cannot be distinguished from B^* via the likelihood.

The notion requires some refinement. Pseudo-unidentifiability obviously occurs whenever $x = 0$, since then $\ell(B; 0) = -T \sum_m \exp\{\nu_m\}$ is a function only of ν and not $A_1, \ldots, A_D$. However, for many finite-length processes $X = (X_0 \cdots X_T)$, $P(X = 0)$ is negligibly small. Moreover, from a practical perspective one would not attempt estimation if $x = 0$, since a process model is frivolous for data with zero variance. On the other hand, process parameters can be such that particular components are unlikely to vary much or at all; for example, when some b_m are dominated by large negative parameters. Thus, pseudo-unidentifiability is a more realistic problem componentwise and for specific parameter regimes, rather than for the process as a whole.

By way of refinement, consider instead the possibility that the mth log-likelihood summand ℓ_m is not a one-to-one function of b_m, so that b_m is pseudo-unidentifiable from x if:

$$\ell_m(b_m; x) = \ell(b_m^*; x) \quad \text{and} \quad b_m \neq b_m^*$$

Now, this property holds on a subset Θ of the parameter space $\mathbb{R}^{1+DM}$ if it holds for every $b_m, b_m^* \in \Theta$. Further, $\ell_m(b_m; x) = \ell(b_m^*; x)$ for every b_m, b_m^* in a nontrivial subspace Θ just in case the mth component $x_m = (x_{0,m} \cdots x_{T,m}) = 0$. The latter event occurs with probability $P(X_m = 0)$. This motivates the following defintion.

Definition 3.11 *(ϵ-pseudo-unidentifiability). Say that b_m is ϵ-pseudo-unidentifiable*

on Θ *if* $P(X_m = 0) > \epsilon$*, or put differently, if:*

$$P\left(X_m \in \{x_m : \ell(b_m; x_m) = \ell(b_m^*; x_m) \quad \forall b_m, b_m^* \in \Theta\}\right) > \epsilon$$

Technically, every b_m is ϵ-pseudo-unidentifiable on every Θ for some $\epsilon > 0$. However, when ϵ is small, pseudo-unidentifiability poses no problems for estimation. Pseudo-unidentifiability is only a realistic problem for large ϵ and Θ.

Examples. Suppose $D = 1$ and $\|a_m^+\|_0 = 0$, so that $a_{mj} \leq 0$ for every $j = 1, \ldots, M$. Then, $X_{t,m}$ is stochastically dominated by a Poisson random variable with rate e^{ν_m}, so $P(X_{t,m} > 0) \leq 1 - \exp\{e^{\nu_m}\}$. As a result:

$$P(X_m = 0) = 1 - P\left(\bigcup_{t=1}^{T}\{X_{t,m} > 0\}\right) \geq 1 - T(1 - \exp\{e^{\nu_m}\})$$

If $\nu_m = -5$ and $T = 100$, the bound gives that $P(X_m = 0) \geq 0.328$, in which case b_m is 0.33-pseudo-unidentifiable on $\Theta = \{b_m \in \mathbb{R}^M : b_{mj} \leq 0\}$. If instead $\nu_m = -6$, b_m is 0.75-pseudo-unidentifiable on the same region. In these examples, negative-signed parameters are challenging to estimate.

In practice, a column b_m may be difficult to estimate due to ϵ-pseudo-unidentifiability if a high proportion of the mth component series x_m is zero. As the example illustrates, this arises when the baseline rate e^{ν_m} for $X_{t,m}$ is small; it can also arise when $X_{t,m}$ receives a strongly negative influence from a component with a high baseline rate. Furthermore, these two cases — large negative ν_m and large negative a'_{dm} — can be difficult to distinguish from one another.

3.5 Discussion

Findings. The chapter's main results are the development of a graphical stability condition for Poisson GVAR(1) processes (Section 3.3.1), the derivation of parameter constraints that ensure specific bounds on process moments (Section 3.3.2), and a characterization of parameter regimes in which estimation is challenging (Section 3.4.2). These results highlight the recognition that Poisson GVAR models only describe plausible data-generating processes under certain complex conditions, and constraints on the magnitudes of positive-signed parameters that depend on the graphical representation of such parameters provide one way to preserve this kind of plausibility. The discussion of pseudo-unidentifiability underscores the point that even plausible data-generating processes can be hard to estimate.

The results in the chapter constitute a first step towards understanding the probabilistic behavior of nonlinear Poisson GVAR processes. Analysis of the linear case is more straightforward, but the parametrization largely excludes negative mutual serial dependence Heinen et al. (2003). By contrast, mixed dependence is possible for nonlinear GVAR processes, but the stochastic process is less analytically tractable. Although it has been acknowledged that process stability is a difficulty for the nonlinear case (Mark et al., 2017), existing approaches address the challenge through data transformations (Mark et al., 2017; Pillow et al., 2008). The results here reveal that process stability is possible to guarantee without such transformations through appropriate constraints on both parameter magnitudes as well as graphical complexity.

Challenges encountered. The unconditional moments of Poisson GVAR processes are challenging to compute in general because the superexponentiation that arises from the log link function does not lead to a straightforward recursion that generalizes beyond one time step. The marginal distributions are no easier to calculate since they require an analytic solution to the summation:

$$\sum_{x_{t,j}=0}^{\infty} \exp\left\{\sum_t \sum_m \left(x_{t,m}\left(\nu_m + \sum_d a'_{dm} x_{t-d}\right) - \exp\left\{\nu_m + \sum_d a'_{dm} x_{t-d}\right\}\right)\right\}$$

These challenges are traversed in this chapter only under certain limitations. The use of a graphical constraint to establish stability for a subclass of order 1 processes is based on two observations. First, when no positive influences are present on a component it is easy to specify a stochastically dominating Poisson random variable for that component that doesn't depend on the process history. Second, the mean of a once-exponentiated Poisson random variable can be calculated using the Poisson moment generating function. Together, these properties make it feasible to calculate or bound unconditional moments when the positive influences don't induce conditional dependence among multiple components and spanning more than a single time step.

Future work. It remains a likelihood that the graphical constraint developed in the chapter can be relaxed. Further work is needed to identify constraints on positive-signed parameters that provide sufficient stability conditions and constraints that ensure specific moment bounds for longer paths in G^+. Further, it may be possible to mitigate unstable structures in G^+ through complementary structure in G^-, thereby preserving stability though interactions between positive and negative serial

dependence. Resolving these subjects would complete the understanding of moment behavior across the full parameter space. Additionally, it would be useful to extend this analysis by determining ergodicity conditions for Poisson GVAR(D) processes, which would allow asymptotic analysis of the MLE for the model class using existing results (Fahrmeir, 1987).

Chapter 4: Sparse estimation of Poisson GVAR models

4.1 Introduction

Chapter 2 presented an empirical method that improves estimation of sparse VAR process parameters relative to L_1-constrained maximum likelihood (LASSO). The empirical method is a two-step procedure that first estimates candidate parameter support sets and then selects from among the candidates and returns a final estimate. Although the empirical performance of the method is promising, the relative contributions of methodological features employed in each step are not well understood. That is, existing experimental work only establishes that the particular combination of support set estimation and support set selection techniques used in the method confers an improvement over LASSO estimation, but the marginal effects on estimation behavior associated with each feature and those associated with other implementation details in the method are not known. In other words, no constructive account of the method is available that demonstrates the utility of each part. The existing study of the algorithm leaves open the possibility that variants or simpler versions of the method may perform equally well or better.

Chapter 3 generalizes the VAR model to Poisson GVAR models, motivated by seeking more natural descriptions of count vector time series. The same sparse network estimation problem considered in Chapter 2 arises in connection with count data

for certain applications. While a generalization of the VAR model has been used in such applications, the performance of different estimators for process parameters in the generalization has not been studied carefully, nor has the empirical method been extended to the generalized model in order to cover a broader range of application contexts.

Purpose and contributions. There are two clear gaps in understanding that emerge from considering prior work: the effects of individual features that are combined in the empirical method, relative to benchmarks, are not known; and the method in its current form cannot cover the full range of applications in network estimation. This chapter addresses these gaps by providing a systematic study of the individual features of the method of Chapter 2 in the context of estimating sparse Poisson GVAR models. The chapter begins by dividing the empirical method into pieces and developing benchmark methods for each piece. The resulting possible combinations are used to generate a set of variant methods, and these variants are compared to one another empirically on synthetic data generated by sparse Poisson GVAR(1) processes to capture the effects of specific design choices on selection and estimation performance. Based on the optimal combination in the empirical comparison, a number of modifications are proposed to the original method. In doing so, the following contributions are made:

(i) development of novel benchmark methods for support estimation and support selection in sparse time series models

(ii) generalization of the empirical method from Chapter 2 to Poisson GVAR models

(iii) proposal of novel modifications to the empirical method from Chapter 2

(iv) development of a simulation framework for constructing stable Poisson GVAR(1) processes

(v) proposal of a novel 'recoverability' statistic to quantify estimation difficulty for process parameters

(vi) experimental assessment of selection and estimation performance for a collection of methods under varying parametric sparsity, parameter dimensions, and data quantities

Chapter organization. The chapter is organized as follows. First, Section 4.2 reviews the estimation method from Chapter 2 and develops variant methods. This proceeds in several steps: subsection 4.2.1 divides the method into constituent features; subsection 4.2.2 posits benchmark design choices for each feature; subsection 4.2.3 enumerates the possible combinations of design choices; and subsection 4.2.4 identifies four specific methodological variants of interest based on initial comparisons. Next, Section 4.3 presents a simulation study of the methods. This also proceeds in several steps: subsection 4.3.1 develops a simulation framework for constructing stable Poisson GVAR(1) processes, proposes a 'recoverability' statistic to quantify estimation difficulty, and presents a method for sampling from the Poisson GVAR(1) parameter spaces of arbitrary dimension and sparsity; subsection 4.3.2 presents a factorial simulation study design to assess Poisson GVAR(1) estimation performance under varying sparsity levels and process dimensions; and subsection 4.3.3 reports

the results of the study carried out with the four methodological variants of interest developed in the previous section. Last, Section 4.4 concludes the chapter with a brief discussion of the main contributions of this work and possible extensions.

4.2 Methodological variants on UoI

The main aim of this section is to produce family of variants based on the method presented in Chapter 2 in which certain pairs of variants in the family differ by exchanging a single methodological design choice for its benchmark alternative while keeping other design choices fixed. Comparing performance among such pairs, which will be the subject of a later section, allows one to isolate the effect of each design choice relative to its benchmark both marginally and conditionally on other design choices, thereby clearly depicting the relative contributions of methodological features.

In service of this aim, the section is divided into four subsections. Subsection 4.2.1 reintroduces the algorithmic framework presented in Chapter 2 in more general terms. Subsection 4.2.2 identifies design choices by contrasting the framework with a benchmark method and identifying differences. Subsection 4.2.3 articulates a collection of methodological variants by enumerating the possible combinations of design choices. Lastly, an initial comparison that narrows the full set of variants to a subset of particular interest is given in subsection 4.2.4.

This portion of work identifies four variants of interest for further study: the combinations of two possible support estimation methods with two possible support

selection methods. In doing so, the variants that are eliminated from consideration reveal design choices that are detrimental to performance.

4.2.1 Algorithmic framework

To begin with, a brief review of the two-step UoI-VAR method presented in Chapter 2 is given, and from this a general description of the algorithmic framework is derived. The UoI-VAR method accomplishes four high-level tasks:

(i) estimation of candidate support sets ('intersection' or first step);

(ii) selection of well-performing supports ('union' or second step);

(iii) combination of the latter to obtain a final support set ('union' or second step); and

(iv) estimation on the final support set ('union' or second step).

These are articulated in the context of estimating, under a sparsity constraint, the Gaussian $VAR(D)$ model for an M-dimensional vector process:

$$X_t = \nu + \sum_{d=1}^{D} A_d X_{t-d} + \epsilon_t \ , \quad \epsilon_t \overset{iid}{\sim} N_M(0, \Sigma)$$

Parameter estimation from data $\{x_t\}_{t=0}^{T}$ (the lower case distinguishes observed vectors x_t in a dataset from random vectors X_t) is articulated in terms of estimating $B = (\nu \ A_1 \cdots A_D)'$. The loss function is denoted by $-\ell(B \ ; \ x_0, \ldots, x_T)$.

Intersection step. The first step of the UoI-VAR algorithm estimates candidate support sets (accomplishing task (i)) by performing an intersection operation on support sets estimated from bootstrap samples of the data. The intersection step begins by forming B_1 bootstrap samples from the data. Then, K LASSO estimates are computed for each bootstrap sample using the regularization path $\lambda_1, \ldots, \lambda_K$. If the bth bootstrap sample is denoted by $\{x_t^*\}_{t=0}^T$, these estimates are:

$$\tilde{B}_{bk} = \text{argmin}_B \left\{ -\ell(B \,;\, x_0^*, \ldots, x_T^*) + \lambda_k P(B) \right\} \,, \quad k = 1, \ldots, K$$

For LASSO estimation, the L_1 penalty $P(B) = \sum_d \sum_i \|a_{di}^T\|_1$ is applied to the rows a_{di}^T of the matrices A_d. The values of the regularization parameter λ_k control the strength of the penalty and consequently the strength of the sparsity constraint applied to $A_1, \ldots, A_D$. Each estimate $\tilde{B}_{bk}$ is associated with a support set:

$$S_{bk} = \{(i,j) : \tilde{b}_{bkij} \neq 0\} \,, \quad k = 1, \ldots, K \,,\; b = 1, \ldots, B_1$$

The index k is generally arranged so that $S_{b1}, \ldots, S_{bK}$ are increasing sets for each $b = 1, \ldots, B_1$. Finally, an intersection of the support sets is taken across bootstrap samples (index b) for each λ_k:

$$S_k = \bigcap_{b=1}^{B_1} S_{bk} \,, \quad k = 1, \ldots, K$$

The series of computations described above is summarized as follows:

for $b = 1$ to B_1 **do**
  draw bootstrap sample $\{x_t^*\}_{t=0}^T$
  for $k = 1$ to K **do**
    estimate $\tilde{B}_{bk} \leftarrow \arg\min_B \{-\ell(B\,;\, x_0^*, \ldots x_T^*) + \lambda_k P(B)\}$
    compute support $S_{bk} \leftarrow \left\{(i,j) : \tilde{b}_{bkij} \neq 0\right\}$
  end for
end for
(output) aggregated supports $S_k \leftarrow \bigcap_{b=1}^{B_1} S_{bk}\,, \quad k = 1, \ldots, K$

Each outputted support set S_k comprises parameters that have a high probability of being selected with the sparsity constraint corresponding to λ_k and under resampling of the data. Relative to a single collection of LASSO estimates computed from the original data, the sets $S_1, \ldots, S_K$ are sparser; however, they are not necessarily nested due to variation in the sparsity pattern of estimates $\tilde{B}_{bk}$ across bootstrap samples for a fixed k.

Union step. The second step of the UoI-VAR algorithm accomplishes tasks (ii)-(iv) in several stages: first, a number of 'good' supports are chosen from $S_1, \ldots, S_K$ based on forecasting accuracy (accomplishing task (ii)); next, the 'good' supports are combined by a union operation to obtain a final support set (accomplishing task(iii)); and last, an estimate is computed on the final support set by averaging estimates computed on bootstrap samples (accomplishing task(iv)).

The first stage selects 'good' supports from the collection $S_1, \ldots, S_K$ based on forecasting accuracy. This begins by forming a collection of B_2 pairs of bootstrap samples. Each support set S_k characterizes a parameter subspace $\mathcal{B}_k = \{B \in \mathbb{R}^{M \times D(D+1)} : \{(i,j) : b_{ij} \neq 0\} = S_k\}$. Now, unpenalized estimates are computed in each of the K subspaces on one of the bootstrap samples in each pair. If $\left\{x_t^{*(1)}\right\}_{t=0}^T$ denotes the first bootstrap

sample in the bth pair, these estimates are:

$$\tilde{B}_{bk} = \text{argmin}_{B \epsilon \mathcal{B}_k} \left\{ -\ell(B \; ; \; x_0^{*(1)}, \ldots, x_T^{*(1)}) \right\} \; , \quad k = 1, \ldots, K \; , \; b = 1, \ldots, B_2$$

Next, for each estimate $\tilde{B}_{bk}$, forecasting errors are computed on the second bootstrap sample in the corresponding (bth) pair. Denoting the one-step forecast of x_t based on B by $\hat{x}_t(B; x_{t-1}, \ldots, x_{t-D})$, if $\left\{ x_t^{*(2)} \right\}_{t=0}^{T}$ is the second bootstrap sample in the bth pair, then the forecast errors are:

$$f_{bk} = \sum_t \| x_t^{*(2)} - \hat{x}_t(\tilde{B}_{bk}; x_{t-1}^{*(2)}, \ldots, x_{t-D}^{*(2)}) \|_2 \; , \quad k = 1, \ldots, K, \; , \; b = 1, \ldots, B_2$$

Then, for each pair of bootstrap samples $b = 1, \ldots, B_2$, the support set associated with the minimum forecasting error f_{bk} is selected:

$$S^{*b} = S_{k^{*b}} \; , \text{ where } k^{*b} = \text{argmin}_k \, f_{bk} \; , \quad b = 1, \ldots, B_2$$

The sets $S^{*1}, \ldots, S^{*B_2}$ comprise the 'good' supports selected from the initial collection $S_1, \ldots, S_K$.

The second stage of the union step combines the selected support sets $S^{*1}, \ldots, S^{*B_2}$ by a union operation over the sparsest $(100 \times \gamma)\%$ of the sets. That is, if the size of the bth selected support set is $|S^{*b}|$, $Q = \{b : |S^{*b}| \leq \gamma \cdot \text{quantile}(|S^{*1}|, \ldots, |S^{*B_2}|)\}$ gives the set of indices corresponding to the sparsest approximately $(100 \times \gamma)\%$ of the

selected supports, and the final support set is:

$$S^* = \bigcup_{b \in Q} S^{*b}$$

This is the (thresholded) collection of parameter subsets selected with high probability that are most predictive under resampling of the data. Thresholding the average by sparsity is an ad-hoc way of evading an overly dense final support set due to one or two dense S^{*b}, which arise rather often in practice.

