

sstvars: Structural Smooth Transition Vector Autoregressive Models in R

Savi Virolainen
University of Helsinki

Abstract

We describe the R package **sstvars**, which provides tools for estimating and analyzing the reduced form and structural smooth transition vector autoregressive (STVAR) models. The package implements various transition weight functions, conditional distributions, identification methods, and parameter restrictions. The model parameters are estimated with the method of maximum likelihood by running multiple rounds of a two-phase estimation procedure in which a genetic algorithm is used to find starting values for a gradient based method. For evaluating the adequacy of the estimated models, **sstvars** utilizes residuals based diagnostics and provides functions for graphical diagnostics and for calculating formal diagnostic tests. **sstvars** also accommodates the estimation of linear impulse response functions, nonlinear generalized impulse response functions, and generalized forecast error variance decompositions. Further functionality includes hypothesis testing, plotting the profile log-likelihood functions about the estimate, simulation from STVAR processes, and forecasting, for example. We illustrate the use of **sstvars** with a quarterly series consisting of two U.S. variables: the percentage change of real GDP and the percentage change of GDP implicit price deflator.

Keywords: smooth transition vector autoregressive model, structural smooth transition vector autoregressive model, regime-switching, SVAR, STVAR, maximum likelihood estimation.

1. Introduction

Linear vector autoregressive (VAR) models are a standard tools in time series econometrics. They can be employed to answer questions about the statistical relationships of different variables or to forecast future values of the process, for example. Structural VAR model allows to trace out the effects of economic shocks that have been identified by the researcher. With an appropriate choice of the autoregressive order p , a linear VAR is often able to filter out autocorrelation from the series very well. If the errors are assumed to follow an autoregressive conditional heteroskedasticity (ARCH) process, the model is also often able to adequately filter out conditional heteroskedasticity from the series.

In some cases, linear VAR models are not, however, capable to capture all the relevant characteristics of the series. This includes shifts in the mean or volatility, and changes in the autoregressive dynamics of the process. Such nonlinear features frequently occur in economic time series when the underlying data generating dynamics vary in time, for example, depending the specific state of the economy.

Various types of time series models capable of capturing this kind of regime-switching behav-

ior have been proposed, one of them is the smooth transition vector autoregressive (STVAR) models that allow to capture gradual shifts in the dynamics of the data. They consist of a finite number of regimes, each of which are linear vector autoregressions that are characterized by different autoregressive coefficients or error term covariance matrices. The package **sstvars** considers STVAR models in which, at each point of time, the observation is a weighted average of the conditional means of the regimes plus a random error whose covariance matrix is a weighted average of the covariance matrices of the regimes. The weights, in turn, are expressed in terms of time-varying transition weights that depend on the preceding observations. Different STVAR models can be created by specifying the transition weights or the error distribution in various ways. See [Hubrich and Teräsvirta \(2013\)](#) for a survey on STVAR literature, including formulations more general than our framework.

This manuscript describes the R package **sstvars** providing a set of easy-to-use tools for STVAR modeling, including unconstrained and constrained maximum likelihood (ML) estimation of the model parameters, residual based model diagnostics, estimation of linear impulse response functions, nonlinear generalized impulse response functions, and generalized forecast error variance decompositions. Further functionality includes hypothesis testing, plotting the profile log-likelihood functions about the estimate, simulation from STVAR processes, and forecasting, for example. Various transition weight functions are accommodated, including logistic weights [Anderson and Vahid \(1998\)](#), multinomial logit weights, exponential weights (e.g., [Hubrich and Teräsvirta 2013](#)), threshold weights ([Tsay 1998](#)), and transition weights that defined as weighted relative likelihoods of the regimes corresponding to the preceding p observations (FILL IN REFERENCE). Currently, the accommodated conditional distributions include Student's t distribution and Gaussian distribution, whereas the accommodated identification methods include recursive identification and identification by heteroskedasticity. More conditional distributions and identification methods are probably added in the future, however.

The estimation of the model parameters can, in some cases, be rather tricky. This is because the transition weights are determined endogenously, which induces a large number of modes to the log-likelihood function and large areas of the parameter space where the log-likelihood function is flat in multiple directions. Therefore, the model parameters are estimated by running multiple rounds of a two-phase estimation procedure in which a modified genetic algorithm is used to find starting values for a gradient based variable metric algorithm. Because of the multimodality of the log-likelihood function, some of the estimation rounds may end up in different local maximum points, thereby enabling the researcher to build models not only based on the global maximum point but also on the local ones. The estimated models can be conveniently examined with the **summary** and **plot** methods.

The remainder of this paper is organized as follows. Section 2 defines the implemented STVAR models and discusses some of their properties. For structural models, identification of the shocks is also covered. Section 4 discusses estimation of the model parameters. We also illustrates how the STVAR models can be estimated and examined with **sstvars** and how various parameter restrictions can be imposed in the estimation. In Section ??, we describe demonstrate the use residual based diagnostics to evaluate model adequacy. Section 6 discusses impulse response analysis, including generalized impulse response functions and generalized forecast error variance decompositions. Section ?? shows how the STVAR models can be built with given parameter values. In Section 8, we first show how to simulate observations from a STVAR process, and then we illustrate how to forecast future values of a STVAR pro-

cess with a simulation-based Monte Carlo method. Finally, Section 9 concludes and collects some useful functions in **sstvars** to a single table for convenience. Throughout this paper, we illustrate the use of **sstvars** with a quarterly series consisting of two U.S. variables: the percentage change of real GDP and the percentage change of GDP implicit price deflator, covering the period from 1959Q1 to 2019Q4.

2. Smooth Transition Vector Autoregressive Models

This section describes the STVAR models implemented in **sstvars**. First, we describe the general framework of STVAR models accommodates by **sstvars** and present a sufficient condition for their ergodic stationarity. Then, we present the implemented specifications transition weight functions and conditional distributions. Finally, we discuss structural STVAR models and implemented methods for identification of the shocks.

2.1. General framework for STVAR models

Let y_t , $t = 1, 2, \dots$, be the d -dimensional time series of interest and \mathcal{F}_{t-1} denote the σ -algebra generated by the random vectors $\{y_{t-j}, j > 0\}$. We consider STVAR models with M regimes and autoregressive order p assumed to satisfy

$$y_t = \sum_{m=1}^M \alpha_{m,t} \mu_{m,t} + B_t e_t, \quad e_t \sim IID(0, I_d) \quad (1)$$

$$\mu_{m,t} = \phi_{m,0} + \sum_{i=1}^p A_{m,i} y_{t-i}, \quad m = 1, \dots, M, \quad (2)$$

$$B_t B_t' = \sum_{m=1}^M \alpha_{m,t} \Omega_m, \quad (3)$$

where $\phi_{m,0} \in \mathbb{R}^d$ are intercept parameters, $\Omega_1, \dots, \Omega_M$ are the positive definite ($d \times d$) covariance matrices of the regimes, B_t is an invertible ($d \times d$) impact matrix that governs the contemporaneous relationships of the shocks and is a function of $\{y_{t-j}, j = 1, \dots, p\}$. For the reduced form model, any invertible matrix B_t that satisfies Equation (3) can be assumed without loss of generality, whereas structural models assume a specific structure on B_t . The structural errors e_t ($d \times 1$) are independent and identically distributed with mean zero and identity covariance matrix, and they are independent of \mathcal{F}_{t-1} . The transition weights $\alpha_{m,t}$ are assumed to be \mathcal{F}_{t-1} -measurable functions of $\{y_{t-j}, j = 1, \dots, p\}$ and to satisfy $\sum_{m=1}^M \alpha_{m,t} = 1$ at all t . They express the proportions of the regimes the process is on at each point of time. Through, a STVAR model with autoregressive order p and M regimes is referred to as a STVAR(p, M) model, whenever the order of the model needs to be emphasized.

Conditional on \mathcal{F}_{t-1} , the conditional mean of the above described process is $\mu_{y,t} \equiv E[y_t | \mathcal{F}_{t-1}] = \sum_{m=1}^M \alpha_{m,t} \mu_{m,t}$, and the conditional covariance matrix is $\Omega_{y,t} \equiv \text{Cov}(y_t | \mathcal{F}_{t-1}) = \sum_{m=1}^M \alpha_{m,t} \Omega_m$. In other words, the conditional mean is a weighted sum the regime-specific means $\mu_{m,t}$ with the weights given by the transition weights $\alpha_{m,t}$, whereas the conditional covariance matrix is a weighted sum of the regime-specific conditional covariance matrices Ω_m . Different STVAR models are obtained by specifying the transition weights or the error distribution (i.e., the conditional distribution) in various ways. See [Hubrich and Teräsvirta \(2013\)](#) for a survey on STVAR literature, including formulations more general than our framework.

The package **sstvars** accommodates models in which the transition weights are functions of $\{y_{t-j}, j = 1, \dots, p\}$, which is required for applicability of the stationarity condition presented below. Moreover, \mathcal{F}_{t-1} -measurability of the transition weights ensures that the true generalized impulse responses functions (Koop, Pesaran, and Potter 1996) can be easily estimated, as completely exogenous switching-variables are excluded from affecting the weights. We also assume that the transition weights are identical for all the individual equations in (1) and (3), which is also required for applicability of the stationarity condition. Consequently, at each t , the process can be described as a weighted sum of linear vector autoregressions.

Stationarity condition

It can be shown that a sufficient condition for the ergodic stationarity of the STVAR model (1)-(3) can be expressed in terms of the joint spectral radius (JSR) of certain matrices (Kheifets and Saikkonen 2020). The JSR of a finite set of square matrices \mathcal{A} is defined by

$$\rho(\mathcal{A}) = \limsup_{j \rightarrow \infty} \left(\sup_{A \in \mathcal{A}^j} \rho(A) \right)^{1/j}, \quad (4)$$

where $\mathcal{A}^j = \{A_1 A_2 \dots A_j : A_i \in \mathcal{A}\}$ and $\rho(A)$ is the spectral radius of the square matrix A .

Consider the companion form AR matrices of the regimes defined as

$$\mathbf{A}_m = \begin{bmatrix} A_{m,1} & A_{m,2} & \cdots & A_{m,p-1} & A_{m,p} \\ I_d & 0 & \cdots & 0 & 0 \\ 0 & I_d & & 0 & 0 \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & I_d & 0 \end{bmatrix}, \quad m = 1, \dots, M. \quad (5)$$

$(dp \times dp)$

Kheifets and Saikkonen (2020, Theorem 1) and FILL IN REFERENCE TO LANNE AND VIROLAINEN WORKING PAPER show that if the following condition holds, the STVAR process is ergodic stationary (both strictly and second-order).

Condition 1 $\rho(\{\mathbf{A}_1, \dots, \mathbf{A}_M\}) < 1$.

Condition 1 is, however, computationally demanding the check in practice with a reasonable accuracy (e.g., Chang and Blondel 2013), making it impractical to use in the estimation. Therefore, we consider a necessary condition for Condition 1 that is easier to check in practice, which is that the usual stability condition is satisfied for each of the regimes. Specifically, we assume the following condition, which is analogous to Corollary 1 of Kheifets and Saikkonen (2020), is necessary for Condition 1.

Condition 2 $\max\{\rho(\mathbf{A}_1), \dots, \rho(\mathbf{A}_M)\} < 1$,

where $\rho(\mathbf{A}_m)$ is the spectral radius of \mathbf{A}_m , $m = 1, \dots, M$.

Note that validity Condition 2 does not imply the validity of Condition 1 that guarantees stationarity of the model. However, in practice models that satisfy Condition 2 and are not very close to breaking this condition should most often satisfy Condition 1. For checking the validity of Condition 1, (Virolainen 2024) implements the formula of Parrilo and Jadbabaie

(2008) for the upper bound proposed by Blondel and Nesterov (2005) (the function `bound_JSR`). But unfortunately the calculations required are computationally very demanding in practice when the model is not small and even a relatively tight upper bound is required to verify Condition 1. Therefore, in the cases where Condition 1 needs to be formally verified, we suggest using the MATLAB toolbox **JSR** by Jungers (2023), as automatically combines several methods for bound the JSR and finds an accurate upper bound with a more reasonable computational effort.

2.2. Specifications of the conditional distribution

Gaussian STVAR model

Assuming the structural errors e_t have standard normal distributions, the conditional distribution of y_t conditional on \mathcal{F}_{t-1} is Gaussian and characterized by the density function

$$f(y_t|\mathcal{F}_{t-1}) = n_d(y_t; \mu_t, \Omega_t) = (2\pi)^{-d/2} \det(\Omega_t)^{-1/2} \exp \left\{ -\frac{1}{2}(y_t - \mu_t)' \Omega_t^{-1} (y_t - \mu_t) \right\}. \quad (6)$$

That is, the conditional distribution is simply the d -dimensional Gaussian distribution with mean μ_t and covariance matrix Ω_t . The Gaussian distribution simple and can be used with all of our transition weight functions, but in some cases it is useful to employ the more heavy tailed Student's t distribution instead.