Lastly, the remainder of the union step averages the unpenalized estimates used to compute each S^{*b} over the indices in Q:

$$\hat{B} = \sum_{b \in Q} \tilde{B}_{k^{*b}}$$

The full series of computations in the union step is summarized as follows:

for $b = 1$ to B_2 **do**
  draw bootstrap samples $\left\{x_t^{*(1)}\right\}_{t=0}^{T}$, $\left\{x_t^{*(2)}\right\}_{t=0}^{T}$
  for $k = 1$ to K **do**
    subspace $\mathcal{B}_k \leftarrow \{B \in \mathbb{R}^{M \times D(D+1)} : \{(i,j) : b_{ij} \neq 0\} = S_k\}$
    unpenalized estimate $\tilde{B}_{bk} \leftarrow \arg\min_{B \in \mathcal{B}_k} \left\{-\ell(B\ ;\ x_0^{*(1)}, \ldots, x_T^{*(1)})\right\}$
    forecast error $f_{bk} \leftarrow \sum_t \|x_t^{*(2)} - \hat{x}_t(\tilde{B}_k; x_{t-1}^{*(2)}, x_{t-D}^{*(2)})\|_2$
  end for
  minimum error $k^{*b} \leftarrow \text{argmin}_k\, f_{bk}$
  selected support $S^{*b} \leftarrow S_{k^{*b}}$
end for
sparsest selected supports $Q \leftarrow \{b : |S^{*b}| \leq \gamma \cdot \text{quantile}(|S^{*1}|, \ldots, |S^{*B_2}|)\}$
final support $S^* \leftarrow \bigcup_{b \in Q} S^{*b}$
(output) final estimate $\hat{B} \leftarrow \sum_{b \in Q} \tilde{B}_{k^{*b}}$

Algorithmic framework. With the foregoing review in place, the algorithmic framework for the method can now be described in greater generality. In its entirety, the method presented above relies on a relatively limited number of operations performed on the data (or bootstrap samples of the data), support sets, estimates, and tuning parameters. These operations are: resampling; estimation of support sets; estimation of parameters constrained to a support set; error metric calculation; and thresholded combination of support sets. By omitting the details of how these operations are performed, the entire algorithm can be described at a high level quite compactly. To that end, these operations are given notations and brief descriptions in Table 4.1.

The functions in Table 4.1 can be composed to describe chains of computations compactly. For example, $\hat{\beta}(\text{train}; \hat{S}(\text{data}; \lambda))$ describes first computing a support set estimated from the full data 'data' with tuning parameter λ and then computing an estimate from training data 'train' on the support set from the first computation. To extend the example, this composition could be further embedded in e to describe next computing an error metric associated with those estimates: $e(\hat{\beta}(\text{train}, \hat{S}(\text{data}, \lambda)))$.

A description of the algorithmic framework in this new notation is:

for $b = 1$ to B_2 **do**
  $(\text{train}_b, \text{test}_b) \leftarrow \text{resample}(\text{data})$
  $e_b^\lambda \leftarrow e\left(\text{test}_b, \hat{\beta}\left(\text{train}_b, \hat{S}(\text{data}, \lambda)\right)\right),\ \forall \lambda \in \Lambda$
  $\lambda_b^* \leftarrow \text{argmin}_\lambda\, e_b^\lambda$
  $S_b \leftarrow \hat{S}(\text{train}_b, \lambda_b^*)$
end for
$S^* \leftarrow t\left(\{S_b\}_{b=1}^{B_2}, \gamma\right)$
$\hat{B} = (\gamma B_2)^{-1} \sum_{b:|S_b| \le |S^*|} \hat{\beta}\left(\text{train}_b, \hat{S}(\text{data}, \lambda_b^*)\right)$

Function	Desription	Example
resample($\cdot$)	produces test data and training data	resample(data) = (train, test)
$\hat{S}(\cdot\,;\lambda)$	produces support set with selection controlled by tuning parameter(s) λ	$\hat{S}(\text{data}; 0.012) = \{1, 3\}$
$\hat{\beta}(\cdot\,; S)$	produces estimates constrained to support set S	$\hat{\beta}(\text{data}; \{1, 3\}) = (6.1\ 00.76\ 0 \cdots 0)^T$
$e(\cdot\,;\beta)$	produces the value of an error metric for parameter value β	$e(\text{data}; (6.1\ 0\ 0.76\ 0 \cdots 0)^T) = 153.8$
$t\left(\{S_j\}_{j=1}^J; \gamma\right)$	produces a single support set from a collection of supports $\{S_j\}_{j=1}^J$ and threshold parameter γ	$t\left(\{\{1\}, \{1, 3\}\}; 0.8\right) = \{1\}$

Table 4.1: Notations for operations figuring in the UoI framework. These functions are intended to create a succinct vocabulary for classes of key operations (*e.g.*, constrained estimation) rather than specific methods for performing operations (*e.g.*, constrained OLS estimation).

To specify the exact UoI-VAR algorithm from this description, simply define $\hat{S}(\cdot\,;\cdot)$ as the intersection step, $e(\cdot\,;\cdot)$ as one-step forecasting error, and $t(\cdot\,;\cdot)$ as the thresholding operation presented above.

Accordingly, the intersection step is simply a means of generating candidate models, but is a relatively inessential detail in the algorithmic framework. In other words, the intersection step amounts to a specific choice of the function $\hat{S}$, but could easily be exchanged for another support estimation method without altering the algorithm as written above. While this methodological choice is not inconsequential, the algorithm itself serves to select a final model, regardless of the means of generating candidate

models. The overall method can therefore be viewed as a combination of a support estimation method (a choice of $\hat{S}$) with a support selection method (the algorithmic framework).

4.2.2 Methodological design choices

In light of the foregoing, two central features figure in the design of the method in Chapter 2. First, there is a support selection method: this is the general algorithmic framework articulated above. Second, there is a support estimation method $\hat{S}$ used by the support selection method: this is the intersection step. In addition, there is an implementation detail in the framework above aimed at saving computation time that constitutes a third important feature. This subsection frames each feature as a choice made relative to a particular benchmark method.

Choice 1: support selection. The distinctive feature of the algorithmic framework above – as a support selection method – is that it selects multiple candidate models using an optimality criterion and then combines the models. This model aggregation strategy is a novel method that stands in stark contrast to selecting a single candidate model. To highlight this contrast, a benchmark method that selects supports by cross validation is shown below as Algorithm 4.1, and compared directly with the model aggregation method, restated as Algorithm 4.2.

The benchmark method generates J pairs of training and test data from the original dataset. On each pair, supports are first estimated from the training set for each value of λ, and then parameters constrained to each support are estimated from

Algorithm 4.1 Support set selection by cross validation.

Require:
data
regularization path Λ
for $j = 1$ to J **do**
$(\text{train}_j, \text{test}_j) \leftarrow \text{resample}(\text{data})$
$e_j^\lambda \leftarrow e\left(\text{test}_j, \hat{\beta}\left(\text{train}_j, \hat{S}(\text{train}_j, \lambda)\right)\right),\ \forall \lambda \in \Lambda$
end for
$\lambda^* \leftarrow \text{argmin}_\lambda\, \bar{e}^\lambda$
Ensure: estimate $\hat{\beta}\left(\text{data}, \hat{S}(\text{data}, \lambda^*)\right)$

the training set. Errors are evaluated for each estimate (and consequently support set) on the test set. This produces one error per value of λ per pair of training and test data. The value of λ with the lowest average error across the J pairs is then selected as optimal. Using this optimal value, a final support set and parameters are estimated on the original data.

Algorithm 4.2 Support set selection by model aggregation.

Require:
data
regularization path Λ
tuning parameter γ
for $j = 1$ to J **do**
$(\text{train}_j, \text{test}_j) \leftarrow \text{resample}(\text{data})$
$e_j^\lambda \leftarrow e\left(\text{test}_j, \hat{\beta}\left(\text{train}_j, \hat{S}(\text{train}_j, \lambda)\right)\right),\ \forall \lambda \in \Lambda$
$\lambda_j^* \leftarrow \text{argmin}_\lambda\, e_j^\lambda$
$S_j \leftarrow \hat{S}(\text{train}_j, \lambda_j^*)$
end for
$S^* \leftarrow t\left(\{S_j\}_{j=1}^J, \gamma\right)$
Ensure: estimate $\hat{\beta}\left(\text{data}, S^*\right)$

The model aggregation method is the same up to the point of computing errors associated with each value of λ for each pair of training and test data. However, it differs from that point forward: instead of averaging the errors across the pairs, an optimal λ is chosen for each pair and the corresponding estimated support set is set aside. This produces a collection of J optimal support sets. The support sets are then aggregated by a union operation, and a single estimate is computed on this final support.[1]

Choice 2: support estimation. In the method of Chapter 2, the intersection step — bootstrap aggregation of LASSO supports as described in Section 4.2.1 — is the method of support estimation. The bootstrap aggregation procedure has already been established in the context of regression, where it is referred to as the bootstrapped LASSO or 'BoLASSO' (Bach (2008)). The BoLASSO stabilizes the selection behavior of LASSO estimates by limiting the support set associated with a given regularization parameter λ to parameters with a high probability of being selected under resampling of the data. Thus, the natural benchmark alternative to the intersection step is to use supports from a single LASSO estimate on the full dataset. These alternatives will be distinguished using the notations in Table 4.2.

Choice 3: conditioning. There is a subtle implementation difference between Algorithm 4.2 and the framework presented at the end of Section 4.2.1: the argument of $\hat{S}(\,\cdot\,, \lambda)$. In Algorithm 4.2, the argument is train_j, meaning that new support

[1] This replaces an average of several estimates as in UoI-VAR and the framework written in Section 4.2.1.

Function	Description
$\hat{S}^{(1)}(\,\cdot\,, \lambda)$	produces support set for LASSO estimate with regularization parameter λ
$\hat{S}^{(2)}(\,\cdot\,, \lambda)$	produces bootstrap aggregated LASSO support set for regularization parameter λ (intersection step with some fixed B_1)

Table 4.2: Notations for support estimation methods.

sets are computed for each training set. In the framework as stated previously, the argument is data, meaning that support sets are computed just once on the full dataset, and the selected supports are always one of this fixed collection of support sets. In other words, the original method conditions the model aggregation procedure on a fixed collection of candidate supports. This conditioning strategy reduces the computation time by performing $\hat{S}$ just once instead of $J + 1$ times, but also limits the flexibility of the procedure.

Other choices. Aside from the support estimation method, support selection method, and conditioning, there are a number of other relatively more minor implementation choices. Among these are the method of resampling, the method of thresholding implemented in model aggregation, the method of parameter estimation, and the error metric. In other words, the precise definitions of each of the operations in Table 4.1. For each of these choices, a single method is fixed based on empirical experimentation. First, while a range of time series bootstrap methods exist (Bühlmann and Künsch (1999); Kreiss and Lahiri (2012); Kunsch et al. (1989); Lahiri et al. (1999); Liu and Singh (1992)), a blockwise jackknife is used. This was found to produce more stable

results than other block bootstrap approaches, including that utilized in Chapter 2. Second, thresholding based on position frequency was found to produce more stable results than thresholding based on sparsity, and as a result the threshold method is fixed as:

$$t\left(\{S_j\}_{j=1}^J, \gamma\right) = \left\{ i \in \bigcup_j S_j : \sum_j \mathbb{1}\{i \in S_j\} \geq \gamma J \right\}$$

Finally, appropriate choices of support-constrained parameter estimation and error metric are somewhat problem-dependent, but in the contexts considered in this chapter, maximum likelihood and deviance are used.

Summary. Three key methodological design choices have been identified above: the method of support estimation, the method of support selection, and conditioning the support selection method on a fixed collection of support sets. Further, there are a few incidental decisions about how to implement specific operations, some of which have been replaced with alternatives that produce more stable behavior than the original UoI-VAR method. The design choices are summarized in Table 4.3. The

Choice	**Options**
Support estimation method	LASSO supports ($\hat{S} \equiv \hat{S}^{(1)}$) or intersection step ($\hat{S} \equiv \hat{S}^{(2)}$)
Support selection method	cross validation (Algorithm 4.1) or model aggregation (Algorithm 4.2)
Conditional support selection	Constrain estimates on training data to $\hat{S}(\text{train}_j; \lambda)$ or $\hat{S}(\text{data}; \lambda)$ when evaluating errors

Table 4.3: Summary of key choices in method design and options for each choice.

particular options that appear in UoI-VAR are aimed at specific improvements relative to the corresponding benchmark option: support estimation by support aggregation and support estimation by model aggregation are each intended to improve support recovery relative to LASSO support estimation and cross validation, respectively; and conditioning is intended to lessen computation time relative to the unconditioned procedures.

4.2.3 Methodological variants

The possible combinations of the options for support estimation, support selection, and conditioning status established in the previous section generate a family of methodological variants whose differences systematically capture the effect of each design choice. These possible combinations are eight in number. However, the model aggregation method involves a threshold parameter (γ) that itself requires tuning. A cross validation strategy is proposed here as Algorithm 4.3: the entire model aggregation method with a range of possible tuning parameters is repeated for a number of training datasets derived from the full data by resampling, and errors are evaluated for each tuning parameter on test datasets. As with the conditioning design choice, there is the possibility for each methodological variant involving model aggregation to condition Algorithm 4.3 on a fixed collection of estimates $\text{model.aggregation}(\text{data}; \gamma)$. In other words, conditioning can be applied at both an inner (model aggregation) and outer (threshold parameter tuning) level for each of the four variants that perform support selection by model aggregation.

Algorithm 4.3 cross validation algorithm for tuning model aggregation threshold.

Require:
data
threshold parameter path Γ
for $i = 1$ to I **do**
$(\text{train}_i, \text{test}_i) \leftarrow \text{resample}(\text{data})$
$e_i^{\alpha} \leftarrow e\left(\text{test}_i, \text{model.aggregation}(\text{train}_i; \gamma)\right),\ \forall \gamma \in \Gamma$
end for
$\alpha^* \leftarrow \text{argmin}_{\gamma}\, \bar{e}^{\gamma}$
Ensure: estimate $\text{model.aggregation}(\text{data}, \gamma^*)$

In sum, all possible combinations of these choices yield twelve algorithmic variants:

1. cross validation with $\hat{S}^{(1)}(\text{train}_j, \lambda)$
2. cross validation with $\hat{S}^{(2)}(\text{train}_j, \lambda)$
3. cross validation with $\hat{S}^{(1)}(\text{data}, \lambda)$
4. cross validation with $\hat{S}^{(2)}(\text{data}, \lambda)$
5. Model aggregation with $\hat{S}^{(1)}(\text{train}_j, \lambda)$ and $\text{model.aggregation}(\text{train}_i, \gamma)$
6. Model aggregation with $\hat{S}^{(2)}(\text{train}_j, \lambda)$ and $\text{model.aggregation}(\text{train}_i, \gamma)$
7. Model aggregation with $\hat{S}^{(1)}(\text{data}, \lambda)$ and $\text{model.aggregation}(\text{train}_i, \gamma)$
8. Model aggregation with $\hat{S}^{(2)}(\text{data}, \lambda)$ and $\text{model.aggregation}(\text{train}_i, \gamma)$
9. Model aggregation with $\hat{S}^{(1)}(\text{train}_j, \lambda)$ and $\text{model.aggregation}(\text{data}, \gamma)$
10. Model aggregation with $\hat{S}^{(2)}(\text{train}_j, \lambda)$ and $\text{model.aggregation}(\text{data}, \gamma)$

11. Model aggregation with $\hat{S}^{(1)}(\text{data}, \lambda)$ and model.aggregation(data, γ)

12. Model aggregation with $\hat{S}^{(2)}(\text{data}, \lambda)$ and model.aggregation(data, γ)

4.2.4 Initial comparison: effect of conditioning

The option to condition procedures by replacing training data with the full data in $\hat{S}(\,\cdot\,, \lambda)$ and model.aggregation($\,\cdot\,, \gamma$) lessens run time by reducing the number of times those operations are performed. However, this conditioning may affect the behavior of the estimators. While the computational advantage of certain variants is attractive, they should not be adopted at too high a cost of estimation performance.

The effect of each conditioning strategy can be figured by comparing estimation performance empirically between the pairs of variants that differ only in the replacement of training data with full data in a single location in the algorithm. Table 4.4 compares average computation times and selection errors between appropriate pairs of variants in estimating a single 5×5 parameter matrix A with 3 nonzero parameters among the 25 matrix positions from Poisson GVAR(1) data realizations. Estimates were computed for each variant on ten data realizations of length 500. The table indicates that the average runtimes for the 'conditioned' variants are faster than the other variant in the pair by up to 50%, but the average false positive counts for the 'conditioned' variants exceed their complementary variants by at least a factor of 2.

The design choices that involve conditioning have a detrimental impact on selection behavior. These are therefore poor design choices, and can be eliminated from further study. In other words, every variant identified under the V2 column

Variants		**Ratio Comparisons (V2:V1)**	
V1	V2	Time ratio (sec)	FP ratio (count)
1	3	0.79:1	23.00:1
2	4	0.56:1	3.88:1
5	7	0.71:1	10.00:1
6	8	0.58:1	41.00:1
5	9	0.80:1	14.00:1
6	10	0.60:1	2.90:0
7	11	0.94:1	2.75:1
8	12	1.05:1	3.14:1

Table 4.4: Initial comparison of methods to assess the effect of conditioning. Comparisons are made between variants differing in only the input data to the operations $\hat{S}$ and model.aggregation and are reported on the basis of the ratio of average computation times and the ratio of average false positive rates (errors by inclusion, or estimated support set elements that are not in the true support set). The variant that uses training data is always reported as variant 1, and the variant that uses full data is always reported as variant 2.

is dropped. The remaining variants are four in number: variants 1, 2, 5, and 6. These capture the design choices pertaining to support estimation and support selection. Specifically, they are the factorial combinations of support estimation by either LASSO supports or support aggregation, and support selection by either cross validation or model aggregation. These remaining methods of interest are summarized in Table 4.5.

	LASSO supports	Support aggregation
cross validation	Variant 1	Variant 2
Model aggregation	Variant 5	Variant 6

Table 4.5: Method variants of interest; factorial combinations of two support estimation methods with two support selection methods.

The subsequent sections empirically study the impact of each design choice on estimation performance by comparing these four variants in a simulation study.

4.3 Simulation study

The four method variants — the possible combinations of support estimation and support selection methods — identified in Section 4.2 represent precise comparisons of the central features of the method in Chapter 2 with benchmark methods. This section studies the performance of each variant empirically to capture the effect of each feature and identify an optimal combination. Both high selection accuracy (recovery of the true parameter support) and low bias are desirable properties for the methods. In addition, maintaining high accuracy and low bias across a range of problem scenarios is desirable: the effect of varying the parameter dimensions and sparsity on method performance is also of interest.

The empirical study is conducted in the context of estimating Poisson GVAR(1) models, and can be viewed from two perspectives. From one point of view, the effects of methodological features are the subject of the study, and the Poisson GVAR(1) model is just a particular context that represents a difficult class of problems in which these effects will be most pronounced. From another point of view, the development of an estimation method for sparse Poisson GVAR(1) models is the subject of the study, precisely because the model class comprises difficult estimation problems that challenge existing methods. In truth, both aims are being served.

The section begins with an extended discussion of fixing Poisson GVAR(1) process

parameters under a specified dimension and sparsity level, citing challenges that arise for this particular family of stochastic processes. Due to the existence of unstable parameter regimes and nearly unidentifiable parameter regimes, care is required in constructing process parameters; Section 4.3.1 presents an algorithm for generating stable parameters and a recoverability statistic for characterizing the difficulty of estimating a particular set of parameters. Section 4.3.2 then gives the design of the simulation study, followed by a presentation of key results in Section 4.3.3.

4.3.1 Parameter generation and parameter recoverability

Poisson GVAR(1) process parameters cannot be drawn at random due to regimes in the parameter space that are unstable (unbounded means and/or variances) and regimes that are nearly unidentifiable (data realizations contain extremely little information about the parameters). Here a method of sampling from the parameter space for an M-dimensional process at a particular sparsity level is given. The method uses an algorithm for generating stable parameters at a given sparsity level to propose parameters and a recoverability statistic to determine whether to accept proposals. The parameter generation algorithm is presented first, followed by the recoverability statistic.

Parameter generation. The parameter generation task consists in constructing an

intercept vector ν and a matrix A such that the process

$$X_{t,m}|X_{t-1} \overset{indep.}{\sim} \text{Poisson}(\lambda_{t,m}), \quad m = 1, \ldots, M,\ t \in \mathbb{Z}$$
$$\log \lambda_t = \nu + AX_{t-1}$$

is stable. It is possible to guarantee stability using only the sparsity pattern of A by placing constraints on the positions of positive-signed entries (Proposition 3.7). Algorithm 4.4 a simple procedure for generating an A with sparsity level s ($100 \times s\%$ of entries are nonzero) that ensures a stable M-dimensional Poisson GVAR(1) process for any intercept ν.

This procedure draws nonzero parameter magnitudes from a uniform distribution, allocates a sign to each parameter, fixes the locations of positive-signed nonzero parameters (the set S^+) and the locations of negative-signed nonzero parameters (the set S^-), and allocates the parameters to the selected locations. The locations S^+ are chosen so that the graph of the positive part of A has no paths exceeding length 1. This constraint is imposed by partitioning the vector components into a 'receiver' set R and an 'influencer' set I and drawing the locations in S^+ with replacement from $R \times I$. This only constructs edges in the graph of the positive part of A that map from I to R, and therefore cannot produce positive-signed paths exceeding length 1. Any such A satisfies stability constraints, so the algorithm guarantees that A is stable. This property holds for any p and θ, so the choice of inputs is largely incidental.

Parameter recoverability. Certain parameters obtainable by Algorithm 4.4 are difficult to estimate, and others are too easy. For example, the following parameters

Algorithm 4.4 Poisson GVAR(1) parameter generation.

Require:
process dimension M
sparsity level s
maximum absolute parameter magnitude θ
expected proportion of positive-signed parameters p
define receiver set $R \leftarrow \{1, \ldots, \text{floor}(M/2)\}$
define influencer set $I \leftarrow \{\text{ceiling}(M/2), \ldots, M\}$
define $S \leftarrow \{(i,j) : i \in R, j \in I\}$
for $k = 1$ to sM^2 **do**

$$\alpha_k \leftarrow \begin{cases} \text{draw from } U(0,\theta) & \text{w.p. } p \\ \text{draw from } U(-\theta,0) & \text{w.p. } 1-p \end{cases}$$

end for
$n_{NZ} \leftarrow |\{k : \alpha_k > 0\}|$
$S^+ \leftarrow$ sample n_{NZ} times without replacement from S
$S^- \leftarrow$ sample $sM^2 - n_{NZ}$ times without replacement from $\{1, \ldots, M\}^2 \setminus S^+$
for $k = 1$ to $|S_+^*|$ **do**
  if $\alpha_k > 0$ **then**
    $A_{s_k^+} \leftarrow \alpha_k$
  else
    $A_{s_k^-} \leftarrow \alpha_k$
  end if
end for
Ensure: A

are challenging to recover:

$$A = \begin{bmatrix} 0 & 0.1 & 0 \\ 0 & 0 & 0.1 \\ -0.1 & 0 & 0 \end{bmatrix} \quad \nu = \begin{bmatrix} -2 \\ -2 \\ -2 \end{bmatrix}$$

The difficulty is that the baseline component rates are $e^{-2} = 0.135$, so data realizations will be mostly zeros, and A modulates the rates by weak multiplicative factors that are hard to detect even with large quantities of data.[2] By contrast, the following parameters are easy to recover:

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad \nu = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

The baseline rates are $e^0 = 1$, and the single multiplicative factor in A is large enough that the corresponding changes in mean are easy to detect with little data.[3] One would expect that no estimation method would recover the first example parameters

[2]Whenever a 1 appears in simulated data, one of the rates changes from 0.1353 to 0.1496 (via positive-signed influence) or to 0.1225 (via negative-signed influence). The baseline rates suggest that ones appear only about 12% of the time for a given component, and that zeros appear about 87% of the time. So one can view estimating each nonzero entry in A as in effect attempting to detect a difference in Poisson rates on the order of 0.01 based on 12 data points per 100 in the dataset.

[3]To see this, consider that a count of 1 in the first component (its baseline mean) modulates the rate of the third component by a factor of $e^1 = 2.72$, so the third component's conditional mean at the following time step is 2.72. This difference in Poisson rates occurs frequently in simulated realizations from this parameter configuration since roughly one in three data points for the third component are expected to be 1. Due to the magnitude and frequency of this particular rate change, the nonzero entry in A is relatively easy to detect with few data points.

well, and any method should recover the second example parameters easily. Therefore, sampling from a moderately challenging region in the parameter space that excludes such possibilities is desirable for conducting meaningful method evaluations. To achieve this, it is helpful to quantify the difficulty of recovery so that one can check whether generated parameters are within the target difficulty range.

The essential difference in recoverability between the two examples above is the information that data realizations are expected to contain about the parameters. Unfortunately, Fisher information has no closed-form solution and is difficult to approximate in this setting in general due to the potential high dimensionality of the parameter space. Fisher information characterizes how difficult it is to find optima on the likelihood surface by quantifying the expected variability in the likelihood around the true parameter values. [4] As an alternative measure of information, a recoverability statistic for Poisson GVAR(1) parameters is constructed based on a deviance-based generalization of the signal-to-noise ratio (SNR) for linear models from Czanner et al. (2008, 2015). First an overview of generalized SNR is given, then extended to the

[4]The likelihood variability around the true parameters can be investigated empirically for each of the examples above to confirm the intuition that the relevant property is information content. The first set of parameters is challenging to recover because the likelihood varies little around the true parameter values, making them difficult to discern; the second set, by contrast, is easy to discern from the likelihood surface due to a clear peak around the true value. In the first (difficult) parameter configuration, computing the log-likelihood based on a simulated realization of length 100,000 for perturbations of A around the true value by multiplicative factors ranging from -3 to 3 produces an optimum when A is nearest to zero. Furthermore, overall the perturbations change the log-likelihood by a factor of at most 0.003 of the optimum value. As a result, the MLE fails because the likelihood does not vary on a meaningful scale. In other words, A is nearly unidentifiable. In the second configuration, computing the log-likelihood from a simulated realization of length 100 for perturbations around the true value by multiplicative factors ranging from -1.2 to 1.2 produces an optimum near the multiplicative factor 1 and an overall change in log-likelihood by up to a factor of 3.3 of the optimum value. By contrast, the likelihood in this example varies considerably about the true value, making the optimum easy to find.

Poisson GVAR(1) case, followed by a discussion of how well the extension captures intuitions about parameter estimability.

In the linear model $y = x\beta + \epsilon$ (with $\mathbb{E}\epsilon = 0$ and $\mathbb{E}\epsilon\epsilon' = \sigma^2 I$), call $x\beta$ the 'signal' component and ϵ the 'noise' component. The SNR is the ratio of signal variability to noise variability. Denoting by μ a vector with each entry equal to the grand mean $\bar{x}'\beta$, where $\bar{x}$ is the vector of column means of x, define the signal variability to be $(\mu - x\beta)'(\mu - x\beta)$. Define the noise variability to be $\epsilon'\epsilon$ or equivalently $(y - x\beta)'(y - x\beta)$. Then the SNR is:

$$\text{SNR} = \frac{(\mu - x\beta)'(\mu - x\beta)}{(y - x\beta)'(y - x\beta)}$$

Low SNR is interpreted as indicating a weak signal. This can arise in two ways: either the covariates x have low variances, in which case β is difficult to estimate,[5] or β is near zero, in which case the relationship between x and y is negligible.

An estimator of SNR can be obtained by replacing μ and β with the estimates $\hat{\mu} = \bar{y}$ (where $\bar{y}$ denotes an $n \times 1$ vector whose entries are $n^{-1}\sum_i y_i$) and $\hat{\beta} = (x'x)^{-1}x'y$. In that case, the estimator is the ratio of the model sum of squares to the residual sum of squares:

$$\hat{\text{SNR}} = \frac{(\bar{y} - \hat{y})'(\bar{y} - \hat{y})}{(y - \hat{y})'(y - \hat{y})} = \frac{SS_{model}}{SS_{residual}}$$

The classic partitioning of sums of squares in regression establishes that for the estimate $\widehat{\text{SNR}}$:

$$\widehat{\text{SNR}} = \frac{(y - \bar{y})'(y - \bar{y}) - (y - \hat{y})'(y - \hat{y})}{(y - \hat{y})'(y - \hat{y})} = \frac{SS_{total} - SS_{residual}}{SS_{residual}}$$

[5] If the covariates x have low variances, then $\det(x'x)$ is small and the variance of the OLS estimator $\sigma^2(x'x)^{-1}$ is large; thus, β is difficult to estimate accurately.

Thus, the estimated SNR can be interpreted as the estimated reduction in residual variance attributable to the covariates, scaled by the estimated residual variance.[6]

SNR can be extended to generalized linear models based on the last expression. While the same partitioning does not hold for the population quantities μ and β, the estimator suggests a closely related quantity. Let $SS(a,b) = (b-a)'(b-a)$, and consider:

$$\text{SNR}^* = \frac{SS(\mu, y) - SS(x\beta, y)}{SS(x\beta, y)}$$

When the errors ϵ are Gaussian, the log-likelihood (written as a function of $x\beta$ and the data) is $\ell(x\beta, y) = -SS(x\beta, y)/2\sigma^2 - n\log(2\pi\sigma^2)/2$. Now SNR^* can be rewritten in terms of deviance, defined as $\text{dev}(\eta, y) = -2(\ell(\eta, y) - \ell(y, y))$. Noting that $\ell(y, y) = -n\log(2\pi\sigma^2)/2$ since $SS(y, y) = 0$, it is immediate that $\text{dev}(\eta, y) = SS(\eta, y)/\sigma^2$. Therefore, one has from premultiplying SNR^* by σ^2/σ^2 that for Gaussian linear models,

$$\text{SNR}^* = \frac{\text{dev}(\mu, y) - \text{dev}(x\beta, y)}{\text{dev}(x\beta, y)}$$

The numerator $\text{dev}(\mu, y) - \text{dev}(x\beta, y)$ can be interpreted as the change in deviance attributable to the regression covariates.

Generalized SNR can be applied to describe the change in deviance attributable to the autoregressive portion of Poisson GVAR(1) processes. Denote the log-likelihood of an M-dimensional Poisson GVAR(1) process realization $x = (x_0, x_1, \ldots, x_T)' \in \mathbb{R}^{(T+1)\times M}$ (written as a function of the Poisson rate parameters $\lambda = (\lambda_0, \lambda_1, \ldots, \lambda_T)'$

[6]For another perspective, note that this is the scaled F statistic for the overall significance test in regression.

where $\lambda_t \in \mathbb{R}^M$) by:

$$\ell(\lambda; x) = \sum_{t=1}^{T} (x_t \log \lambda_t - \lambda_t) + C$$

Let $\lambda(A, \nu; x) \in \mathbb{R}^T$ be given by $\log \lambda_t(A, \nu; x_{t-1}) = \exp\{\nu + Ax_{t-1}\}$ for $t \geq 1$. Then, deviance is defined as: $\text{dev}(A, \nu; x) = -2(\ell(\lambda(A, \nu; x); x) - \ell(x; x))$. Now define recoverability of A and ν from data x as the scaled change in deviance attributable to the autoregressive component:

$$\text{recoverability}(A, \nu; x) = \frac{\text{dev}(0, \nu; x) - \text{dev}(A, \nu; x)}{\text{dev}(A, \nu; x)}$$

If the likelihood varies little in A, the change in deviance from setting A to zero will be small — note that the numerator is simply the difference in log-likelihood at (A, ν) and at $(0, \nu)$. Conversely, if the likelihood is sensitive to A, the change in deviance will be comparatively large. Intuitively, this quantity captures the information that the data realization x contains about A: if x is informative for A, recoverability should be large; otherwise, it should be small. Of course, the intercept ν modulates how informative data realizations are for any particular matrix A; smaller values of ν diminish the recoverability.

Recoverability is expressed as a function of the parameters conditional on a particular realization x. A simple strategy to estimate unconditional recoverability is to approximate its expected value by a parametric bootstrap: repeatedly simulate x using the parameters A and ν and compute the average of the quantity across realizations. Empirically, a modest number (10) of long (10,000) simulated realizations produces a stable estimate with relatively low variance. Executing this procedure for

the examples given at the beginning of the section produces a recoverability score of 0.00034 for the difficult example and a recoverability score of 6.65 for the easy example, aligning with intuition.

The proposed quantity measures recoverability of *at least part of A but not all of A*. Note that the log-likelihood can be written as a sum of log-likelihood components corresponding to the rows of A:

$$\ell(\lambda; x) = \sum_{t=1}^{T} (x_t \log \lambda_t - \lambda_t) + C = \sum_{m=1}^{M} \left(\sum_{t=1}^{T} (x_{t,m} \log \lambda_{t,m} - \lambda_{t,m}) + C_m \right) \stackrel{\text{def}}{=} \sum_{m=1}^{M} \ell_m(\lambda_m; x_m)$$

Now denote rows of A by a'_m, columns of x by x_m, elements of ν by ν_m, and define $\lambda_m(a'_m, \nu_m; x) \in \mathbb{R}^T$ by $\log \lambda_{t,m}(a'_m, \nu_m; x_{t-1}) = \nu_m + a'_m x_{t-1}$. The deviance can be decomposed into componentwise deviances:

$$\operatorname{dev}(A, \nu; x) = \sum_{m=1}^{M} \left(-2(\ell_m(\lambda_m(a'_m, \nu_m; x); x_m) - \ell_m(x_m; x_m))\right) \stackrel{\text{def}}{=} \sum_{m=1}^{M} \operatorname{dev}(a'_m, \nu_m; x)$$

Therefore, recoverability can be rewritten as:

$$\text{recoverability}(A, \nu; x) = \sum_{m=1}^{M} \frac{\operatorname{dev}(0, \nu_m; x) - \operatorname{dev}(a'_m, \nu_m; x)}{\operatorname{dev}(A, \nu; x)}$$

Thus, if the change in deviance away from zero is considrable for any single row of A, A has a high recoverability score even if other rows of A are difficult to estimate from the likelihood.

Consider the following example that is a mixture of the parameters for the third component from the easy example with the parameters for the second component

from the hard example:

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0.1 \\ 1 & 0 & 0 \end{bmatrix} \quad \nu = \begin{bmatrix} 0 \\ -2 \\ 0 \end{bmatrix}$$

An estimate of $\mathbb{E}(\text{recoverability}(A, \nu; x))$ based on 10 simulated realizations of length 10,000 each is 7.46; however, the contribution to $\text{recoverability}(A, \nu; x)$ from the second component is 0.01 and the contribution from the third component is 7.45. In other words, the parameters associated with the third process component are easy to recover and those associated with the second are difficult, but $\text{recoverability}(A, \nu; x)$ does not reflect this discrepancy. This is a limitation with respect to how precisely $\text{recoverability}(A, \nu; x)$ represents ease or difficulty of estimation. However, the quantity is sufficiently useful for the present purposes as high values indicate that at least some subset of parameters can be estimated easily and low values indicate that no such subset exists, so it still captures the extremes that should be avoided in parameter generation.[7]

Sampling from the parameter space. Algorithm 4.4 and the recoverability statistic are used in conjunction to sample from the Poisson GVAR(1) parameter space for any dimension M and sparsity s. The method is simple: first, Algorithm 4.4 is used to generate a proposal for A; then, the proposal is accepted if estimated recoverability

[7]The quantity could be easily refined to the level of individual parameters, in the sense that a separate recoverability score could be calculated for each nonzero parameter a_{mj}, such as $\frac{\text{dev}(0,\nu_m;x)-\text{dev}((0\cdots a_{mj}\cdots 0),\nu_m;x)}{\text{dev}((0\cdots a_{mj}\cdots 0),\nu_m;x)}$. From this, a composite metric could be formed that more precisely measures overall recoverability of A.

is within a target range. This method is given in detail as Algorithm 4.5.

Algorithm 4.5 Sample from the Poisson GVAR(1) parameter space.

Require:
parameter generation settings M, s, θ, p
intercept $\nu \in \mathbb{R}^M$
recoverability range $(r_{\min}, r_{\max})$
recoverability estimation settings T_{sim}, N
while $\bar{r} \notin (r_{\min}, r_{\max})$ **do**
$A \leftarrow$ Algorithm4.4(M, s, θ, p)
for $n = 1$ to N **do**
generate data $x \in \mathbb{R}^{M \times T_{\text{sim}}}$ from Poisson GVAR(1) with A, ν
$r_n \leftarrow$ recoverability$(A, \nu; x)$
end for
$\bar{r} \leftarrow N^{-1} \sum_n r_n$
end while
Ensure: A

4.3.2 Simulation design

With the foregoing methodology established for sampling from the Poisson GVAR(1) parameter space of arbitrary dimension at a prespecified sparsity level (Algorithm 4.5), and the candidate estimation methods developed in Section 4.2 (Table 4.5), it is now possible to describe cogently the simulation study from which the key results in this chapter are derived.

In the study, each method is used to estimate the M-dimensional Poisson GVAR(1)

model:

$$X_{t,m}|X_{t-1} \overset{indep.}{\sim} \text{Poisson}(\lambda_{t,m}), \quad m = 1, \ldots, M,\ t \in \mathbb{Z}$$
$$\log \lambda_t = \nu + AX_{t-1}$$

The parameter of interest is A and it is assumed to be s-sparse ($\|\text{vec}(A)\|_0 = sM^2$). Synthetic data of length T are generated according to this process, and each method is used to recover the parameter A. The dimension M, sparsity s, and realization length T are varied.

Study design. The study is a factorial design generated by three levels of process dimensions, $M = 10, 15, 20$, and three levels of sparsity, $s = 0.01, 0.02, 0.05$. For each factorial combination, three data realization lengths $T = 500, 1000, 2000$ are considered. For the main effects there is parameter-level replication: 10 different A matrices are generated for each factorial combination. Further, there is also realization-level replication: for each factorial combination, each A, and each realization length T, 5 datasets are generated. In total, the number of datasets constructed in the design is $5 \times 3 \times 10 \times 3 \times 3 = 1350$ (data replicates × realization lengths × parameter replicates × sparsity levels × dimensions). Each of the four methods are applied to every dataset and the estimates are recorded.

Parameter sampling settings. To sample from the parameter space for a given factorial combination of dimension M and sparisty s, Algorithm 4.5 is implemented with parameter generation settings $\theta = 1$ and $p = 0.3$ and target recoverability between

$r_{\min} = 0.5$ and $r_{\max} = 1.5$ based on $N = 10$ realizations of length $T_{\text{sim}} = 100,000$. Although the recoverability statistic is a function of both the intercept ν and the autoregressive parameter A, the parameter of interest is A. Obtaining an A in the target recoverability range can be made easier by conditioning on a fixed intercept for each factorial combination of dimension and sparsity (rather than also generating an intercept). To accomplish this, a large number of A matrices were generated as above for each setting, and recoverability was estimated across a coarse grid of intercepts ν with equal entries ν_m. The intercept that gave a distribution of estimated recoverabilities centered nearest the target range was then fixed. The intercept entries fixed for each combination are shown below.