Student's t STVAR model

To accommodate more heavy tailed data, instead of using Gaussian errors one may consider Student's t errors and assume the shocks e_t are Student's t distributed with the mean zero, identity covariance matrix, and $\nu > 2$ degrees of freedom (where the assumption $\nu > 2$ is made to ensure the existence of second moments). The Student's t STVAR model has the conditional distribution, conditional \mathcal{F}_{t-1} , characterized by the density function

$$f(y_t|\mathcal{F}_{t-1}) = t_d(y_t; \mu_t, \Omega_t, \nu) = C_d(\nu) \det(\Omega_t)^{-1/2} \left(1 + \frac{(y_t - \mu_t)' \Omega_t^{-1} (y_t - \mu_t)}{\nu - 2} \right)^{-(d+\nu)/2}, \quad (7)$$

where

$$C_d(\nu) = \frac{\Gamma\left(\frac{d+\nu}{2}\right)}{\sqrt{\pi^d (\nu - 2)^d} \Gamma\left(\frac{\nu}{2}\right)}, \quad (8)$$

and $\Gamma(\cdot)$ is the gamma function. The conditional distribution is, hence, the d -dimensional Student's t distribution with mean μ_t , covariance matrix Ω_t , and ν degrees of freedom. Note that the parametrization differs from the conventional one, as the distribution is parametrized with a covariance matrix instead a scale matrix (see, e.g., Meitz, Preve, and Saikkonen 2023, Appendix A for details about the parametrization).

The Student's t errors are more flexible than the Gaussian ones, but they are not accommodated by the transition weights defined as weighted ratios of the regime's stationary densities (see Section 2.3.1). This is because they require the knowledge of the stationary distributions of the regimes corresponding to p consecutive observations, and the stationary distribution is not known for the Student's t regimes (Virolainen 2021a, see, e.g.,). Moreover, Student's t

STVAR models are more difficult to estimate in practice than the Gaussian ones, and estimation of the STVAR models can be demanding.

2.3. Specifications of the transition weights

Various specifications of the transition weights $\alpha_{m,t}$ can be considered to obtain smooth transition VARs with different properties. We assume that the transition weights are functions of $\{y_{t-j}, j = 1, \dots, p\}$, because it is required for applicability of the stationarity condition discussed in Section 2.1.1. Moreover, \mathcal{F}_{t-1} -measurability of the transition weights ensures that the true generalized impulse responses functions can be easily estimated, as completely exogenous switching-variables are excluded from affecting the weights.¹ We also assume that the transition weights are identical for all the individual equations in (1) and (3), which is required for applicability of the stationarity condition. Consequently, at each t , the process can be described as a weighted sum of linear VARs.

Weighted relative likelihoods

If the conditional distribution is specified to be Gaussian, weighted relative likelihoods of the regimes can be used as transition weights (FILL IN REFERENCE TO LANNE AND VIROLAINEN). In this specification, the transition weights depend on the full distribution of the preceding p observations, they are defined identically to the mixing weights in the Gaussian mixture vector autoregressive (GMVAR) model of Kalliovirta, Meitz, and Saikkonen (2016). Denoting $\mathbf{y}_{t-1} = (y_{t-1}, \dots, y_{t-p})$, the transition weights are defined as

$$\alpha_{m,t} = \frac{\alpha_m n_{dp}(\mathbf{y}_{t-1}; \mathbf{1}_p \otimes \mu_m, \Sigma_{m,p})}{\sum_{n=1}^M \alpha_n n_{dp}(\mathbf{y}_{t-1}; \mathbf{1}_p \otimes \mu_n, \Sigma_{n,p})}, \quad m = 1, \dots, M, \quad (9)$$

where $\alpha_1, \dots, \alpha_M$ are transition weight parameters that satisfy $\sum_{m=1}^M \alpha_m = 1$ and $n_{dp}(\cdot; \mathbf{1}_p \otimes \mu_m, \Sigma_{m,p})$ is the density function of the dp -dimensional Gaussian distribution with mean $\mathbf{1}_p \otimes \mu_m$ and covariance matrix $\Sigma_{m,p}$. The symbol $\mathbf{1}_p$ denotes a p -dimensional vector of ones, \otimes is Kronecker product, $\mu_m = (I_d - \sum_{i=1}^p A_{m,i})^{-1} \phi_{m,0}$, and the covariance matrix $\Sigma_{m,p}$ is given in Lütkepohl (2005, Equation (2.1.39)), but using the parameters of the m th regime. That is, $n_{dp}(\cdot; \mathbf{1}_p \otimes \mu_m, \Sigma_{m,p})$ corresponds to the density function of the stationary distribution of the m th regime.

The transition weights are thus weighted ratios of the stationary densities of the regimes corresponding to the preceding p observations. This specification is appealing, as it states that the greater the weighted relative likelihood of a regime is, the greater the weight of this regime is. The regimes are, hence, formed based on the statistical properties of the data and are not affected by the choice of the switching variables similarly to the logistic weights.

In the GMVAR model of Kalliovirta *et al.* (2016), the definition of the mixing weights also leads to attractive theoretical properties such as the knowledge of the stationary distribution of $p + 1$ consecutive observations. But this is not the case in our STVAR model, as the structure of the model is different. The STVAR model, in turn, has the advantage that it allows for smooth transitions between the regimes, and it is potentially easier to estimate in practice, as the conditional distribution is Gaussian as opposed to the mixture Gaussian

¹Weaker forms of exogeneity of specific variables can, however, be imposed by constraining the AR matrices $A_{m,i}$, $m = 1, \dots, M$, $i = 1, \dots, p$, or the impact matrix B_t accordingly (see Section 4.6)

distribution of the GMVAR model. The STVAR model also has the advantage in structural analysis that after the shocks have been identified, they can be recovered from the data. In the GMVAR model, the shocks cannot be generally recovered because the regime that generated each observation is not known. The GMVAR model has been implemented to the R package **gmvar** Virolainen (2018a), which works quite similarly to **sstvars**.

Logistic transition weights

A common specification assumes logistic transition weights (e.g., Anderson and Vahid 1998) that vary according to the level of the switching variable, which we assume to be a lagged endogenous variable. Here we assume that model has only two regimes ($M = 2$), and in the next section, we show how the logistic weights generalize to multinomial logit weights that accommodate more regimes.

The logistic transition weights are defined as

$$\alpha_{1,t} = 1 - \alpha_{2,t}, \quad (10)$$

$$\alpha_{2,t} = [1 + \exp\{-\gamma(y_{it-j} - c)\}]^{-1}, \quad (11)$$

where y_{it-j} is the j th lagged observation ($j \in \{1, \dots, p\}$) of the i th variable ($i \in \{1, \dots, d\}$), $c \in \mathbb{R}$ is a location parameter, and $\gamma > 0$ is a scale parameter. The location parameter c determines the mid point of the transition function, i.e., the value of the (lagged) switching variable when the weights are equal. The scale parameter γ , in turn, determines the smoothness of the transitions (smaller γ implies smoother transitions), and it is assumed strictly positive so that $\alpha_{2,t}$ is increasing in y_{it-j} .