	$M = 10$	$M = 15$	$M = 20$
$s = 0.01$	$\nu_m = 0.5$	$\nu_m = 0.4$	$\nu_m = 0.3$
$s = 0.02$	$\nu_m = 0.5$	$\nu_m = 0.4$	$\nu_m = 0.3$
$s = 0.05$	$\nu_m = 0.1$	$\nu_m = 0.1$	$\nu_m = 0.1$

Table 4.6: Intercept entries fixed for each combination of process dimension and parameter sparsity in the simulation study. The sample averages of estimated recoverabilities of 50 generated A matrices for the intercepts shown in the cells ranged from 0.79 to 1.57 and the sample standard deviations ranged from 0.02 to 0.07.

4.3.3 Simulation results

The simulation study results are summarized in Figures 4.1, 4.2, and 4.3, and Supplementary Figures S.4, S.6, and S.5. The main figures focus on comparing selection errors associated with each method. Selection errors are reported as the proportion of entries in A whose zero/nonzero status is misclassified by an estimate $\hat{A}$. So for

example, an error rate of 0.1 indicates that 10% of the entries in $\hat{A}$ are either false positives (zero-valued entries of A erroneously estimated as nonzero in $\hat{A}$) or false negatives (nonzero-valued entries in A erroneously estimated as zero in $\hat{A}$). These error rates are compared on average for each combination of sparsity s, dimension M, and realization length T, and also on a dataset-by-dataset basis.

Figure 4.1 shows the average selection error for each method, plotted against realization length T, for each combination of parameter sparsity s and process dimension M. The total number of nonzero parameters for each combination of M and s is given by sM^2. Thus, the lowest-dimensional and sparsest case ($M = 10$ and $s = 0.01$) represents problems involving selection and estimation of a single nonzero autoregressive parameter out of a possible 100; and the highest-dimensional and densest case ($M = 20$ and $s = 0.05$) represents problems involving selection and estimation of 20 nonzero parameters out of a possible 400. For all methods, the average error rates range from 0 to 0.1 across combinations of M, s, T, and, as one might expect in general, selection errors decrease with T and increase with M and s. Only in the lowest-dimensional and sparsest setting do all methods select equally well on average. Otherwise, support estimation by support aggregation outperforms support estimation by LASSO supports, support selection by model aggregation outperforms support selection by cross validation, and the performance gaps widen as s and M increase. The combination performs favorably, maintaining a $0-2\%$ error rate across cases.

The support estimation method has the relatively larger impact on average selection error compared with the support selection method, which produces a rela-

tively narrower difference. In particular, the combination of support estimation by support aggregation (the better-performing support estimation method) with support selection by cross validation (the worse-performing support selection method) outperforms the combination of support estimation by LASSO supports (the worse-performing support estimation method) with support selection by model aggregation (the better-performing support selection method).

These behaviors are driven largely by false positive rates. Supplementary Figures S.6 and S.5 show selection error comparisons parsed separately (rather than as a combined metric) in terms of false positive and false negative counts. Therein, the false negative counts are nearly equal (and low) on average across methods (Supplementary Figure S.5), whereas the false positive counts differ considerably across methods (Supplementary Figure S.6).

Figure 4.2 shows the marginal effect of the support estimation method on a dataset-by-dataset basis. For each dataset, the selection error rate (again, the proportion of misclassified entries in A) from the support aggregation method is plotted against the selection error rate from the LASSO support estimation method, keeping the support selection method fixed. These scatterplots are produced for each combination of sparsity s and dimension M, shown as the rows and columns of the figure. Each scatterplot includes the simulated datasets of every length ($T = 500, 1000, 2000$) corresponding to all generated parameters (ten distinct A matrices) without aesthetic distinction. Simple regression lines are superimposed to help visualize the trend, along with a $y = x$ reference. The trends, and in fact the majority of the points themselves, are below the reference line, indicating that the selection error rates for the two meth-

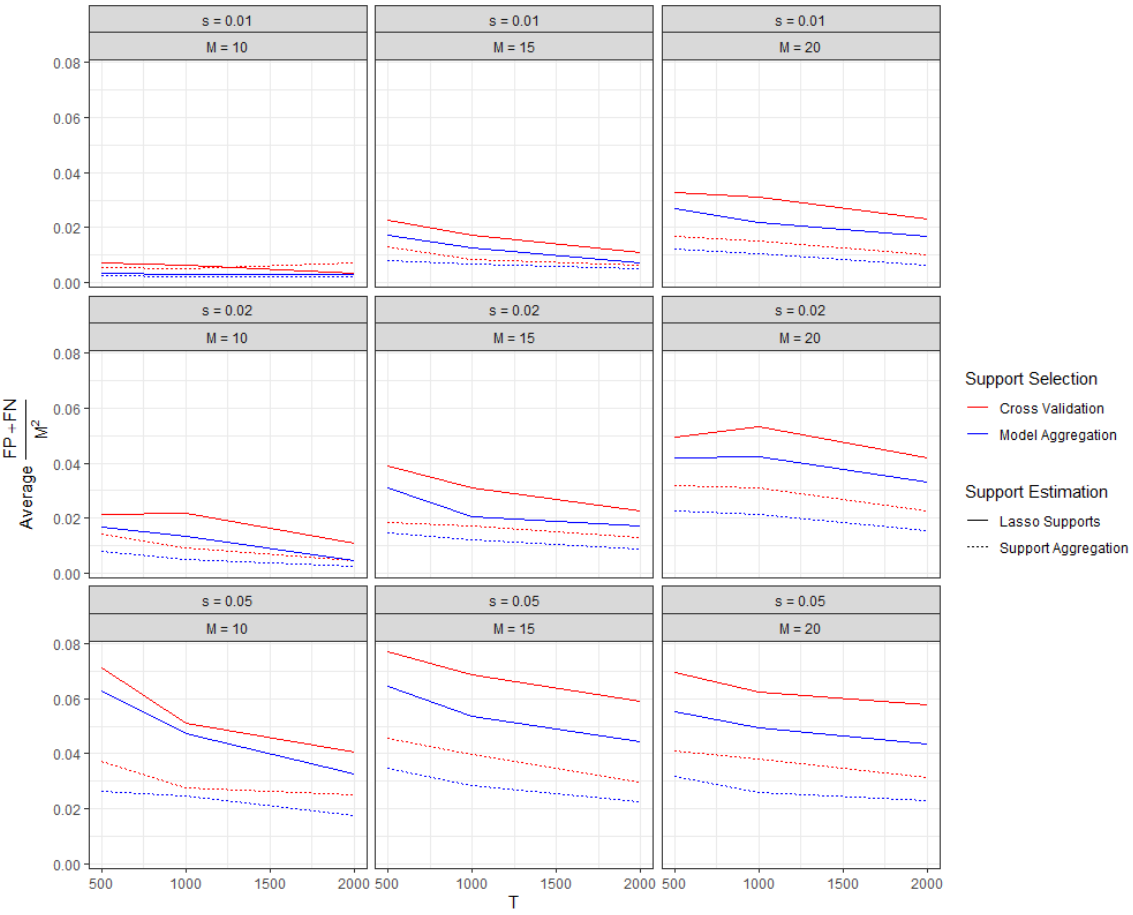


Figure 4.1: Average selection errors observed in simulation, reported as the average proportion of matrix entries that are incorrectly classified as zero or nonzero, for each combination of support estimation and support selection method. Average selection errors are plotted against simulated realization length. Separate panels are shown for each combination of sparsity s (row) and process dimension M (column).

ods utilizing support aggregation for support estimation are lower than the selection error rates for the two methods utilizing LASSO support estimation. In other words, the marginal effect of exchanging support aggregation for LASSO support estimation is to lower the selection error rate.

Figure 4.3 shows the marginal effect of the support selection method on a dataset-by-dataset basis. The same type of comparison is given as in Figure 4.2, except as applied to the support selection method rather than the support estimation method: that is, the scatterplots show the selection error rates for support selection by model aggregation plotted against the selection error rates for support selection by cross validation, with the support estimation methods fixed. The trends are all below the reference line, and in all but the $M = 10$ cases, most of the scatter is also below the reference line, indicating that the selection error rates are lower for support selection by model aggregation than by cross validation, in combination with either support estimation method. The marginal effect of exchanging model aggregation for cross validation, then, is to lower the selection error rates.

The dataset-by-dataset comparisons (Figs. 4.2, 4.3) are consistent with the behavior of average selection errors shown in Figure 4.1: both model aggregation and support aggregation confer improvements in selection accuracy in any combination, yet model aggregation confers a relatively narrower improvement.

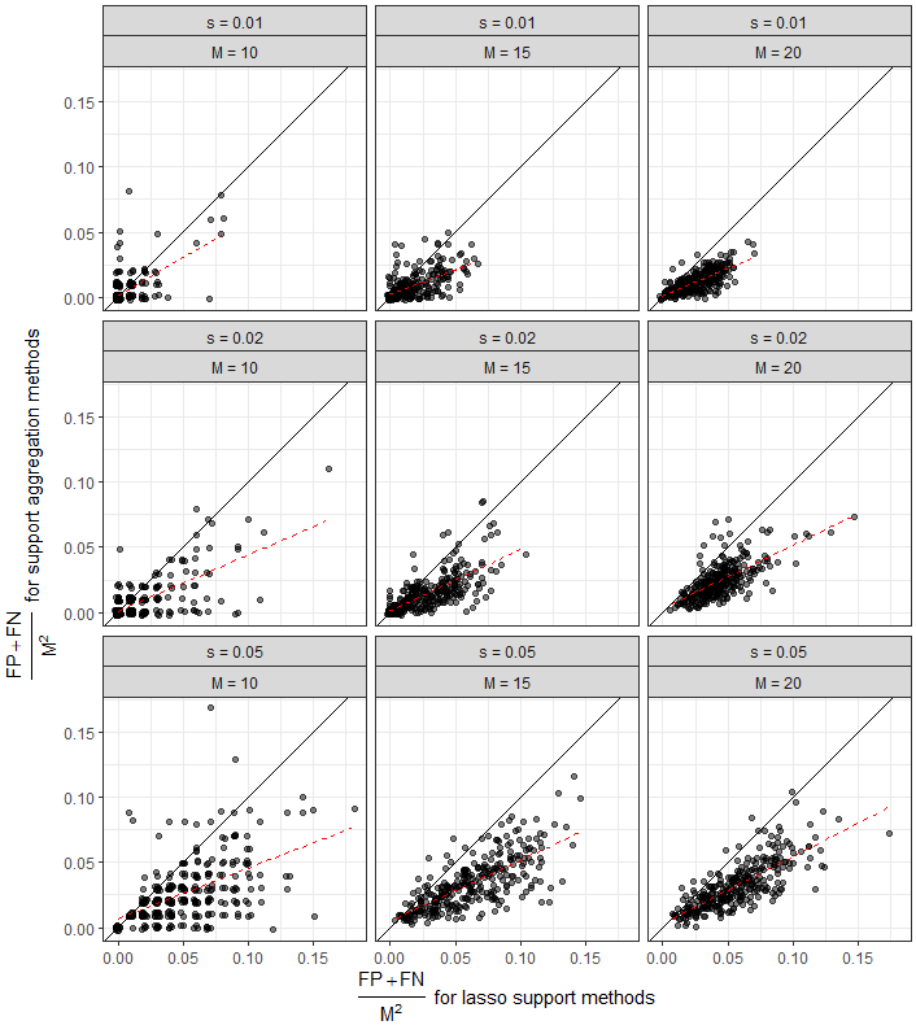


Figure 4.2: Marginal effect of support estimation method on selection errors. In each panel, the proportion of incorrectly classified (zero/nonzero) matrix entries given by the LASSO support estimation method (with either support selection method) is plotted against the proportion of incorrectly classified matrix entries given by the support aggregation method (with the same support selection method) for each simulated dataset. A dashed simple linear regression line is overlaid on the scatterplot to help visualize the trend. Separate panels are shown for each combination of sparsity s (row) and process dimension M (column).

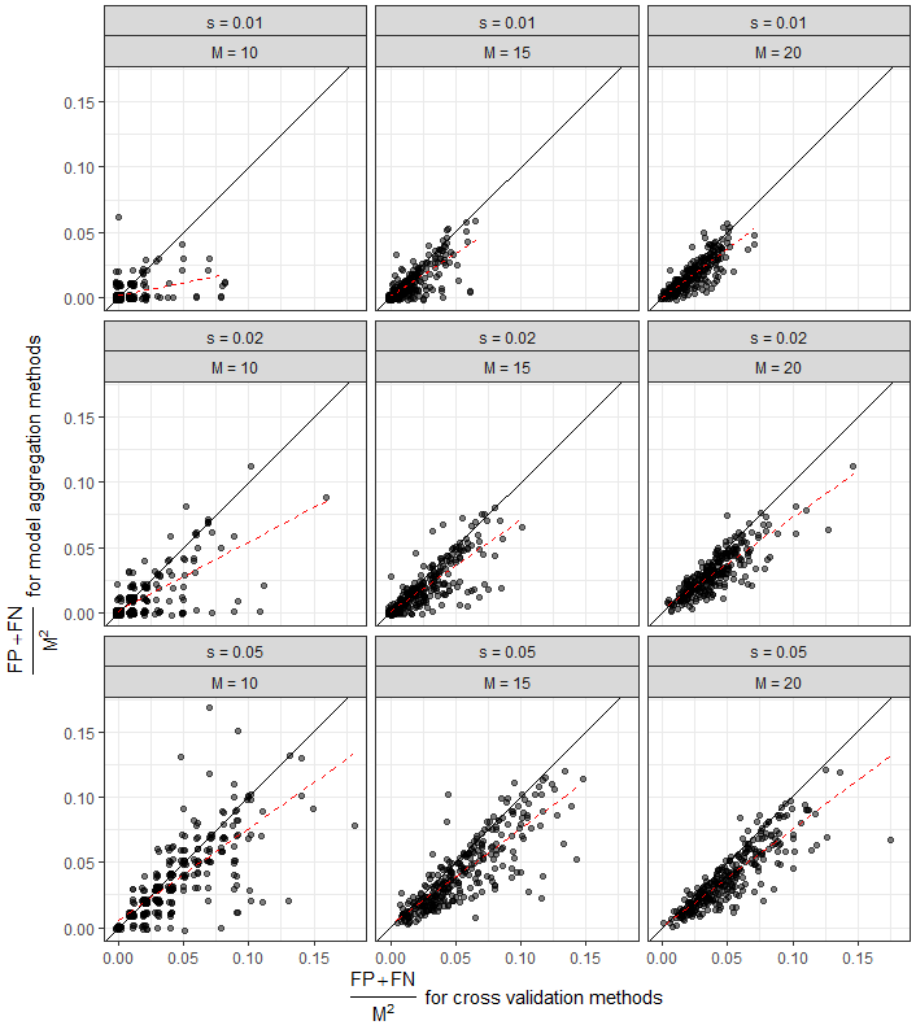


Figure 4.3: Marginal effect of support selection method on selection errors. In each panel, the proportion of incorrectly classified (zero/nonzero) matrix entries given by the cross validation support selection method (with either support estimation method) is plotted against the proportion of incorrectly classified matrix entries given by the model aggregation support selection method (with the same support estimation method) for each simulated dataset. A dashed simple linear regression line is overlaid on the scatterplot to help visualize the trend. Separate panels are shown for each combination of sparsity s (row) and process dimension M (column).

4.4 Discussion

Findings. The simulation study presented in this chapter indicates that both support aggregation and model aggregation marginally improve selection performance, and that their combination performs best among all those considered. The average selection error rates for this method vary between roughly 0-3% — that is, 0-3% of estimated parameters are erroneously zero or erroneously nonzero — depending on process dimension, parametric sparsity, and time series length. Support aggregation (the optimally-performing support estimation method) contributes relatively more to improving selection accuracy in comparison with its benchmark, reducing average error rates by up to roughly 3%; however, model aggregation (the optimally-performing support selection method) also produces considerable improvements, reducing average error rates by up to 2%. The combination reduces average error rates by up to 5% relative to the worst-performing method. Further, the study indicates that false positive rates — rates of erroneous nonzero estimates — drive the difference in selection performance, and that estimation errors are comparable despite large differences in selection errors.

The preliminary method variant comparisons showed that conditioning support selection on a fixed set of supports — a strategy employed in the original empirical method to reduce computation time — produces a deterioration of selection accuracy. Performing unconditional support selection clearly improves the method.

Additionally, a number of modifications were introduced. First, a novel thresholding strategy was proposed for the model aggregation method: thresholding position

frequencies in the collection of all selected supports rather than thresholding the elements in the collection based on sparsity quantiles. Second, the final estimation method was simplified from an average of multiple estimates to a single estimate. Lastly, an alternative resampling strategy was proposed: a blockwise jackknife scheme replaced the moving block bootstrap used in the original method. All proposals were based on improved stability observed in anecdotal experiments.

Beyond identifying optimal method design, the general methodological framework was extended to sparse estimation of Poisson GVAR models, which can be applied to multivariate count data without data transformation where classical VAR models cannot. This both broadens the range of potential applications for the proposed method, and also establishes some baseline methods for model estimation and methodological performance comparisons where few previously existed.

In the course of modifying and extending the method, a number of contributions were made in order to develop the simulation study. First, an algorithm for generating sparse stable parameter matrices with both positive-signed and negative-signed entries was developed. Second, a recoverability metric was proposed to quantify difficulty of estimation in the absence of closed-form expressions or simple approximations for Fisher information.

Challenges encountered. The majority of challenges encountered in the course of this work arose in the context of computations. First, the computationally intensive nature of the methods — repeated estimation of moderately high-dimensional models from resamples of the data — imposed feasibility limits on the data dimen-

sions considered in the simulation study, the number of replications at the parameter level and at the dataset level, and the number of levels of dimension, sparsity, and time series length considered. In many of the higher-dimensional settings, a single estimate could take several minutes to compute, and when aggregated over all the repetitions required in the study, simulation for such factorial combinations could take weeks to complete. A second computational challenge had to do with adequate selection of the range of values used for the regularization parameter λ. The standardization of process parameters according to estimated recoverability had the effect of standardizing the regularization path across factorial combinations. However, this was a lucky coincidence; in general, the scale of appropriate λ values varies dramatically over the parameter space. Lastly, the regularized likelihood optimizations were based on vectorizing data matrices and using LASSO-GLM, and a number of practical hurdles with commerically available implementations required developing manual implementations of pathwise coordinate descent algorithms for LASSO-GLM. In initial iterations, R and MATLAB implementations were considered. It was discovered that the `glmnet` package in R does not handle user-specified regularization paths well, and since the package calls routines written in other code environments, many errors are untraceable. On the other hand, MATLAB's implementations are more stable under perturbation of λ, but do not currently leverage sparse linear algebra operations, which are critical to efficient computation in the VAR/GVAR context. Moreover, when intercepts are folded into the likelihood optimization, an unusual strategy for distributing the penalty is required since VAR/GVAR models have multiple intercepts, and neither implementation is easily adaptable to accommodate this

requirement.

Future work. This work could be extended in several directions. First, the novel portion of the proposed method is the model aggregation technique. This technique is articulated in fully general terms, and could be directly applied in any model selection problem; exploring the performance of the method in other contexts (sparse regression or GLM, sparse factorizations, etc.) relative to existing selection methods could reveal similar improvements to those observed in this work and provide insight into the problem conditions under which it performs well. Second, less constrained frameworks for simulating process parameters could be developed to allow for exploration of more complex processes. Third, the development of computationally efficient implementations of the methods developed in this work would allow for explorations of performance in higher-dimensional settings and enhance understanding of the method's advantages and limitations, and dissemination of existing or improved implementations would encourage the methods' use.

Chapter 5: Conclusion

The foregoing chapters developed a resampling-intensive empirical method for sparse estimation of Poisson GVAR(D) models in stages. Chapter 2 presented a preliminary version of the method that was developed in the context of VAR models for continuous data and illustrated its application in a network analysis of financial time series. Chapter 3 considered a generalization of the VAR model to conditionally Poisson count vectors and provided a rigorous analysis of the stability properties of the corresponding class of stochastic processes. The analysis highlighted some of the challenges associated with Poisson GVAR models while also showing that stability can be preserved for flexible parameter regimes without the use of data transformations. Chapter 4 refined the methodology presented earlier in the thesis and demonstrated its advantages in the context of sparse estimation of Poisson GVAR(D) models through a simulation study. Very little methodology exists for this particular estimation problem, so the novel method was compared with the only existing method (Hall et al., 2016a,b, 2018; Mark et al., 2017). Considerable improvements in selection accuracy, largely achieved through false positive control, were demonstrated across data dimensions and averaging over various model parametrizations.

A significant practical challenge in the development of the methodology was the stability of the Poisson GVAR model, since simulation experiments required generating parameters, parameter configurations (*i.e.*, graphical structures), and subse-

quently, data. In order to avoid having results on estimation performance depend too heavily on specific data-generating processes in the model class, heuristics for sampling from the parameter space were needed, and fully random draws from the parameter space too often result in unstable or uncomputable processes. The need for these heuristics motivated much of the work presented in Chapter 3 and called attention to the somewhat surprising fact that the same degree of rigor in model specification of univariate generalized autoregression (Davis and Liu, 2012; Fahrmeir and Tutz, 1994; Neumann et al., 2011; Zeger and Qaqish, 1988) has not been present in the literature on vector extensions of these models. Chapter 3 represents an initial effort towards closing this gap.

Another practical challenge was computation on large scales. Between the simulation studies and data application, the process (vector) dimensions considered in this thesis range from 10 to 80, and the empirical method presented in the thesis has been successfully applied to data of vector dimension around 200 using relatively naive implementations on a laptop computer and to data of vector dimension 1000 using a distributed implementation on a multi-node system (see Appendix C and Balasubramanian et al. (2020)). While the term 'high-dimensional' is used variously, most authors explicitly compute VAR models on datasets with tens to hundreds of vector components (Fan et al., 2011), and most methodological works focus on data with 10-50 vector components (including the chapters presented here; see also Basu et al. (2015); Davis et al. (2016)). In practice, larger-scale data are often handled by applying preliminary dimension reduction or selection techniques prior to model estimation (Pillow et al., 2008). In general, it is not empirically known how selec-

tion and estimation performance scale in process dimension, and many applications extrapolate far beyond scales explored in methodological work. One benefit of the improved false positive rates that result from using the resampling-based methods presented in Chapters 2 and 4 is that controlled selection errors are more likely under increases in vector dimension to application scale.

Several extensions of the work here are possible, and have been described in the concluding discussions in each chapter. Two in particular are worth underscoring. First, the key innovation of the methodological work in Chapters 2 and 4 is the use of model aggregation in place of model (hyperparameter) selection by cross validation, and Chapter 4 demonstrates the selection improvements associated with this innovation in the context of Poisson GVAR estimation, which represents a relatively challenging selection problem due to dependence, nonlinearity, and discreteness. While it has been known for some time that cross validation is not in general a consistent selection technique (Shao, 1993; Zhang, 1993), the technique is used anyway in a wide range of applications. More recent methodological research has examined the consistency of selection by cross validated LASSO estimation for linear models (Chetverikov et al., 2019; Lei, 2019; Zhao and Yu, 2006). Model aggregation provides a potential alternative to cross validation, and the results of Chapter 4 suggest it may be a promising technique in a broader range of selection problems. Second, the probabilistic analysis of Poisson GVAR processes presented in Chapter 3 gives only a partial understanding of the process behavior focusing on process moments. Determination of ergodicity conditions is a key effort that would support theory for likelihood estimation and inference, and would thus represent a valuable further contribution.

Bibliography

Abdelhakim Aknouche, Wissam Bentarzi, and Nacer Demouche. On periodic ergodicity of a general periodic mixed poisson autoregression. *Statistics & Probability Letters*, 134:15–21, 2018.

Andrew Arnold, Yan Liu, and Naoki Abe. Temporal causal modeling with graphical granger methods. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 66–75, 2007.

Gowtham Atluri, Anuj Karpatne, and Vipin Kumar. Spatio-temporal data mining: A survey of problems and methods. *ACM Computing Surveys (CSUR)*, 51(4):1–41, 2018.

Francis R Bach. Bolasso: model consistent lasso estimation through the bootstrap. In *Proceedings of the 25th international conference on Machine learning*, pages 33–40, 2008.

Francis R Bach and Michael I Jordan. Learning graphical models for stationary time series. *IEEE transactions on signal processing*, 52(8):2189–2199, 2004.

Mahesh Balasubramanian, Trevor D Ruiz, Brandon Cook, Mr Prabhat, Sharmodeep Bhattacharyya, Aviral Shrivastava, and Kristofer E Bouchard. Scaling of union of intersections for inference of granger causal networks from observational data. In *2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 264–273, 2020.

Matteo Barigozzi and Christian Brownlees. Nets: Network estimation for time series. *Journal of Applied Econometrics*, 34(3):347–364, 2019.

Matteo Barigozzi and Marc Hallin. A network analysis of the volatility of high dimensional financial series. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 66(3):581–605, 2017.

Danielle Bassett and Olaf Sporns. Network neuroscience. *Nature Neuroscience*, 20 (3):353–364, 2017.

Sumanta Basu. *Modeling and Estimation of High-dimensional Vector Autoregressions.* PhD thesis, The University of Michigan, 2014.

Sumanta Basu, George Michailidis, et al. Regularized estimation in sparse high-dimensional time series models. *The Annals of Statistics*, 43(4):1535–1567, 2015.

Kristofer Bouchard, Alejandro Bujan, Fred Roosta, Shashanka Ubaru, Mr Prabhat, Antoine Snijders, Jian-Hua Mao, Edward Chang, Michael W Mahoney, and Sharmodeep Bhattacharya. Union of intersections (uoi) for interpretable data driven discovery and prediction. In *Advances in Neural Information Processing Systems*, pages 1078–1086, 2017.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

Johannes Bracher and Leonhard Held. Multivariate endemic-epidemic models with higher-order lags and an application to outbreak detection. *arXiv preprint arXiv:1901.03090*, 2019.

Patrick T Brandt and Todd Sandler. A bayesian poisson vector autoregression model. *Political Analysis*, pages 292–315, 2012.

Nick Bromer. Superexponentiation. *Mathematics Magazine*, 60(3):169–174, 1987.

Emery Brown. Theory of point processes for neural systems. In Carson Chow, Boris Gutkin, David Hansel, Claude Meunier, and Jean Dalibard, editors, *Methods and models in neurophysics*, volume 80 of *Les Houches*, chapter 14, pages 691–727. Elsevier Science, 1 edition, 2005.

Emery N Brown, Robert E Kass, and Partha P Mitra. Multiple neural spike train data analysis: state-of-the-art and future challenges. *Nature neuroscience*, 7(5): 456–461, 2004.

Peter Bühlmann and Hans R Künsch. Block length selection in the bootstrap for time series. *Computational Statistics & Data Analysis*, 31(3):295–310, 1999.

Peter Bühlmann and Sara Van De Geer. *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media, 2011.

György Buzsáki, Costas Anastassiou, and Christof Koch. The origin of extracellular fields and currents – eeg, ecog, lfp and spikes. *Nature Reviews Neuroscience*, 13: 407–420, June 2012.

Laura Cavalcante, Ricardo J Bessa, Marisa Reis, and Jethro Browell. Lasso vector autoregression structures for very short-term wind power forecasting. *Wind Energy*, 20(4):657–675, 2017.

Changhua Chen, Richard A Davis, and Peter J Brockwell. Order determination for multivariate autoregressive processes using resampling methods. *Journal of multivariate analysis*, 57(2):175–190, 1996.

Denis Chetverikov, Zhipeng Liao, and Victor Chernozhukov. On cross-validated lasso in high dimensions. Technical report, UCLA, 2019. Working paper.

David R Cox, Gudmundur Gudmundsson, Georg Lindgren, Lennart Bondesson, Erik Harsaae, Petter Laake, Katarina Juselius, and Steffen L Lauritzen. Statistical analysis of time series: Some recent developments. *Scandinavian Journal of Statistics*, pages 93–115, 1981.

Gabriela Czanner, Sridevi V Sarma, Uri T Eden, and Emery N Brown. A signal-to-noise ratio estimator for generalized linear model systems. In *Proceedings of the world congress on engineering*, volume 2, 2008.

Gabriela Czanner, Sridevi V Sarma, Demba Ba, Uri T Eden, Wei Wu, Emad Eskandar, Hubert H Lim, Simona Temereanca, Wendy A Suzuki, and Emery N Brown. Measuring the signal-to-noise ratio of a neuron. *Proceedings of the National Academy of Sciences*, 112(23):7141–7146, 2015.

Rainer Dahlhaus. Graphical interaction models for multivariate time series. *Metrika*, 51(2):157–172, 2000.

Rainer Dahlhaus and Michael Eichler. Causality and graphical models in time series analysis. *Oxford Statistical Science Series*, pages 115–137, 2003.

Richard A Davis and Heng Liu. Theory and inference for a class of observation-driven models with application to time series of counts. *arXiv preprint arXiv:1204.3915*, 2012.

Richard A Davis, William TM Dunsmuir, and Ying Wang. Modeling time series of count data. *Statistics Textbooks and Monographs*, 158:63–114, 1999.

Richard A Davis, William TM Dunsmuir, and Sarah B Streett. Observation-driven models for poisson counts. *Biometrika*, 90(4):777–790, 2003.

Richard A Davis, Pengfei Zang, and Tian Zheng. Sparse vector autoregressive modeling. *Journal of Computational and Graphical Statistics*, 25(4):1077–1096, 2016.

Jethro Dowell and Pierre Pinson. Very-short-term probabilistic wind power forecasts by sparse vector autoregression. *IEEE Transactions on Smart Grid*, 7(2):763–770, 2015.

Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

Michael Eichler. *Graphical models in time series analysis*. PhD thesis, Universität Heidelberg, 1999.

L Fahrmeir. A note on asymptotic testing theory for nonhomogeneous observations. *Stochastic processes and their applications*, 28(2):267–273, 1988.

Ludwig Fahrmeir. Asymptotic likelihood inference for nonhomogeneous observations. *Statistische Hefte*, 28(1):81, 1987.

Ludwig Fahrmeir and Heinz Kaufmann. Consistency and asymptotic normality of the maximum likelihood estimator in generalized linear models. *The Annals of Statistics*, pages 342–368, 1985.

Ludwig Fahrmeir and Gerhard Tutz. *Multivariate Statistical Modeling Based on Generalized Linear Models*. Springer-Verlag: New York, 1994.

Jianqing Fan, Jinchi Lv, and Lei Qi. Sparse high-dimensional models in economics. *Annu. Rev. Econ*, 3:291–317, 2011.

Konstantinos Fokianos. Count time series models. In *Handbook of statistics*, volume 30, pages 315–347. Elsevier, 2012.

Konstantinos Fokianos and Dag Tjøstheim. Log-linear poisson autoregression. *Journal of Multivariate Analysis*, 102(3):563–578, 2011.

Konstantinos Fokianos and Dag Tjøstheim. Nonlinear poisson autoregression. *Annals of the Institute of Statistical Mathematics*, 64(6):1205–1225, 2012.

Konstantinos Fokianos, Anders Rahbek, and Dag Tjøstheim. Poisson autoregression. *Journal of the American Statistical Association*, 104(488):1430–1439, 2009.

Jerome Friedman, Trevor Hastie, Holger Höfling, and Robert Tibshirani. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1(2):302–332, 2007.

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1): 1, 2010.

Clive WJ Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: journal of the Econometric Society*, pages 424–438, 1969.

Eric C Hall, Garvesh Raskutti, and Rebecca Willett. Inference of high-dimensional autoregressive generalized linear models. *arXiv preprint arXiv:1605.02693*, 2016a.

Eric C Hall, Garvesh Raskutti, and Rebecca Willett. Inferring high-dimensional poisson autoregressive models. In *2016 IEEE Statistical Signal Processing Workshop (SSP)*, pages 1–5, 2016b.

Eric C Hall, Garvesh Raskutti, and Rebecca M Willett. Learning high-dimensional generalized linear autoregressive models. *IEEE Transactions on Information Theory*, 65(4):2401–2422, 2018.

Fang Han, Huanran Lu, and Han Liu. A direct estimation of high dimensional stationary vector autoregressions. *Journal of Machine Learning Research*, 2015.

Andréas Heinen and Erick Rengifo. Multivariate autoregressive modeling of time series count data using copulas. *Journal of Empirical Finance*, 14(4):564–583, 2007.

Andréas Heinen, Erick Rengifo, et al. Multivariate modelling of time series count data: an autoregressive conditional poisson model. Technical report, Universite catholique de Louvain, 2003.

Nan-Jung Hsu, Hung-Lin Hung, and Ya-Mei Chang. Subset selection for vector autoregressive processes using lasso. *Computational Statistics & Data Analysis*, 52(7):3645–3657, 2008.

Anthony R Ives, B Dennis, KL Cottingham, and SR Carpenter. Estimating community stability and ecological interactions from time-series data. *Ecological monographs*, 73(2):301–330, 2003.

PA Jacobs and PAW Lewis. A mixed autoregressive-moving average exponential sequence and point process (earma 1, 1). *Advances in Applied Probability*, pages 87–104, 1977.

Eugenia Kalnay, Masao Kanamitsu, Robert Kistler, William Collins, Dennis Deaven, Lev Gandin, Mark Iredell, Suranjana Saha, Glenn White, John Woollen, et al. The ncep/ncar 40-year reanalysis project. *Bulletin of the American meteorological Society*, 77(3):437–472, 1996.

Dimitris Karlis and Evdokia Xekalaki. Mixed poisson distributions. *International Statistical Review/Revue Internationale de Statistique*, pages 35–58, 2005.

Anuj Karpatne, James Faghmous, Jaya Kawale, Luke Styles, Mace Blank, Varun Mithal, Xi Chen, Ankush Khandelwal, Shyam Boriah, Karsten Steinhaeuser, et al. Earth science applications of sensor data. In *Managing and Mining Sensor Data*, pages 505–530. Springer, 2013.

Anuj Karpatne, Imme Ebert-Uphoff, Sai Ravela, Hassan Ali Babaie, and Vipin Kumar. Machine learning for the geosciences: Challenges and opportunities. *IEEE Transactions on Knowledge and Data Engineering*, 2018.

Benjamin Kedem and Konstantinos Fokianos. *Regression models for time series analysis*, volume 488. John Wiley & Sons, 2005.

Donald Ervin Knuth. Mathematics and computer science: coping with finiteness. *Science (New York, NY)*, 194(4271):1235–1242, 1976.

Jens-Peter Kreiss and Soumendra Nath Lahiri. Bootstrap methods for time series. In Tata Rao, Suhasini Rao, and C. Rao, editors, *Handbook of Statistics*, volume 30 of *Time Series Analysis: Methods and Applications*, chapter 1, pages 3–26. Elsevier, 2012.

Hans R Kunsch et al. The jackknife and the bootstrap for general stationary observations. *The Annals of Statistics*, 17(3):1217–1241, 1989.

SN Lahiri et al. Theoretical comparisons of block bootstrap methods. *The Annals of Statistics*, 27(1):386–404, 1999.

AJ Lawrance and PAW Lewis. An exponential moving-average sequence and point process (ema1). *Journal of Applied Probability*, pages 98–113, 1977.

Jing Lei. Cross-validation with confidence. *Journal of the American Statistical Association*, pages 1–20, 2019.

Regina Y Liu and Kesar Singh. Moving blocks jackknife and bootstrap capture weak dependence. *Exploring the limits of bootstrap*, 225:248, 1992.

Aurélie C Lozano, Naoki Abe, Yan Liu, and Saharon Rosset. Grouped graphical granger modeling for gene expression regulatory networks discovery. *Bioinformatics*, 25(12):i110–i118, 2009a.

Aurelie C Lozano, Hongfei Li, Alexandru Niculescu-Mizil, Yan Liu, Claudia Perlich, Jonathan Hosking, and Naoki Abe. Spatial-temporal causal modeling for climate change attribution. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 587–596, 2009b.

Helmut Lütkepohl. *New introduction to multiple time series analysis*. Springer Science & Business Media, 2005.

Ben Mark, Garvesh Raskutti, and Rebecca Willett. Network estimation via poisson autoregressive models. In *2017 IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 1–5, 2017.

Ed McKenzie. Some arma models for dependent sequences of poisson counts. *Advances in Applied Probability*, pages 822–835, 1988.

Nicolai Meinshausen and Peter Bühlmann. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473, 2010.

Nicolai Meinshausen, Peter Bühlmann, et al. High-dimensional graphs and variable selection with the lasso. *The annals of statistics*, 34(3):1436–1462, 2006.

Michael H Neumann et al. Absolute regularity and ergodicity of poisson count processes. *Bernoulli*, 17(4):1268–1284, 2011.

K Ord, C Fernandes, and AC Harvey. Time series models for multivariate series of count data. *Ed. T. Subba Rao, Developments in Time Series Analysis*, pages 295–309, 1993.

Jack HW Penm and RD Terrell. On the recursive fitting of subset autoregressions. *Journal of Time Series Analysis*, 3(1):43–59, 1982.

Bijan Pesaran, Martin Vinck, Gaute Einevoll, Anton Sirota, Pascal Fries, Markus Siegel, Wilson Truccolo, Charles Schroeder, and Ramesh Srinivasan. Investigating large-scale brain dynamics using field potential recordings: analysis and interpretation. *Nature neuroscience*, 21(7):903–919, 2018.

Jonathan W Pillow, Jonathon Shlens, Liam Paninski, Alexander Sher, Alan M Litke, EJ Chichilnisky, and Eero P Simoncelli. Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature*, 454(7207):995–999, 2008.

Huitong Qiu, Fang Han, Han Liu, and Brian Caffo. Joint estimation of multiple graphical models from high dimensional time series. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(2):487–504, 2016.

Trevor Ruiz, Sharmodeep Bhattacharyya, Mahesh Balasubramanian, and Kristofer Bouchard. Sparse and low-bias estimation of high dimensional vector autoregressive models. *Proceedings of Machine Learning Research*, 120:55–64, 2020.

Jun Shao. Linear model selection by cross-validation. *Journal of the American statistical Association*, 88(422):486–494, 1993.

Ali Shojaie and George Michailidis. Discovering graphical granger causality using the truncating lasso penalty. *Bioinformatics*, 26(18):i517–i523, 2010.

Christopher A Sims. Macroeconomics and reality. *Econometrica: journal of the Econometric Society*, pages 1–48, 1980.

Song Song and Peter J Bickel. Large vector auto regressions. *arXiv preprint arXiv:1106.3915*, 2011.