Compared to weighted relative likelihoods, an advantage of the logistic weights is that it allows to specify switching variables in a way that leads to the regimes the econometrician is interested in in a specific application. For instance, if one is interested in how the effects of the shocks vary along with business cycle fluctuations, y_{it-j} may be set as a lagged output gap variable. STVAR models with logistics weights are also easier to estimate than those with the transition weights determined by weighted relative likelihoods of the regimes. A disadvantage is that the empirical results depend highly on the choice of the switching variable, and only the level of the switching variable affects the transition weights.

Multinomial logit transition weights

The logistic transition weights can be generalized to multinomial logit weights that accommodate more than two regimes as well as multiple lags of multiple switching variables as regressors in the logit sub model. The generality, however, comes at the cost of significantly more difficult estimation in the practice and loss of the intuitive interpretations of the parameters of the transition function. With $M \geq 2$ regimes, we specify the multinomial logit weights as

$$\alpha_{m,t} = \frac{\exp\{\gamma'_m z_{t-1}\}}{\sum_{n=1}^M \exp\{\gamma'_n z_{t-1}\}}, \quad m = 1, \dots, M, \quad (12)$$

where z_{t-1} is an $(k \times 1)$ \mathcal{F}_{t-1} -measurable vector containing the (lagged) switching variables and a constant term, γ_m , $m = 1, \dots, M - 1$, are $(k \times 1)$ coefficient vectors, and the last one is normalized as $\gamma_M = 0$ ($k \times 1$) to facilitate identification.² Denote the set of switching

²Burgard, Neuenkirch, and Nöckel (2019) specify the mixing weights of their mixture VAR in a similar fashion, but unlike us, they allow for exogenous switching variables.

variables as $I \subset \{1, \dots, d\}$ (with the indices in I corresponding to the ordering of the variables in y_t) and assume that $\tilde{p} \in \{1, \dots, p\}$ lags are included in the transition weights. We assume

$$z_{t-1} = (1, \tilde{z}_{\min\{I\}}, \dots, \tilde{z}_{\max\{I\}}), \quad \tilde{z}_j = (y_{it-1}, \dots, y_{it-\tilde{p}}), \quad i \in I. \quad (13)$$

So $k = 1 + |I|\tilde{p}$ where $|I|$ is the cardinality of the set I (i.e., the number of elements in I). For instance, if the switching variables are the first two variables in y_t , $I = \{1, 2\}$ and only the first lag is included, $\tilde{p} = 1$, we have $z_{t-1} = (1, y_{1t-1}, y_{2t-1})$.

The specification implies

$$\log \frac{\alpha_{m,t}}{\alpha_{M,t}} = \gamma'_m z_{t-1}, \quad m = 1, \dots, M-1. \quad (14)$$

Our specification assumes that all the lags up to the lag \tilde{p} are included in the transition weights. The inclusion of only some specific lags in the transition weights is, however, accommodated by imposing constraints on the parameters $\alpha \equiv (\gamma_1, \dots, \gamma_{M-1})$ $((M-1)k \times 1)$. Specifically, **sstvars** assumes constraints of the form

$$\alpha = R\xi + r, \quad (15)$$

where R is a known $((M-1)k \times l)$ constraint matrix, r is a known $((M-1)k \times 1)$ constant, and ξ is an unknown $(l \times 1)$ parameter. For instance, by assuming that R is a matrix of zeros, the weight parameter α can be constrained to a known constant.

The logistic weights discussed in the previous section, with y_{it-j} as the switching variable for some lag $j \in \{1, \dots, p\}$ and $i \in \{1, \dots, d\}$, are obtained as a special case as follows. Assume $M = 2$, $\tilde{p} = j$, and $I = \{i\}$, so that $z_{t-1} = (1, y_{it-1}, \dots, y_{it-j})$. Then, impose the constraints $r = 0$ and

$$R = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 1 \end{bmatrix} \quad (j+1 \times 2), \quad (16)$$

so $\xi = (\gamma_{1,1}, \gamma_{j+1,1})$, where $\gamma_{l,m}$ is the l th element of γ_m .

A direct calculation shows that the "scale parameter" is $-\gamma_{j+1,1}$ and the "location parameter" is $\frac{\gamma_{1,1}}{-\gamma_{j+1,1}}$. The linear constraints (15) do not, however, enable to constrain the location parameter $\frac{\gamma_{1,1}}{-\gamma_{j+1,1}}$ to a specific value while leaving the scale parameter $-\gamma_{(j+1),1}$ unconstrained (or vice versa), or to constrain the scale parameter strictly positive. Therefore, it is more convenient to use the logistic weights and parametrization discussed in Section 2.3.2 when only two regimes and one lag of one switching variable are used.

Exponential transition weights

Exponential transition weights (see, e.g., [Teräsvirta 1994](#)) vary according to the level of the switching variable, which we assume to be a lagged endogenous variable. Similarly to the logistic transition weights discussed in Section 2.3.2, the exponential weights depend on a location parameter c and a scale parameter γ that determine the mid point of the transition curve and smoothness of the transitions, respectively. But instead of logistic transition function, we consider an exponential transition function.

Specifically, we assume $M = 2$ and define the exponential transition weights as

$$\alpha_{1,t} = 1 - \alpha_{2,t}, \quad (17)$$

$$\alpha_{2,t} = 1 - \exp\{-\gamma(y_{it-j} - c)^2\} \quad (18)$$

where y_{it-j} is the j th lagged observation ($j \in \{1, \dots, p\}$) of the i th variable ($i \in \{1, \dots, d\}$), $c \in \mathbb{R}$ is a location parameter, and $\gamma > 0$ is a scale parameter. The location parameter c determines the value of the (lagged) switching variable when the process is completely in first regime, i.e., $\alpha_{1,t} = 1$ and $\alpha_{2,t} = 0$. The closer y_{it-j} is to c , the greater the weight of the first regime is. Conversely, when the deviation of y_{it-j} from c increases, the weight of the second regime increases (and the weight of the first regime decreases). The scale parameter γ , in turn, determines the smoothness of the transitions (smaller γ implies smoother transitions), and it is assumed strictly positive so that $\alpha_{2,t} \in [0, 1]$ for all y_{it-j} .