Jitkomut Songsiri, Joachim Dahl, and Lieven Vandenberghe. Graphical models of autoregressive processes., 2010.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

Umberto Triacca, Alessandro Attanasio, and Antonello Pasini. Anthropogenic global warming hypothesis: testing its robustness by granger causality analysis. *Environmetrics*, 24(4):260–268, 2013.

Wilson Truccolo, Uri Eden, Matthew Fellows, John Donoghue, and Emery Brown. A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. *Journal of Neurophysiology*, 93(2): 1074–1089, 2005.

Ruey S Tsay. *Analysis of financial time series*, volume 543. John wiley & sons, 2005.

Pedro A Valdés-Sosa, Jose M Sánchez-Bornot, Agustín Lage-Castellanos, Mayrim Vega-Hernández, Jorge Bosch-Bayard, Lester Melie-García, and Erick Canales-Rodríguez. Estimating brain functional connectivity with sparse multivariate autoregression. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 360(1457):969–981, 2005.

Anita J van der Kooij. *Prediction accuracy and stability of regression with optimal scaling transformations*. PhD thesis, Lieden University, 2007.

Yuxiao Wang, Chee-Ming Ting, and Hernando Ombao. Modeling effective connectivity in high-dimensional cortical source signals. *IEEE Journal of Selected Topics in Signal Processing*, 10(7):1315–1325, 2016.

Stephen R Wassell. Superexponentiation and fixed points of exponential and logarithmic functions. *Mathematics Magazine*, 73(2):111–119, 2000.

Wing Hung Wong et al. Theory of partial likelihood. *The Annals of statistics*, 14(1): 88–123, 1986.

Scott L Zeger and Bahjat Qaqish. Markov regression models for time series: a quasi-likelihood approach. *Biometrics*, pages 1019–1031, 1988.

Ping Zhang. Model selection via multifold cross validation. *The Annals of Statistics*, pages 299–313, 1993.

Peng Zhao and Bin Yu. On model selection consistency of lasso. *Journal of Machine learning research*, 7:2541–2563, 2006.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2): 301–320, 2005.

APPENDICES

Appendix A: Algorithms

A.1 Coordinate descent for VAR(D) estimation

Weighted least squares with elastic net. The algorithms that follow for computing regularized MLEs for vector autoregressive models and generalized vector autoregressive models are all based on coordinate descent updates for the weighted least squares regression estimate with an elastic net penalty. This section presents this core update rule used in subsequent algorithms.

Consider estimation of the regression model

$$z = X\beta + \epsilon$$

where z is the response, X is the matrix of fixed covariate information, β is a vector of fixed but unknown parameters, and ϵ is a vector of uncorrelated random errors with mean zero.

The weighted least squares estimate of β is given by

$$\hat{\beta} = \text{argmin}_\beta \left\{ \frac{1}{2}(z - X\beta)'W(z - X\beta) \right\}$$

where W is a diagonal matrix of known weights. Denote the objective function by

$$f(\beta) = \frac{1}{2}(z - X\beta)'W(z - X\beta) \tag{A.1}$$

The weighted least squares estimate with an elastic net penalty is obtained by adding a penalty term $P(\beta)$ to the objective function

$$\hat{\beta} = \text{argmin}_\beta \left\{ \frac{1}{2}(z - X\beta)'W(z - X\beta) + \lambda P(\beta) \right\} \tag{A.2}$$

where $P(\beta) = \frac{1-\alpha}{2}\|\beta\|_2^2 + \alpha\|\beta\|_1$ and λ is a hyperparameter that controls the strength of the penalty. The hyperparameter α controls the relative balance of the L_2 (ridge) and L_1 (LASSO) terms in the penalty.

The coordinate descent approach to computing $\hat{\beta}$ in Equation (A.2) is to iteratively minimize the objective function in each β_j conditional on the current values of the other parameters β_k for $k \neq j$ until convergence. Each iteration can be accomplished by a simple update rule that can be equivalently derived by a number of approaches (Friedman et al., 2007, 2010; van der Kooij, 2007; Zou and Hastie, 2005). Here the rule is derived from the update given by Newton's method.

For the purpose of taking derivatives in β_j, the first portion of the objective function $f(\beta)$ can be rewritten as

$$f(\beta) = \frac{1}{2}\left(z - \sum_{k \neq j} x_k\beta_k - x_j\beta_j\right)' W \left(z - \sum_{k \neq j} x_k\beta_k - x_j\beta_j\right)$$

where x_j denotes the jth column of X. The first and second partial derivatives of

this portion with respect to β_j are

$$\begin{aligned}
\frac{\partial}{\partial \beta_j} f(\beta) &= -\left(z - \sum_{k \neq j} x_k \beta_k - x_j \beta_j\right)' W x_j \\
&= \beta_j x_j' W x_j - \left(z - \sum_{k \neq j} x_k \beta_k\right)' W x_j \\
&= \beta_j x_j' W x_j - \left(z - z^{(j)}\right)' W x_j \\
\frac{\partial^2}{\partial^2 \beta_j} f(\beta) &= x_j' W x_j
\end{aligned}$$

where $z^{(j)}$ denotes $\sum_{k \neq j} x_k \beta_k$. The quantity $(z - z^{(j)})$ is known as the jth partial residual.

Now, the derivatives of the penalty portion of the objective function are

$$\begin{aligned}
\frac{\partial}{\partial \beta_j}\left(\lambda \frac{1-\alpha}{2}\|\beta\|_2^2 + \lambda\alpha\|\beta\|_1\right) &= \lambda(1-\alpha)\beta_j + \lambda\alpha \frac{\partial}{\partial \beta_j}|\beta_j| \\
&= \begin{cases} \lambda(1-\alpha)\beta_j + \lambda\alpha & \text{, if } \beta_j > 0 \\ \lambda(1-\alpha)\beta_j - \lambda\alpha & \text{, if } \beta_j < 0 \\ \lambda(1-\alpha)\beta_j + \lambda\alpha\partial|\beta_j| & \text{, if } \beta_j = 0 \end{cases} \\
\frac{\partial^2}{\partial^2 \beta_j}\left(\lambda \frac{1-\alpha}{2}\|\beta\|_2^2 + \lambda\alpha\|\beta\|_1\right) &= \lambda(1-\alpha)
\end{aligned}$$

where $\partial|\beta_j|$ denotes a subgradient. Supposing the current estimate is $\hat{\beta}^{(k)}$, the Newton update for minimization in the jth coordinate is

$$\hat{\beta}_j^{(k+1)} \longleftarrow \hat{\beta}_j^{(k)} - \frac{\frac{\partial}{\partial \beta_j}\left(f(\beta) + \lambda P(\beta)\right)\big|_{\beta=\hat{\beta}^{(k)}}}{\frac{\partial^2}{\partial^2 \beta_j}\left(f(\beta) + \lambda P(\beta)\right)\big|_{\beta=\hat{\beta}^{(k)}}} \tag{A.3}$$

When articulated casewise according to whether $\hat{\beta}_j^{(k)} > 0$ or $\hat{\beta}_j^{(k)} < 0$ or $\hat{\beta}_j^{(k)} = 0$, Equation (A.3) gives the update rule (after minor algebraic simplification)

$$\hat{\beta}_j^{(k+1)} \longleftarrow \begin{cases} \frac{(z-z^{(j)})' W x_j - \lambda\alpha}{x_j' W x_j + \lambda(1-\alpha)} & \text{, if } \hat{\beta}_j^{(k)} > 0 \\ \frac{(z-z^{(j)})' W x_j + \lambda\alpha}{x_j' W x_j + \lambda(1-\alpha)} & \text{, if } \hat{\beta}_j^{(k)} < 0 \\ \frac{(z-z^{(j)})' W x_j - \lambda\alpha c}{x_j' W x_j + \lambda(1-\alpha)} & \text{, if } \hat{\beta}_j^{(k)} = 0 \end{cases}$$

where $c \in [-1, 1]$. Now if $|(z - z^{(j)})' W x_j| \leq \lambda\alpha$, the (sub)gradient in Equation (A.3) is not a descent direction, and in this case, the optimal value of the objective function is achieved by setting $\hat{\beta}_j^{(k+1)}$ to zero. Rewriting the conditions in terms of $(z - z^{(j)})' W x_j$ and dropping the iteration count k yields the update rule

$$\hat{\beta}_j \longleftarrow \begin{cases} \frac{(z-z^{(j)})' W x_j - \lambda\alpha}{x_j' W x_j + \lambda(1-\alpha)} & \text{, if } (z - z^{(j)})' W x_j > 0 \text{ and } |(z - z^{(j)})' W x_j| > \lambda\alpha \\ \frac{(z-z^{(j)})' W x_j + \lambda\alpha}{x_j' W x_j + \lambda(1-\alpha)} & \text{, if } (z - z^{(j)})' W x_j < 0 \text{ and } |(z - z^{(j)})' W x_j| > \lambda\alpha \\ 0 & \text{, if and } |(z - z^{(j)})' W x_j| \leq \lambda\alpha \end{cases}$$

Finally, the update can be re-expressed in terms of the soft-threshold function

$$S(x, y) = \text{sign}(x)(|x| - y)_+ = \begin{cases} x - y & \text{, if } x > 0 \text{ and } |x| > y \\ x + y & \text{, if } x < 0 \text{ and } |x| > y \\ 0 & \text{, if } |x| \leq y \end{cases}$$

compactly as the coordinate update

$$\hat{\beta}_j \longleftarrow \frac{S\left((z - z^{(j)})' W x_j, \lambda\alpha\right)}{x_j' W x_j + \lambda(1-\alpha)}$$

The coordinate descent algorithm using this update rule and a gradient norm stopping criterion is given as Algorithm A.1.

Algorithm A.1 Coordinate descent algorithm for computing the weighted least squares estimate with an elastic net penalty.

Require:
Data z, X; hyperparameters λ, α; and convergence threshold δ
Initialize $\hat{\beta}$
while $\|\nabla f(\hat{\beta})\|_2 > \delta$ **do**
for $j = 1, \ldots, J$ **do**
$z^{(j)} \longleftarrow \sum_{k \neq j} x_k \hat{\beta}_k$
$\hat{\beta}_j \longleftarrow \frac{S\left((z - z^{(j)})' W x_j, \lambda\alpha\right)}{x_j' W x_j + \lambda(1-\alpha)}$
end for
end while
Ensure: $\hat{\beta}$

VAR estimation. Algorithm A.1 can be applied directly to estimation of a VAR model after an appropriate vectorization transformation of the data. However, a slight modification of the algorithm results in more efficient computations. First the data transformation is described, followed by the modification strategy.

In Chapter 2, Equation (2.2) expresses the VAR(D) model as a multivariate mul-

tiple regression $Y = UB + E$ where

$$Y = \begin{bmatrix} x'_T \\ x'_{T-1} \\ \vdots \\ x'_D \end{bmatrix} \quad U = \begin{bmatrix} 1 & x'_{T-1} & \cdots & x'_{T-D} \\ 1 & x'_{T-2} & \cdots & x'_{T-D-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x'_{D-1} & \cdots & x'_0 \end{bmatrix} \quad B = \begin{bmatrix} \nu' \\ A'_1 \\ \vdots \\ A'_D \end{bmatrix} \quad E = \begin{bmatrix} \epsilon'_T \\ \epsilon'_{T-1} \\ \vdots \\ \epsilon'_D \end{bmatrix}$$

This can be further expressed as a univariate regression $\mathcal{Y} = \mathcal{X}\beta + \mathcal{E}$ where the mean portion and response are given by

$$\mathcal{Y} \stackrel{\text{def}}{=} \text{vec}Y \,, \quad \mathcal{X} \stackrel{\text{def}}{=} U \otimes I_M \,, \quad \text{and } \beta \stackrel{\text{def}}{=} \text{vec}B \tag{A.4}$$

Now, a simple strategy for computing the LASSO estimate of $\nu, A_1, \ldots, A_D$, given by

$$\hat{\beta} = \text{argmin}_\beta \left\{ \|\mathcal{Y} - \mathcal{X}\beta\|_2^2 + \lambda \sum_{d=1}^{D} \sum_{i,j} |a_{dij}| \right\}$$

is to apply Algorithm A.1 directly with $X = \mathcal{X}, z = \mathcal{Y}, \alpha = 0, W = I$, and setting $\lambda = 0$ in the update rule whenever $\beta_j = \nu_k$, so as to avoid applying the penalty to the intercept. However, the matrix $\mathcal{X}$ is a block-diagonal matrix comprising M copies of U, and the corresponding blocks of $\mathcal{Y}$ are the univariate series $x_m = (x_{mT}\ x_{m(T-1)} \cdots x_{mD})'$

$$\mathcal{Y} = \begin{bmatrix} x_1 \\ \hdashline x_2 \\ \hdashline \vdots \\ \hdashline x_M \end{bmatrix} \qquad \mathcal{X} = \begin{bmatrix} U & 0 & \cdots & 0 \\ \hdashline 0 & U & \cdots & 0 \\ \hdashline \vdots & \vdots & \ddots & \vdots \\ \hdashline 0 & 0 & \cdots & U \end{bmatrix}$$

As a result, computation and storage of $\mathcal{X}$ occupies unnecessary memory, so direct application of Algorithm A.1 to $\mathcal{X}$ and $\mathcal{Y}$ is inefficient.

Due to the sparsity pattern in $\mathcal{X}$, the computations involved in the update rule for the lth coordinate are given under index transformations by operations involving only U and Y. Specifically, denoting the row and column dimensions of U as $N = T - D$ and $P = DM + 1$ and the row and column dimensions of $\mathcal{X}$ as $K = NM$ and $L = M(DM+1)$, one has

$$\left(\mathcal{Y} - \mathcal{Y}^{(l)}\right)' \mathcal{X}_l = \left(x_m - x_m^{(p)}\right)' u_p$$
$$\mathcal{X}_l' \mathcal{X}_l = u_p' u_p$$

where the index transformation is given by

$$(n, m) = \left(k\text{mod}M + M\mathbb{1}\{k\text{mod}M = 0\}, \frac{k - k\text{mod}M}{M} + \mathbb{1}\{k = k\text{mod}M\}\right)$$
$$(p, m) = \left(l\text{mod}P + P\mathbb{1}\{l\text{mod}P = 0\}, \frac{l - l\text{mod}P}{P} + \mathbb{1}\{l = l\text{mod}P\}\right)$$

The computation of $x_{mn}^{(p)} = \sum_{j \neq p} u_{nj} b_{jm}$ depends only on the mth column of B.

Consequently, a more efficient strategy for computing the LASSO estimate for a VAR(D) model is to divide the coordinate iterations into blocks corresponding to the blocks of B and compute the update rules using data stored in the formats specified by U and Y. A slight ridge penalty is included for numerical stability. This strategy is given as Algorithm A.2.

Algorithm A.2 Coordinate descent algorithm for computing the LASSO VAR(D) estimate with blockwise coordinate updates.

Require:
Data Y, U, hyperparameters λ, α, and convergence threshold δ
Initialize $\hat{B}$
while $\|\nabla f(\hat{B})\|_2 > \delta$ **do**
for $m = 1, \ldots, M$ **do**
for $p = 1, \ldots, DM + 1$ **do**
$x_{mn}^{(p)} \longleftarrow \sum_{j \neq p} u_{nj} \hat{b}_{jm}$ for $n = 1, \ldots, N$

$$\hat{b}_{pm} \longleftarrow \begin{cases} \frac{\left(x_m - x_m^{(p)}\right)' u_p}{u_p' u_p} & \text{, if } p = 1 \\ \frac{S\left(\left(x_m - x_m^{(p)}\right)' u_p, \lambda\alpha\right)}{u_p' u_p + \lambda(1-\alpha)} & \text{, if } p > 1 \end{cases}$$

end for
end for
end while
Ensure: $\hat{B}$

A.2 Coordinate descent for GVAR(D) estimation

The strategy above can be extended directly to estimation of generalized vector autoregressive models since the data transformations and predictor structure exhibit analogous sparsity patterns. This section describes the coordinate descent algorithm for computing coefficient estimates in univariate GLMs with an elastic net penalty, and then restates the blockwise update strategy from the previous section in this case.

Penalized IRLS for univariate generalized linear models. Consider the univariate generalized linear model: a response vector y and covariate information X where

$$y_i \overset{indep.}{\sim} p(\cdot\,; \theta_i) \quad \text{and} \quad g(\mathbb{E}y) = g(\mu) = \eta = X\beta$$

where p is an exponential family density of the form

$$p(y_i;\theta_i) \propto \exp\left\{\frac{y_i\theta_i - b(\theta_i)}{a(\phi_i)}\right\}$$

Assume that g is the canonical link function that results from equating $\eta = \theta$. The log-likelihood as a function of the linear predictor η is, up to a constant,

$$\ell(\eta; x, y) = \sum_{i=1}^{n}\left(\frac{y_i\eta_i - b(\eta_i)}{a(\phi_i)}\right)$$

The classical technique for unpenalized maximum likelihood estimation maximizes $\ell(\eta; x, y)$ iteratively by:

1. computing an approximation $\tilde{\ell}$ to the log-likelihood from current estimates;
2. maximizing the approximation $\tilde{\ell}$ to get new estimates;
3. repeating 1 - 2 until the estimates converge.

Conveniently, the approximation $-\tilde{\ell}$ takes the form of the loss function for a weighted least squares problem for which a closed form solution is available, which eases calculations in step 2.

Consider now computing the penalized MLE

$$\hat{\beta} = \text{argmin}_\beta \{-\ell(\eta; x, y) + \lambda P(\beta)\} \tag{A.5}$$

The same strategy for unpenalized estimation is applicable with the penalty, though generally iterative methods will replace closed form solutions for the optimization

step (step 2). The objective function $-\ell(\eta; x, y) + \lambda P(\beta)$ can be approximated by the loss function for a penalized weighted least squares problem. For the elastic net penalty, the approximation can be maximized with respect to β via Algorithm A.1. The penalized MLE in Equation (A.5) with the elastic net penalty can therefore be found iteratively by:

1. computing the approximation $-\tilde{\ell}(\eta; x, y) + \lambda P(\beta)$ to the (negative) penalized log-likelihood from current estimates;

2. maximizing the approximation $-\tilde{\ell}(\eta; x, y) + \lambda P(\beta)$ with respect to one coordinate via the update rule in Algorithm A.1 to get new estimates;

3. repeating 1 - 2 for each coordinate and cycling through the coordinates until the estimates converge.

This procedure is given as Algorithm A.3.

Algorithm A.3 Coordinatewise IRLS algorithm for estimation of generalized linear models with an elastic net penalty.