Threshold transition weights

Threshold transition weights assume discrete regime switches such that the regime switches when the level of the switching variable exceeds or falls below a threshold value. This type model nonlinear VARs are often referred to as Threshold VAR (TVAR) models (Tsay 1998) or so-called self-exciting TVAR models (due to the endogenous switching-variable). We interpret the TVAR model as a special case of the STVAR models, despite of the regime switches being discrete rather than smooth.

For a model with $M > 1$ regimes, consider the $M - 1$ threshold values $r_1, \dots, r_{M-1} \in \mathbb{R}$ such that $r_1 < \dots < r_{M-1}$, and suppose the switching variable is $i \in \{1, \dots, d\}$ with the lag $j \in \{1, \dots, p\}$. The transition function is defined as

$$\alpha_{m,t} = \begin{cases} 1 & \text{if } r_{m-1} < y_{it-j} \leq r_m, \\ 0 & \text{otherwise,} \end{cases} \quad (19)$$

where $r_0 \equiv -\infty$, $r_M \equiv \infty$, and $m = 1, \dots, M$. In other words, at each t , the model defined in Equations (1)-(3) and (19) reduces to a linear VAR corresponding to one the regimes that is determined according to the level of the switching variable y_{it-j} .

Compared to smoothly varying transition weights, threshold transition weights have the advantage that the resulting model is easier to estimate in practice and the regimes have clear interpretations. An obvious disadvantage is the inability to capture smooth transition between the regimes or more complex regime-switching dynamics that depend on other factors than just on the level of the switching variable.

3. Structural STVAR models

The STVAR models specified in (1)-(3) directly incorporates the orthogonal, identically and independently distributed structural shocks e_t . The shocks are identified by finding an impact matrix B_t that satisfies Equation (3) for all t and recovers the shocks of interest. There are in general multiple solutions to B_t , since there are d^2 variables but only $d(d+1)/2$ unique equations in (3). Therefore, further restrictions are required for identification of the shocks. Currently, `sstvars` supports two types of identification methods: recursive identification by the

lower-triangular Cholesky decomposition and identification by conditional heteroskedasticity. Other identification methods will be likely added the package in the future.

3.1. Recursive identification

A conventional way of identifying the shocks is to impose restrictions on the impact responses of variables. A commonly applied identification is to assume a recursive lower-triangular structure on B_t , implying that B_t is obtained as the Cholesky decomposition of the conditional covariance matrix $\Omega_{y,t}$. The recursive identification is straightforward to apply and it allows many of the impact responses to vary in time but constraints many of them to zero. This is particularly disadvantageous if the shock of interest is ordered last or almost last (which is typically the case in small-scale monetary policy shock applications), as then the impact responses of many of the variables to the shock of interest are zero, and therefore, time-invariant.

With our specification of the STVAR model, recursive identification does not, however, allow to impose over-identifying restrictions on the impact matrix B_t . This is because there does not exist a direct parametrization of a lower-triangular B_t such that the conditional covariance matrix $\Omega_{y,t}$ is a weighted sum of the regime-specific covariance matrices with time-varying weights.

3.2. Identification via heteroskedasticity

An alternative identification method proposed by [Lütkepohl and Netšunajev \(2017\)](#) for structural VARs with smooth transitions in variances (see also the seminal paper by [Rigobon 2003](#)) identifies the shocks by simultaneously diagonalizing the covariance matrices $\Omega_1, \dots, \Omega_M$. This restricts the relative impact responses of the variables to be constant over time (for each shock) but does not necessarily require any zero restrictions. Since [Lütkepohl and Netšunajev \(2017\)](#) assume only two regimes and do not normalize the conditional covariance matrix of the structural error to a constant, their specification does not directly apply to our model. Therefore, we adopt the more suitable specification of [Virolainen \(2021b\)](#), and decompose the covariance matrices as

$$\Omega_m = W \Lambda_m W', \quad m = 1, \dots, M, \quad (20)$$

where the diagonal of $\Lambda_m = \text{diag}(\lambda_{m1}, \dots, \lambda_{md})$, $\lambda_{mi} > 0$ ($i = 1, \dots, d$), contains the eigenvalues of the matrix $\Omega_m \Omega_1^{-1}$ and the columns of the nonsingular W are the related eigenvectors (that are the same for all m by construction). When $M = 2$, the decomposition (20) always exists ([Muirhead 1982](#), Theorem A9.9), but for $M > 2$ its existence requires that the matrices $\Omega_m \Omega_1^{-1}$ share the common eigenvectors in W . This is, however, testable.

The impact matrix is then obtained as

$$B_t = W \left(\sum_{m=1}^M \alpha_{m,t} \Lambda_m \right)^{1/2}, \quad (21)$$

where $\Lambda_1 = I_d$. The shocks are identified up to ordering and sign if none of the pairs of λ_{mi} , $i = 1, \dots, d$, is identical for all $m = 2, \dots, M$. Assuming that this condition is satisfied, the shocks can be labelled according to the unrestricted impact responses on B_t , and if necessary, further economically motivated restrictions can be imposed. The additional economic restrictions on

the impact matrix testable, as they are overidentifying. See [Virolainen \(2021b\)](#) for a more detailed discussion on the identification and labeling of the shocks.

Shocks identified by heteroskedasticity impose constant relative impact responses for the variables, making them unsuitable for some applications concerned with time-varying impulse response functions. Similarly to the recursive identification, this method is straightforward to apply, but unlike the recursive identification, it does not necessarily require any zero constraints on the impact responses. Compared to the recursive identification, identification by heteroskedasticity is therefore particularly advantageous when the recursive identification would imply that the shock of interest is order last. In this case, the assumption of time-invariant relative impact responses is less restrictive than the zero restrictions of the recursive identification. In contrast, the recursive identification is particularly appealing when the shock of interest is ordered first (or almost first), as then the impact responses to the shock of interest can vary freely in time.

[Bacchiocchi and Fanelli \(2015\)](#), [Bacchiocchi, Castelnuovo, and Fanelli \(2016\)](#), and [Angelini, Bacchiocchi, Caggiano, and Fanelli \(2019\)](#) propose alternative way of identifying the shocks by combining heteroskedasticity with zero restrictions on the impact matrices. Their method allows some of the impact responses vary in time also relative to the other variables, but the required untestable zero constraints may be challenging to justify economically. But since their identification method is not directly applicable to the STVAR models specified in Section 2, we do not discuss it further.

4. Estimation

The parameters of the reduced form STVAR model are collected to the vector

$$\boldsymbol{\theta} = (\phi_{1,0}, \dots, \phi_{m,0}, \varphi_1, \dots, \varphi_M, \sigma, \alpha, \nu), \quad (22)$$

where $\varphi_m = (\text{vec}(A_{m,1}), \dots, \text{vec}(A_{m,p}))$, $m = 1, \dots, M$, $\sigma = (\text{vech}(\Omega_1), \dots, \text{vech}(\Omega_M))$, α contains the transition weight parameters, ν is the degrees of freedom parameter (which is dropped for Gaussian models), vec is a vectorization operator that stacks the columns of a matrix on top of each other, and vech stacks the columns of a matrix from the main diagonal downwards (including the main diagonal). Structural models identified by heteroskedasticity assume $\sigma = (\text{vec}(W), \lambda_m, \dots, \lambda_M)$, $\lambda_m = (\lambda_{m1}, \dots, \lambda_{md})$, $m = 2, \dots, M$, whereas recursively structural models use the same parameter vector as reduced form models.

If relative stationary densities are used as transition weights, $\alpha = (\alpha_1, \dots, \alpha_{M-1})$ (α_M is not included because it is obtained from the constraint $\sum_{m=1}^M \alpha_m = 1$), where we assume, for identification, that $\alpha_1, \dots, \alpha_{M-1}$ in a decreasing order. For exponential and logistic transition weights, $\alpha = (c, \gamma)$, for multinomial logit transition weights $\alpha = (\gamma_1, \dots, \gamma_{M-1})$ (γ_M is not included because $\gamma_M = 0$ is assumed for identification), and with threshold transition weights, $\alpha = (r_1, \dots, r_{M-1})$.

4.1. Log-likelihood function

`sstvars` employs the method of maximum likelihood (ML) for estimating the parameters of the STVAR models. Indexing the observed data as $y_{-p+1}, \dots, y_0, y_1, \dots, y_T$, the conditional

log-likelihood function conditional on the initial values $\mathbf{y}_0 = (y_{-p+1}, \dots, y_0)$ is given as

$$L_t(\boldsymbol{\theta}) = \sum_{t=1}^T l_t(\boldsymbol{\theta}) = \sum_{t=1}^T \log d_d(y_t; \mu_{y,t}, \Omega_{y,t}, \nu). \quad (23)$$

where $d_d(y_t; \mu_{y,t}, \Omega_{y,t}, \nu)$ is the d -dimensional conditional density of the process, conditional on \mathcal{F}_{t-1} , at time t , given in Section ?? . When the conditional distribution is Gaussian, the degrees of freedom parameter ν is dropped from the right side of (23). That is, the ML estimator of $\boldsymbol{\theta}$ maximizes the log-likelihood function $L_t(\boldsymbol{\theta})$ over the parameter space.

We summarize the constraints imposed on the parameter space in the following assumption.

Assumption 1 *The true parameter value $\boldsymbol{\theta}_0$ is an interior point of $\boldsymbol{\Theta}$, which is a compact subset of $\{\boldsymbol{\theta} = (\phi_{1,0}, \dots, \phi_{m,0}, \varphi_1, \dots, \varphi_M, \sigma, \alpha, \nu) \in \mathbb{R}^{M(d+d^2p+d(d+1)/2)} \times S \times (2, \infty) : \Omega_m \text{ is positive definite for all } m = 1, \dots, M, \text{ and Condition 1 holds.}\}$.*

Above, $S = (0, 1)^{M-1}$ for relative densities transition weights, $S = \mathbb{R} \times (0, \infty)$ for logistic and exponential weights, $S = \mathbb{R}^{(M-1)k}$ for multinomial logistic weights, and $S = \mathbb{R}^{M-1}$ for threshold weights. As noted before, Condition 1 is not necessary, but it ensures stationarity and ergodicity of the process. In estimation, we impose the more easily verified Condition 2, however, and the sufficient condition can be check after the estimation, if necessary. As noted in Section 2.1.1, the sufficient condition is in practice likely satisfied if the necessary condition is satisfied and not very close to being violated. Given that under Condition ?? the process is ergodic stationary, there is not particular reason to believe that the standard asymptotic results of consistency and limiting Gaussian distribution would not apply to the ML estimator.

4.2. Two-phase estimation procedure

Finding the ML estimate amounts maximizing the log-likelihood function (23) over a high dimensional parameter space satisfying the constraints summarized in Assumption 1. Due to the complexity of the log-likelihood function, numerical optimization methods are required. The maximization problem can be challenging in practice due to the dependence of the transition weights on the preceding observations, which induces a large number of modes to the surface of the log-likelihood function, and large areas to the parameter space, where it is flat in multiple directions.

Therefore, we follow Meitz *et al.* (2023) and Virolainen (2022b) and employ a two-phase estimation procedure that is run for a large number of times. In the first phase, a genetic algorithm is used to find parameter values (hopefully) near local maximums. Since genetic algorithms tend to converge slowly near local solutions, a gradient based variable algorithm (Nash 1990, algorithm 21, implemented by R Core Team 2022) is ran for each of the starting values, resulting in a number of alternative local solutions. Some of the estimation rounds may end up in saddle points or near-the-boundary points that are not local solutions, and some of the local solutions may be inappropriate for econometric inference (for instance, there might be only a few observations from some of the regimes). After the estimation rounds have been ran, the researcher can choose the local solution that maximizes the log-likelihood among the appropriate local solutions. Inappropriate solutions are automatically filtered by `sstvars`, but this functionality can also be turned off and the researchers can use estimates based any

estimation round. The R package **sstvars** employs a modified genetic algorithm that works similarly to the one described in the R packages **uGMAR** (Virolainen 2018b) and **gmvarKit** (Virolainen 2018a) (the genetic algorithm and iemployed in former is briefly described in Virolainen 2022b). See Virolainen (2022a, Chapter 3) for a related discussion on complex numerical estimation problems using the two-phase procedure.

4.3. Examples of unconstrained estimation

In this section, we demonstrate how to estimate STVAR models with **sstvars**. In the examples, we only consider $p = 1$ models for simplicity and because then the code outputs fit in the margins better, estimation times are shorter, etc. This order may not be best in the modeling perspective, however.

In **sstvars**, the STVAR models are defined as class **stvar** S3 objects, which can be created with given parameter values using the constructor function **STVAR** (see Section ??) or by using the estimation function **fitSTVAR**, which estimates the parameters and then builds the (reduced form) model. Structural models are estimated based on a reduced form model with the function **fitSSTVAR**. For estimation, **fitSTVAR** needs to be supplied with a multivariate time series and the arguments specifying the model. The necessary arguments for specifying the model include the autoregressive order **p**, the number of regimes **M**, the transition weight function **weight_function**, with some weight functions the switching variable(s) **weightfun_pars**, and the conditional distribution **cond_dist**.