Require: data x, y; hyperparameters λ, α; convergence threshhold δ
  Initialize $\hat{\beta}$
  while $\|\nabla \ell(\hat{\eta}; x, y)\|_2 > \delta$ **do**
    for $j = 1, \ldots, J$ **do**
      $W \longleftarrow \left(\text{diag}(g'(\hat{\mu}))\right)^{-1}$
      $z \longleftarrow x\hat{\beta} - (y - \hat{\mu})W^{-1}$
      $z^{(j)} \longleftarrow \sum_{k \neq j} x_k \hat{\beta}_k$
      $\hat{\beta}_j \longleftarrow \frac{S\left((z - z^{(j)})' W x_j, \lambda\alpha\right)}{x_j' W x_j + \lambda(1-\alpha)}$
    end for
  end while
Ensure: $\hat{\beta}$

The update rules for z and W in Algorithm A.3 are derived from a Taylor expansion around current estimates. Let $\hat{\eta}$ denote the linear predictor from the current estimates $\hat{\beta}$, $\ell(\eta)$ denote the log-likelihood (suppressing the data arguments), v and W denote the gradient and Hessian of $\ell(\eta)$. A second-order Taylor expansion of the log-likelihood about $\hat{\eta}$ gives

$$\begin{aligned}
\ell(\eta) &\approx \ell(\hat{\eta}) + (\eta - \hat{\eta})'\ell'(\hat{\eta}) - \frac{1}{2}(\eta - \hat{\eta})'\ell''(\hat{\eta})(\eta - \hat{\eta}) \\
&= C + (\eta - \hat{\eta})'\ell'(\hat{\eta}) - \frac{1}{2}(\eta - \hat{\eta})'\ell''(\hat{\eta})(\eta - \hat{\eta}) \\
&= C + (\eta - \hat{\eta})'v - \frac{1}{2}(\eta - \hat{\eta})'W(\eta - \hat{\eta}) \\
&= C^* - \frac{1}{2}(\eta - \hat{\eta} - vW^{-1})'W(\eta - \hat{\eta} - vW^{-1}) \\
&= C^* - \frac{1}{2}(\eta - z)'W(\eta - z)
\end{aligned}$$

with $z = \hat{\eta} - vW^{-1}$. Now W and z depend on the current estimates and can be derived in general for any GLM. Since by hypothesis $\mu_i = b'(\theta_i) = g^{-1}(\eta_i)$,

$$\begin{aligned}
\frac{\partial \ell}{\partial \eta_i} &= \left(\frac{y_i - b'(\theta_i)}{a_i(\phi)}\right)\frac{\partial \theta_i}{\partial \eta_i} \\
&= \left(\frac{y_i - b'(\theta_i)}{a_i(\phi)}\right)\frac{\partial \theta_i}{\partial \mu_i}\frac{\partial \mu_i}{\partial \eta_i} \\
&= \left(\frac{y_i - b'(\theta_i)}{a_i(\phi)}\right)\frac{1}{g'(\mu_i)}\frac{1}{b''(\theta_i)} \\
&= (y_i - \mu_i)\left(a_i(\phi)g'(\mu_i)b''(\theta_i)\right)^{-1}
\end{aligned}$$

When $a_i(\phi) = 1$ and g is the canonical link, $g'(\mu_i) = \frac{\partial \eta_i}{\partial \mu_i}$ and $b''(\theta_i) = \frac{\partial \mu_i}{\partial \theta_i}$, so

$$\frac{\partial \ell}{\partial \eta_i} = y_i - \mu_i$$

and

$$\frac{\partial^2 \ell}{\partial \eta_i \partial \eta_j} = \begin{cases} -\frac{1}{g'(\mu_i)} & i = j \\ 0 & i \neq j \end{cases}$$

So $W = \text{diag}(1/g'(\mu))$ and $v = (y_1 - \mu_1 \cdots y_N - \mu_N)$. Therefore, given current estimates, updating the approximation amounts to computing

$$z = X'\hat{\beta} - (y - \hat{\mu})W^{-1} \tag{A.6}$$

$$W = (g'(\hat{\mu}))^{-1} \tag{A.7}$$

Equations (A.6) and (A.7) give the updates that appear in the outer while loop of Algorithm A.3.

GVAR(D) estimation. Algorithm A.3 can be modified for efficient estimation of GVAR(D) models with an elastic net penalty using a blockwise update strategy for the inner while loop. The modification bears an identical relationship to Algorithm A.3 as Algorithm A.2 bears to Algorithm A.1.

The GVAR(D) model for data $\{x_t \in \mathbb{R}^M\}_{t=0}^T$ is the random process $\{X_t \in \mathbb{R}^M\}_{t=0}^T$ characterized by the conditional mean structure

$$g\left(\mathbb{E}\left(X_t \mid X_{t-1} = x_{t-1}, \ldots, X_{t-D} = x_{t-D}\right)\right) = \nu + \sum_{d=1}^{D} A_d x_{t-d}$$

under the assumption that for each $m = 1, \ldots, M$ and each t

$$X_{t,m} \mid X_{t-1} = x_{t-1}, \ldots, X_{t-D} = x_{t-D} \overset{indep.}{\sim} p\left(\cdot \,\middle|\, \theta_{t,m} = \nu_m + \sum_{d=1}^{D} a'_{d,m} x_{t-d} \right)$$

where $p(\cdot|\theta)$ is an exponential family density and g is the canonical link so that $\eta_{t,m} = \theta_{t,m} = g(\mu_{t,m}) = \mathbb{E}(X_{t,m}|X_{t-1}, \ldots, X_{t-D}))$. Denoting the data by $X = (X_0 \cdots X_T)$ log-likelihood by $\ell(B; X)$, the penalized maximum likelihood estimate of B is

$$\hat{B} = \arg\min_B \{-\ell(B; X) + \lambda P(B)\}$$

where the explicit form of the likelihood is given in Chapter 3 as Equation (3.3). Here let P be the elastic net penalty applied only to the elements of $A_1, \ldots, A_D$

$$P(B) = \frac{1-\alpha}{2} \sum_{d,i,j} a_{dij}^2 + \alpha \sum_{d,i,j} |a_{dij}|$$

An iterative estimation algorithm based on Algorithm A.3 is derived below by vectorizing the problem and developing a blockwise update strategy analogous to the VAR(D) case (Algorithm A.2).

The GVAR(D) model can be expressed as a multivariate generalized regression model

$$g\left(\mathbb{E}(Y|U)\right) = g\left(\mu\right) = \eta = UB$$

expressed in terms of Y, U, B where

$$Y = \begin{bmatrix} X_T' \\ X_{T-1}' \\ \vdots \\ X_D' \end{bmatrix} \quad U = \begin{bmatrix} 1 & X_{T-1}' & \cdots & X_{T-D}' \\ 1 & X_{T-2}' & \cdots & X_{T-D-1}' \\ \vdots & \vdots & \ddots & \vdots \\ 1 & X_{D-1}' & \cdots & X_0' \end{bmatrix} \quad B = \begin{bmatrix} \nu' \\ A_1' \\ \vdots \\ A_D' \end{bmatrix}$$

Analogous to the VAR(D) case, the core model relationship can be expressed in univariate terms by applying the vectorization transformations given in Equation (A.4), with the result

$$\mathcal{Y} = \begin{bmatrix} x_1 \\ \hdashline x_2 \\ \hdashline \vdots \\ \hdashline x_M \end{bmatrix} \qquad \mathcal{X} = \left[\begin{array}{c:c:c:c} U & 0 & \cdots & 0 \\ \hdashline 0 & U & \cdots & 0 \\ \hdashline \vdots & \vdots & \ddots & \vdots \\ \hdashline 0 & 0 & \cdots & U \end{array}\right] \qquad \beta = \text{vec}(B)$$

where $x_m = (x_{0,m} \cdots x_{T,m})'$ denote the univariate series in each component so that

$$g\left(\mathbb{E}(\mathcal{Y}|\mathcal{X})\right) = g\left(\text{vec}(\mu)\right) = \text{vec}(\eta) = \mathcal{X}\beta$$

It is straightforward to verify by inspection that if the the elements of $\mathcal{Y}$ are assumed to be Poisson conditional on $\mathcal{X}$ with the corresponding mean, the likelihoods of β and B are identical, *i.e.*,

$$\ell(\beta; \mathcal{Y}, \mathcal{X}) = \ell(B; Y, U)$$

since the likelihood contributions from each $x_{t,m}$ are exactly the same. Therefore, the estimate $\hat{B}$ can be recovered from the estimate

$$\hat{\beta} = \text{argmin}_{\beta} \{-\ell(\beta; \mathcal{Y}, \mathcal{X}) + \lambda P^*(\beta)\}$$

where $P^*(\beta) = P(B)$. Now, following the discussion in the previous section, due to the sparsity pattern of $\mathcal{X}$ the key operations in coordinate updates that result from directly applying Algorithm A.3 to the vectorized data $\mathcal{X}$ and $\mathcal{Y}$ can be expressed in terms of U and X. The result is given as Algorithm A.4, following the index conventions used in Algorithm A.2.

Algorithm A.4 Coordinatewise IRLS algorithm for computing LASSO GVAR(D) estimates with blockwise updates.

Require:
Data Y, U, hyperparameters λ, α, and convergence threshold δ
Initialize $\hat{B}$
while $\|\nabla \ell(\hat{B}; X)\|_2 > \delta$ **do**
for $m = 1, \ldots, M$ **do**
for $p = 1, \ldots, DM + 1$ **do**
$W_m \longleftarrow (\text{diag}(g'(\hat{\mu}_m))^{-1}$
$z_m \longleftarrow U\hat{b}_m - (x_m - \hat{\mu}_m)W_m^{-1}$
$z_{mn}^{(p)} \longleftarrow \sum_{j \neq p} u_{nj}\hat{\beta}_j$ for $n = 1, \ldots, N$
$\hat{b}_{pm} \longleftarrow \begin{cases} \frac{\left(z_m - z_m^{(p)}\right)' W_m u_p}{u_p' W_m u_p} & \text{, if } p = 1 \\ \frac{S\left(\left(z_m - z_m^{(p)}\right)' W_m u_p, \lambda\alpha\right)}{u_p' W_m u_p + \lambda(1-\alpha)} & \text{, if } p > 1 \end{cases}$
end for
end for
end while
Ensure: $\hat{B}$

Appendix C: Computation

This appendix describes scaling experiments that explore the effect of dataset size on runtimes in a distributed computing environment and identify computational bottlenecks for execution of the UoI-VAR algorithm.

Overview of scaling experiments. The scalability of the UoI-VAR method was explored in empirical experiments using a distributed implementation of the algorithm. One set of experiments explored weak scaling: the effect of simultaneous increase of data dimensions and computation cores on runtime. Another explored strong scaling: the effect of increasing computation cores on runtime given fixed data dimensions. Scaling in these experiments is reported as a function of the memory required to store $\mathcal{Y}$ and $\mathcal{X}$ arising from time series data; the dimension M of the synthetic input data $x_t \in \mathbb{R}^M{}_{t=0}^T$ used in the experiments varied between $M = 356$ and $M = 1000$ and was adjusted with $T = 2M$ to yield target memory requirements.

For these experiments, a distributed implementation of the UoI-VAR method was developed with bootstrap-level and regularization path parallelisms in C++ using Message Passing Interface (MPI) for inter-nodal communication and the program was executed on Cori Knight's Landing supercomputer at Lawrence Berkeley National Laboratory. It was found that generating data for bootstrap samples in both the intersection and union steps can be quite challenging because of immense inter-nodal communication; a data distribution strategy developed specifically for the UoI

algorithmic framework was used to address this challenge (Balasubramanian et al., 2020).

The implementation that exploits regularization path parallelism and computes LASSO estimates in a distributed fashion via the Alternating Direction Method of Multipliers (ADMM) algorithm (Boyd et al., 2011) using publicly available codes. An important consequence of this strategy for computing LASSO estimates is that it dramatically increases memory requirements for LASSO and least squares computations within the UoI-VAR algorithm relative to the coordinate descent approach described in Appendix A. The ADMM codes require explicitly constructing the vectorized quantities $\mathcal{Y}^*$ and $\mathcal{X}^* = U^* \otimes I_M$ and rely on sparse matrix operations to improve the efficiency of subsequent computations. While the Kronecker product calculation is costly and the result is a potentially very large object that contains no information over and above U^*, the advantage is that regularization path parallelism can be easily implemented. By contrast, the coordinate descent codes avoid this operation and compute solutions using just U^*, but proceed in serial across the regularization path. This is not a strictly necessary tradeoff, and a manual implementation of ADMM adjusted for the VAR problem could exploit the additional level of parallelism without increasing memory requirements.

Weak scaling. Figure C.1 shows the runtime performance of the algorithm on synthetic datasets under weak scaling, *i.e.*, under increases in dataset size concurrent with proportional increases in the number of computation cores. In this scaling experiment, the total runtime performance is divided into computation, communication,

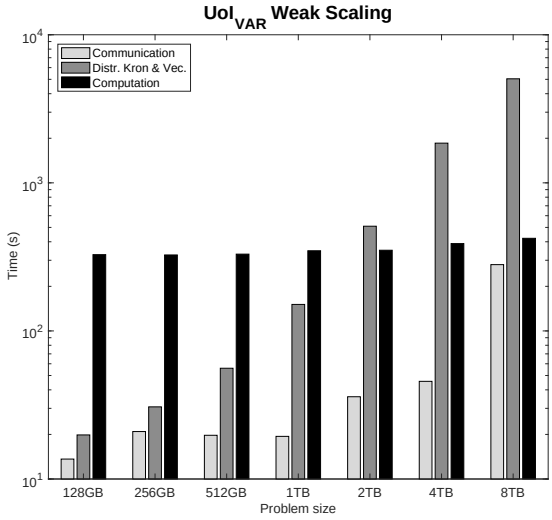


Figure C.1: Runtime analysis of UoI-VAR under weak scaling (computation cores increase with data dimensions). Total runtime for the method is divided into computation, communication, and data distribution, and runtimes associated with each division are plotted on a logarithmic scale against the memory required to store the data.

and distribution times: computation time is associated with calculating LASSO and OLS estimates on bootstrap samples; communication time is associated with MPI calls in the program; and distribution time is dominated by distribution of the bootstrap data quantities $\mathcal{Y}^*, \mathcal{X}^*$ to computation nodes. These three times are plotted against problem size in the figure on a logarithmic scale. As the problem size increases, LASSO computation runtime remains nearly constant, but the distribution runtime increases exponentially and the communication runtime increases above a problem size of 1TB. This reflects that for relatively smaller data dimensions, computation dominates total runtime, but above a certain threshold, distribution (and to a lesser extent communication) dominate runtime. Further, the exponential in-

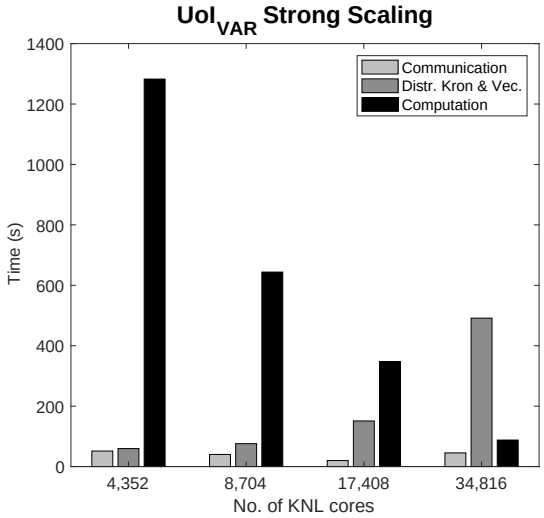


Figure C.2: Runtime analysis of UoI-VAR under strong scaling (computation cores increase but data dimensions remain fixed). Total runtime for the method is divided into computation, communication, and data distribution, and runtimes associated with each division are plotted on a logarithmic scale against the memory required to store the data.

crease in distribution runtimes indicates that the implementation strategy involving explicit construction of $\mathcal{Y}$ and $\mathcal{X}$ creates a computational bottleneck that becomes problematic for massive data.

Strong scaling. A second set of experiments explored strong scaling on synthetic data by recording computation, communication, and distribution runtimes on a 1TB dataset under increases — specifically, consecutive doublings — in the number of computation cores available. Figure C.2 shows each runtime plotted on a linear scale against number of available cores. The results indicate a decrease in computation time and an increase in distribution time. This suggests that the computation bottleneck associated with distribution time due to explicit construction of $\mathcal{Y}$ and $\mathcal{X}$ can be

mitigated to some extent by limiting the number of cores: for the 1TB dataset in this particular experiment, a number of cores around roughly 18,000 (near the third set of times assessed using 17,408 cores) achieves a reasonable balance between computation and distribution time. More broadly put, choosing a large but moderate number of cores relative to data dimensions can achieve substantial efficiency in computation time without exacerbating distribution time.

Summary of findings. The scaling experiments indicate two important findings. First, above certain dataset dimensions, distribution time associated with vectorizing the bootstrap samples (*i.e.*, forming $\mathcal{X}^*$ and $\mathcal{Y}^*$) to compute estimates dominates runtimes. Second, for a fixed data dimension, distributing computations across too few cores prolongs computation time but limits distribution time, and distributing computations across too many cores prolongs distribution times but diminishes computation time. Both findings indicate that distribution time is the main computational bottleneck for algorithm execution. This result suggests more broadly that the vectorization strategy has its limitations. Based on the weak scaling experiments, the strategy can be expected to fail in applications involving massive data due to the exponential increase in distribution time. It is worth highlighting that although this finding was observed in the context of UoI-VAR runtimes, the bottleneck identified arises from vectorization, which is a common general strategy for computing VAR model estimates (of any type) using regression methods (of any type). In other words, the limitation observed is not specific to the UoI-VAR method, and in fact is much more general. This suggests that for VAR estimation in large-scale applica-

tions, it is essential to develop efficient computation strategies as alternatives to direct implementation of regression methods. The modified pathwise coordinate descent algorithm developed in Appendix XX provides an example of this kind of approach.

Appendix S: Supplementary Figures

Supplementary figures for Chapter 2

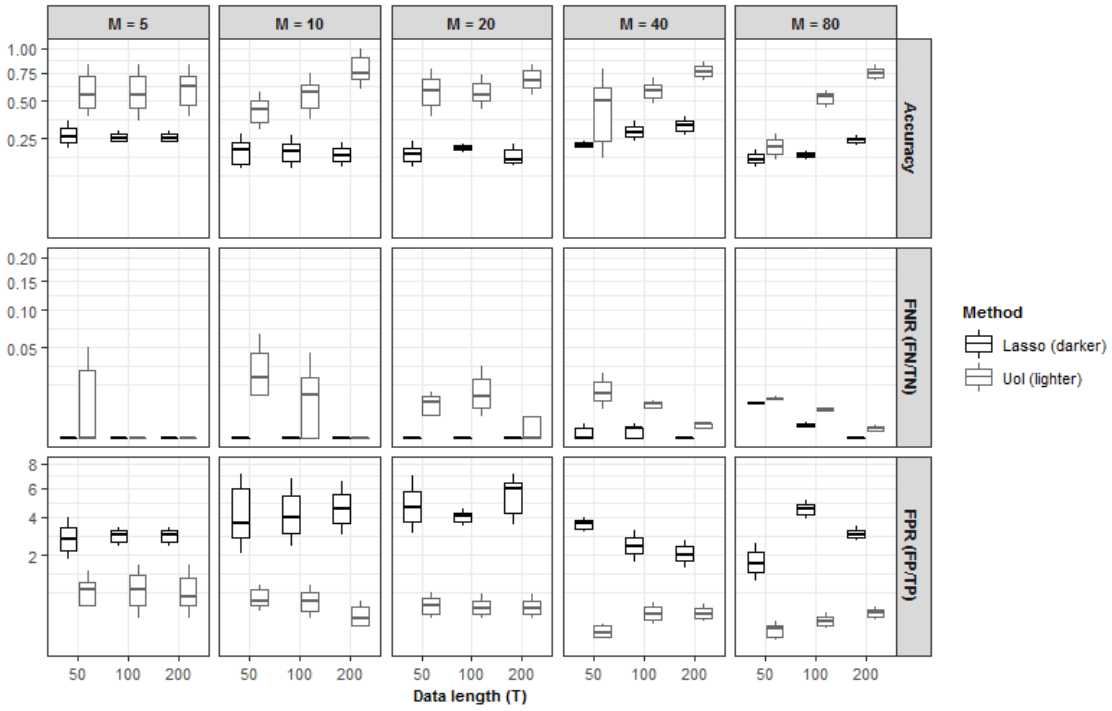


Figure S.1: Selection behavior detail for UoI-VAR and LASSO estimators observed in the simulation study of Section 2.4.1. Panel rows distinguish selection accuracy, false negative rates, and false positive rates; panel columns distinguish dimensions M; and in each panel boxplots for the row metric for each estimator are plotted against time series length T on the horizontal axis.

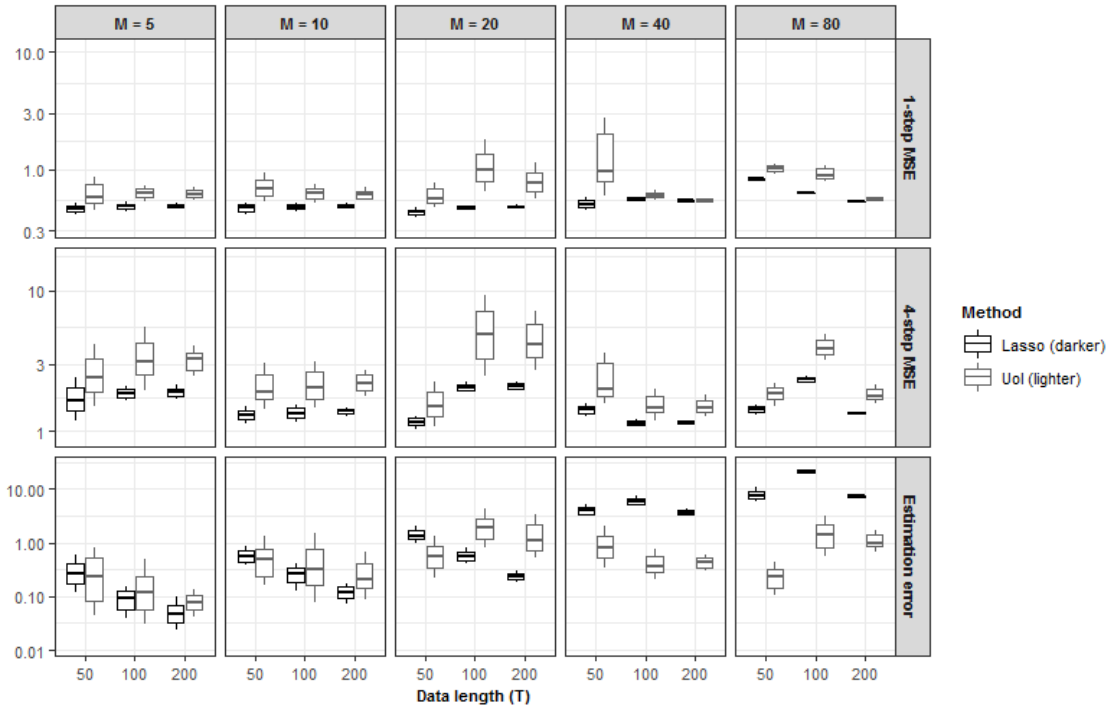


Figure S.2: Forecast and estimation behavior detail for UoI-VAR and LASSO estimators observed in the simulation study of Section 2.4.1. Panel rows distinguish one-step forecast errors, four-step forecast errors, and estimation error; panel columns distinguish dimensions M; and in each panel boxplots for the row metric for each estimator are plotted against time series length T on the horizontal axis.

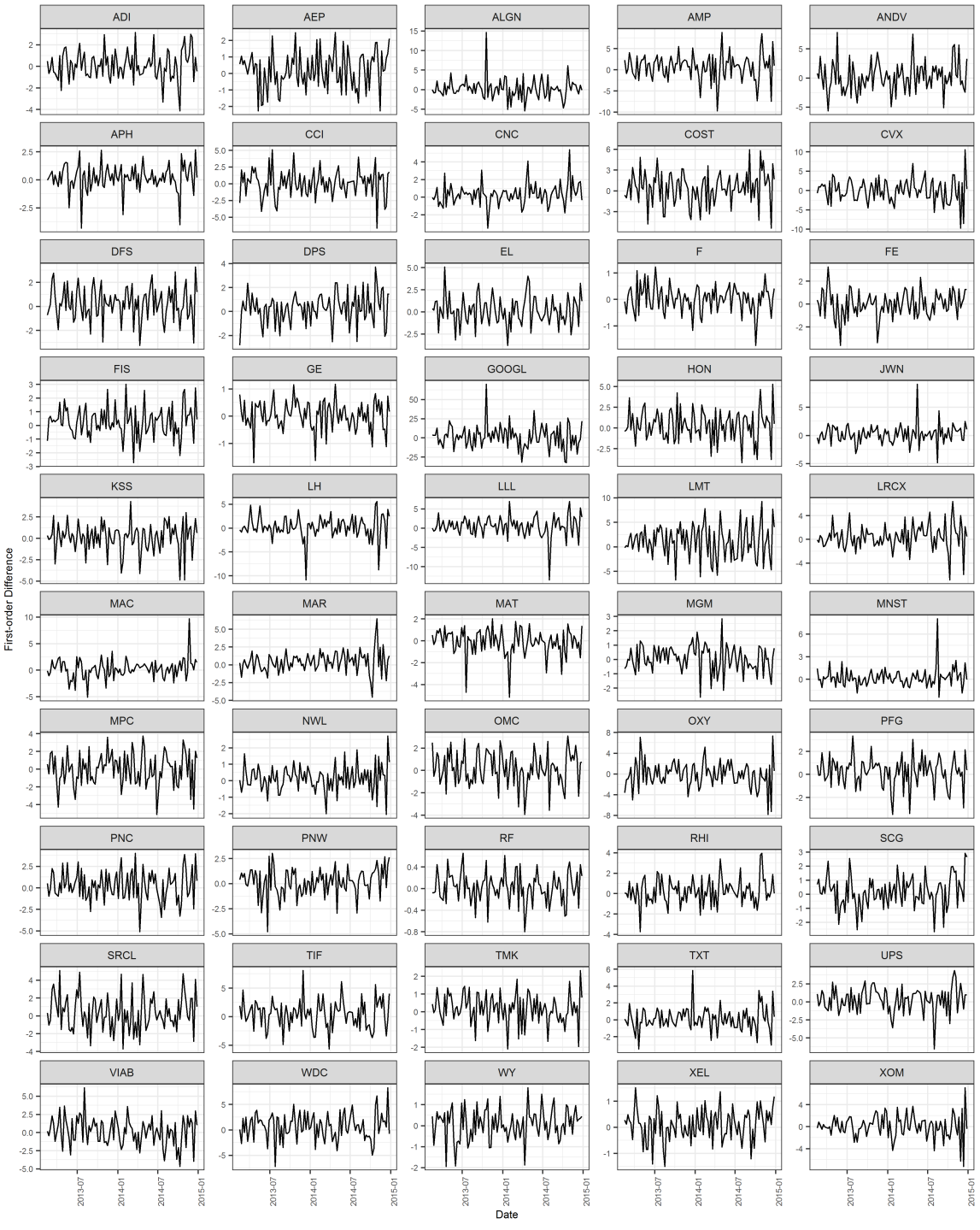


Figure S.3: First differences of S&P dataset analyzed in the example causal analysis application in Section 2.4.2. Each panel shows first differences of weekly closes of one of 50 randomly chosen publicly traded companies listed on the S&P 500 index in 2013-2014.

Supplementary figures for Chapter 4

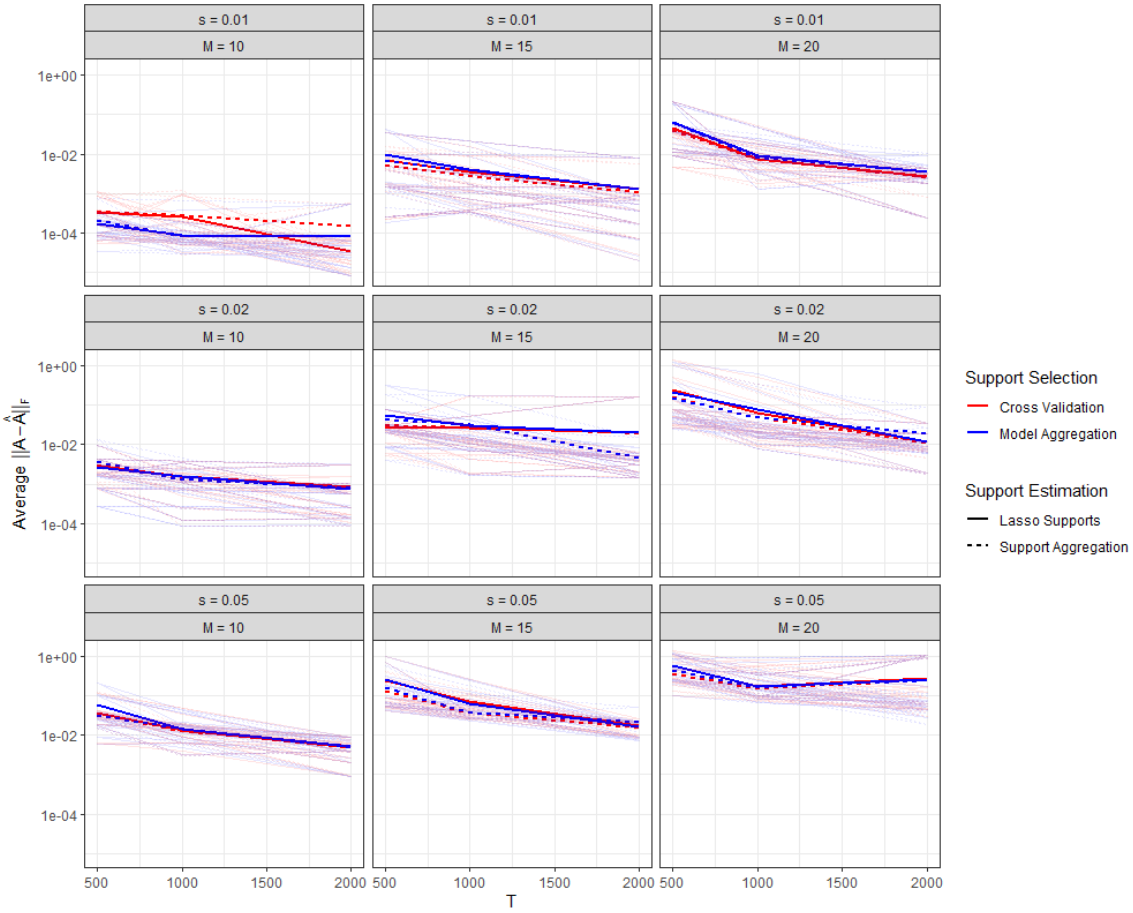


Figure S.4: Comparison of estimation error between combinations of support selection and support estimation methods observed in the simulation of Section 4.3. For each combination of sparsity s (panel rows) and dimension M (panel columns) in the simulation, estimation error $\|A - \hat{A}\|_F$ for each of the estimates computed per method is plotted against realization length T. The bold lines are average estimation errors per method taken across each of the five simulated datasets for each of the ten generated parameter matrices per simulation setting.

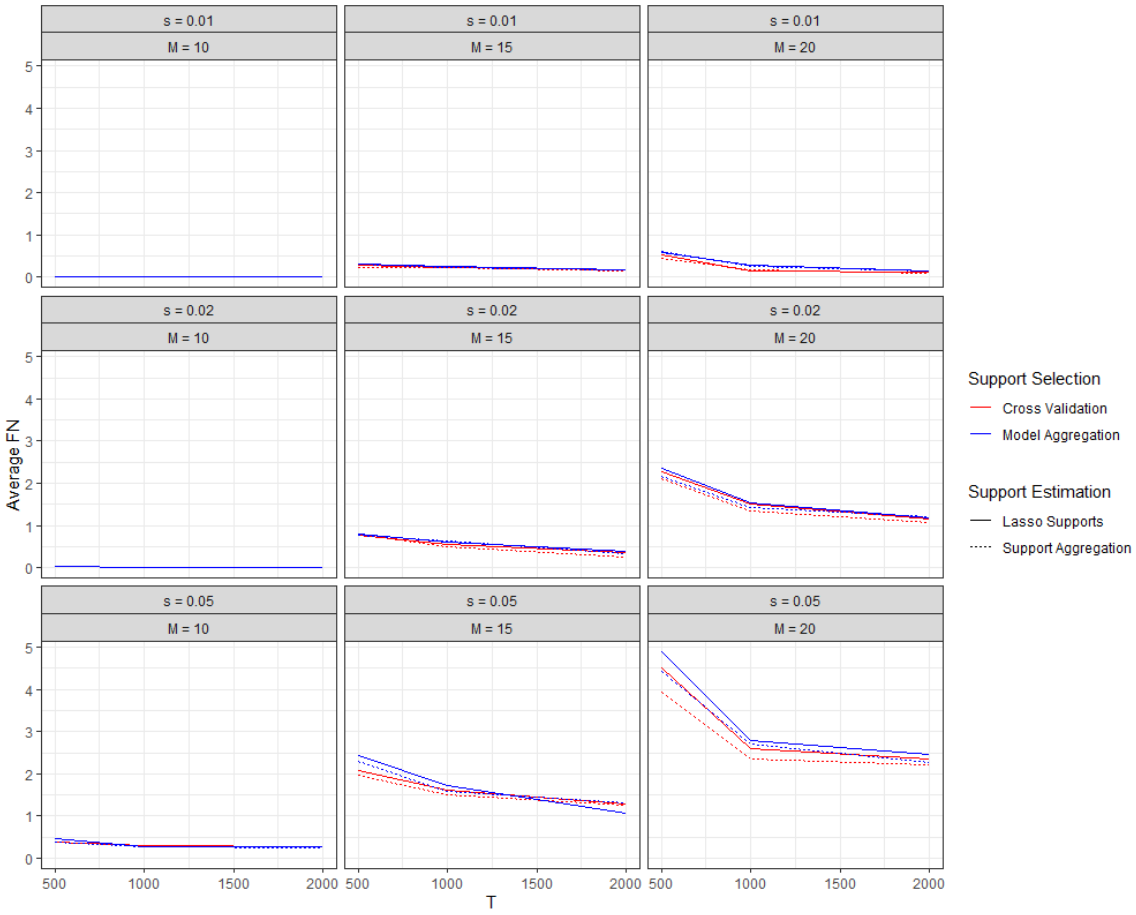


Figure S.5: Average false negative counts for each combination of support selection and support estimation method observed in the simulation of Section 4.3. For a given estimate $\hat{A}$ of A, the false negative count is the number of nonzero parameters in A that are estimated as zero in $\hat{A}$; or, expressed in terms of the true and estimated support sets, the quantity $|S \setminus \hat{S}|$. Average false negative counts are computed for each combination of sparsity s (panel rows), dimension M (panel columns), and realization length T (horizontal axis), and taken across each of the five simulated datasets for each of the ten generated parameter matrices per combination.

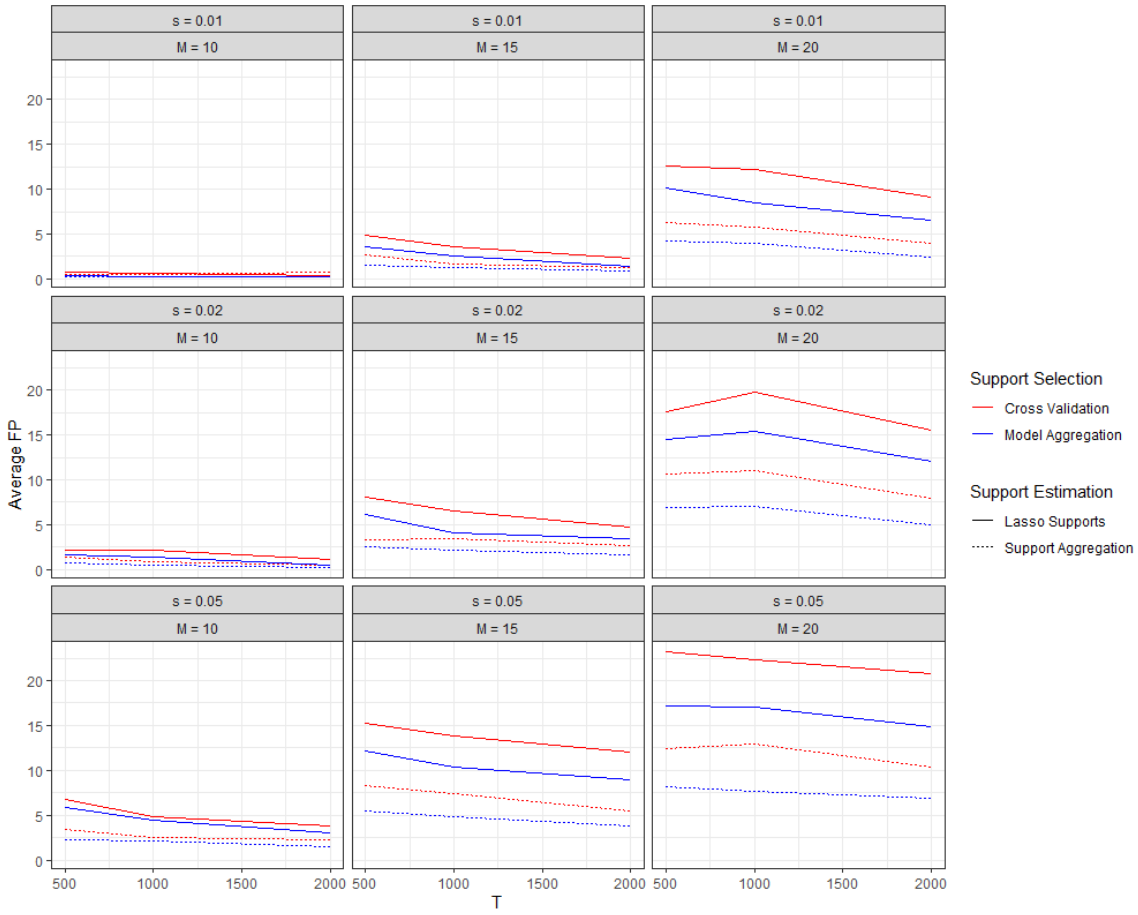


Figure S.6: Average false positive counts for each combination of support selection and support estimation method observed in the simulation of Section 4.3. For a given estimate $\hat{A}$ of A, the false positive count is the number of zero-valued parameters in A that are estimated as nonzero in $\hat{A}$; or, expressed in terms of the true and estimated support sets, the quantity $|\hat{S} \setminus S|$. Average false negative counts are computed for each combination of sparsity s (panel rows), dimension M (panel columns), and realization length T (horizontal axis), and taken across each of the five simulated datasets for each of the ten generated parameter matrices per combination.