Additional arguments may be supplied to **fitSTVAR** in order to specify, most importantly, how many estimation rounds should be performed (**nrounds**) and how many central processing unit (CPU) cores should be used in the estimation (**ncores**). Some of the estimation rounds may end up in local-only maximum points or saddle points, but reliability of the estimation results can be improved by increasing the number of estimation rounds. A large number of estimation rounds may be required particularly when the number of regimes is large or there are many variables in the data, as the surface of the log-likelihood function becomes increasingly more challenging. It is also possible to adjust the settings of the genetic algorithm that is used to find the starting values. The available options are listed in the documentation of the function **GAFit** to which the arguments adjusting the settings will be passed.

In general, we recommend being conservative with choice of **M** due to the identification problems induced if the number of regimes is chosen too large. Also, estimation of models that contain more than two regimes can be extremely challenging. Another important thing to know about estimation is that the estimation algorithm performs very poorly if some of the AR coefficients very large, significantly larger than one. This means that you need scale each component time series so that they vary approximately in the same magnitude. For instance, typically in macroeconomic time series, log-differences should be multiplied by hundred. If the suitable scales are not obvious, you can try out different scales and estimate linear VARs with your favorite package to see whether the AR coefficients are in a reasonable range. When a suitable scale is found, proceed to the STVAR models.

We illustrate the use of **sstvars** with a quarterly series consisting of two U.S. variables: the percentage change of real GDP and the percentage change of GDP implicit price deflator, covering the period from 1959Q1 to 2019Q4. The following code fits a STVAR($p = 1, M = 2$)

model with Gaussian conditional distribution and relative stationary densities as transition weights by performing 16 estimation rounds with 8 CPU cores. We use only 16 estimation rounds in this simplistic example to shorten the estimation time, knowing beforehand that the given seeds produce the desired result (in this simplistic case, majority the estimation rounds end up in the MLE anyway, though). **In practice, hundreds or even thousands of estimation rounds is often required to obtain reliable results. The larger the dimension of the series is and the larger the order of the model is and the more there are regimes, the more estimation rounds is required.**

The argument `seeds` supplies the seeds that initialize the random number generator at the beginning of each call to the genetic algorithm, thereby yielding reproducible results.

```
R> library(gmvarkit)
R> data("gdpdef", package="gmvarkit")
R> fit12t <- fitGSMVAR(gdpdef, p=1, M=2, model="StMVAR", ncalls=16, ncores=8,
+   seeds=1:16)
```

Using 8 cores for 16 estimations rounds...

Optimizing with a genetic algorithm...

```
|+++++| 100% elapsed=26s
```

Results from the genetic algorithm:

The lowest loglik: -277.038

The mean loglik: -258.416

The largest loglik: -252.168

Optimizing with a variable metric algorithm...

```
|+++++| 100% elapsed=06s
```

Results from the variable metric algorithm:

The lowest loglik: -276.974

The mean loglik: -252.439

The largest loglik: -243.719

Calculating approximate standard errors...

Finished!

Warning messages:

1: In `warn_df(p = p, M = M, params = params, model = model)` :

The model contains overly large degrees of freedom parameters. Consider switching to the appropriate G-StMVAR model by setting the corresponding regimes to GMVAR type with the function `'stmvar_to_gstmvar'`.

2: In `warn_eigens(ret)` :

Regime 2 has near-singular error term covariance matrix! Consider building a model from the next-largest local maximum with the function `'alt_gsmvar'` by adjusting its argument `'which_largest'`.

The progression of the estimation process is reported with a progress bar giving an estimate of the remaining estimation time. Also statistics on the spread of the log-likelihoods are printed after each estimation phase. The progress bars are generated during parallel computing with the package **pbapply** (Solymos and Zawadzki 2020).

The function throws a warning (the second warning) because at least one the regimes contains a near-singular covariance matrix. This kind of unreasonable boundary points can often be

disregarded, and the model can be built based on a reasonable estimate found from a local maximum that is clearly in the interior of the parameter space. Models based on the next-best local maximum can be built with the function `alt_gsmvar` by adjusting its argument `which_largest`.

The following code builds a StMVAR model based on the second-largest local maximum found in the estimation:

```
R> fit12t_alt <- alt_gsmvar(fit12t, which_largest=2)
```

Warning message:

```
In warn_df(p = p, M = M, params = params, model = model) :
```

```
  The model contains overly large degrees of freedom parameters. Consider
  switching to the appropriate G-StMVAR model by setting the corresponding
  regimes to GMVAR type with the function 'stmvar_to_gstmvar'.
```

The estimates can be examined with the `print`.

```
R> print(fit12t_alt)
```

Reduced form StMVAR model:

```
p = 1, M = 2, d = 2, #parameters = 21, #observations = 244 x 2,
conditional log-likelihood, intercept parametrization, no AR parameter
constraints
```

Regime 1

Mixing weight: 0.83

Regime means: 0.78, 0.54

Df parameter: 7.57

```
      Y      phi0      A1      Omega      1/2
1 y1 = [ 0.55 ] + [ 0.33 -0.04 ] y1.1 + ( [ 0.42 0.00 ] ) eps1
2 y2   [ 0.12 ]   [ 0.05 0.71 ] y2.1 ( ARCH_mt [ 0.00 0.04 ] ) eps2
```

Regime 2

Mixing weight: 0.17

Regime means: 0.66, 1.67

Df parameter: 92497.86

```
      Y      phi0      A1      Omega      1/2
1 y1 = [ 1.60 ] + [ 0.13 -0.61 ] y1.1 + ( [ 1.21 -0.04 ] ) eps1
2 y2   [ 0.48 ]   [ -0.03 0.72 ] y2.1 ( ARCH_mt [ -0.04 0.14 ] ) eps2
```

The parameter estimates are reported for each mixture component separately so that the estimates can be easily interpreted. Each regime's autoregressive formula is presented in the form

$$y_t = \varphi_{m,0} + A_{m,1}y_{t-1} + \dots + A_{m,p}y_{t-p} + \Omega_{m,t}^{1/2}\varepsilon_{m,t}. \quad (24)$$

If $\Omega_{m,t}^{1/2}$ is time varying, it printed in the form $\Omega_{m,t}^{1/2} = (\text{ARCH_mt}\Omega_m)^{1/2}$ where ARCH_mt is the $\omega_{m,t}$ defined in (??). No numerical value is given to the ARCH scalar, as it is time-varying. The other statistics are listed above the formula, including the mixing weight parameter α_m , the unconditional mean μ_m , and the degrees freedom parameter ν_m .

The above printout shows that the second regime's degrees of freedom parameter estimate is very large, which might induce numerical problems. However, since a StMVAR model with some degrees of freedom parameters tending to infinity coincides with the G-StMVAR model with the corresponding regimes switched to GMVAR type, one may avoid the problems by switching to the appropriate G-StMVAR model (see Virolainen 2021b). Switching to the appropriate G-StMVAR model is recommended also because it removes the redundant degrees of freedom parameters from the model, thereby reducing its complexity. The function `stmvar_to_gstmvar` does this switch automatically by first removing the large degrees of freedom parameters and then estimating the G-StMVAR model with a variable metric algorithm (Nash 1990, algorithm 21) using the induced parameter vector as the initial value.

To exemplify, the following code switches all the regimes of the StMVAR model `fit12t_alt` with a degrees of freedom parameter estimate larger than 100 to GMVAR type, and then estimates the corresponding G-StMVAR model.

```
R> fit12gs <- stmvar_to_gstmvar(fit12t_alt, maxdf=100)
```

We use the `summary` method to obtain a more detailed printout of the estimated the G-StMVAR model:

```
R> summary(fit12gs)
```

Reduced form G-StMVAR model:

```
p = 1, M1 = 1, M2 = 1, d = 2, #parameters = 20, #observations = 244 x 2,
conditional log-likelihood, intercept parametrization, no AR parameter
constraints
```

```
log-likelihood: -247.50, AIC: 534.99, HQIC: 563.13, BIC: 604.85
```

Regime 1 (GMVAR type)

```
Moduli of 'bold A' eigenvalues: 0.75, 0.10
```

```
Cov. matrix 'Omega' eigenvalues: 1.21, 0.14
```

```
Mixing weight: 0.17
```

```
Regime means: 0.66, 1.67
```

```
      Y      phi0      A1      Omega      1/2
1 y1 = [ 1.60 ] + [ 0.13 -0.61 ] y1.1 + [ 1.21 -0.04 ]      eps1
2 y2   [ 0.48 ]   [ -0.03  0.72 ] y2.1   [ -0.04  0.14 ]      eps2
```

Error term correlation matrix:

```
      [,1] [,2]
[1,] 1.000 -0.087
[2,] -0.087 1.000
```

Regime 2 (StMVAR type)

Moduli of 'bold A' eigenvalues: 0.70, 0.34

Cov. matrix 'Omega' eigenvalues: 0.42, 0.04

Mixing weight: 0.83

Regime means: 0.78, 0.54

Df parameter: 7.57

	Y	phi0	A1		Omega	1/2
1	y1 = [0.55]	+	[0.33 -0.04]	y1.1 + ([0.42 0.00]) eps1
2	y2 [0.12]		[0.05 0.71]	y2.1 (ARCH_mt [0.00 0.04])		eps2

Error term correlation matrix:

	[,1]	[,2]
[1,]	1.000	0.014
[2,]	0.014	1.000

Print approximate standard errors with the function 'print_std_errors'.

In the G-StMVAR model, estimates for GMVAR type regimes are reported before StMVAR type regimes, in a decreasing order according to the mixing weight parameter estimates. As shown above, the model `fit12gs` incorporates one GMVAR type regime and one StMVAR type regime. Estimates of the unconditional mean, the first p autocovariances and autocorrelations (including the unconditional covariance matrix) can be obtained from the element `$uncond_moments` of the model object. The conditional moments calculated using the data are available for the process (`$total_cmeans` and `$total_ccovs`) as well as for the regimes separately (`$regime_cmeans` and `$regime_ccovs`). These conditional moments can be conveniently plotted along the series with the function `cond_moment_plot`.

Approximate standard errors can be printed with the function `print_std_errors`, which prints the standard errors in the same form as the print method prints the estimates. Note that the last mixing weight parameter estimate does not have an approximate standard error because it is not parametrized. Likewise, there is no standard error for the intercepts if mean parametrization is used (by setting `parametrization = "mean"` in `fitGSMVAR`) and vice versa. In order to obtain standard errors for the regimewise unconditional means or intercepts, one can easily swap between the mean and intercept parametrizations with the function `swap_parametrization`.

To exemplify, the following code prints approximate standard errors for the model `fit12gs`:

```
R> print_std_errors(fit12gs)
```

Reduced form model:

p = 1, M = 2, conditional log-likelihood, intercept parametrization,
no AR parameter constraints

APPROXIMATE STANDARD ERRORS

Regime 1 (GMVAR type)

Mixing weight: 0.124

	Y	phi0		A1		Omega	1/2	
1	Y1 =	[0.759]	+	[0.149 0.386]	Y1.1	+	[0.264 0.061]	eps1
2	Y2	[0.234]		[0.050 0.118]	Y2.1		[0.061 0.029]	eps2

Regime 2 (StMVAR type)

Df parameter: 2.740

	Y	phi0		A1		Omega	1/2	
1	Y1 =	[0.127]	+	[0.079 0.196]	Y1.1	+	([0.070 0.011]	eps1
2	Y2	[0.037]		[0.024 0.064]	Y2.1	(ARCH_mt	[0.011 0.008]	eps2

Missing values are reported when **gmvar** is not able to calculate the standard error. This typically happens either because there is an overly large degrees of freedom parameter estimate in the model or because the estimation algorithm did not stop a local maximum. In the former case, switch to the appropriate G-StMVAR with the function `stmvar_to_gstmvar`. In the latter case, make sure the estimate is not an unreasonable near-the-boundary point. If it is, it might be appropriate to consider the next-best local maximum with the function `alt_gsmvar`. If it is not a near-the-boundary point, try running more iterations of the variable metric algorithm with the function `iterate_more`. Section 4.4 discusses how to evaluate with **gmvar** whether the estimate is a local maximum (and how to improve the reliability that it is the global maximum).

Other statistics reported in the summary printout include the log-likelihood and values of the information criteria, moduli of the eigenvalues of the 'bold A' matrix (see (??)) and eigenvalues of the covariance matrix Ω_m . If some of the moduli are very close to one, the related estimates are near the boundary of the stationarity region. If some of the eigenvalues of Ω_m close to zero, the related estimates are near the boundary of positive-definiteness region. As mentioned already multiple times, this kind of near-the-boundary point might be unreasonable and maximize the log-likelihood function for a technical reason, so it might be more appropriate to consider the next-best local maximum with the function `alt_gsmvar`.

This is possible in **gmvar**, because the estimation function `fitGSMVAR` stores the estimates from all the estimation rounds so that a GSMVAR model can be built based on any one of them, most conveniently with the function `alt_gsmvar`. The desired estimation round can be specified either with the argument `which_round` or `which_largest`. The former specifies the round in the estimation order, whereas the latter specifies it in a decreasing order of the log-likelihoods.

It is also possible to automatically filter out inappropriate estimates by setting the argument `filter_estimates=TRUE` in `fitSGMVAR`. Then, the function will automatically filter out estimates that it deems "inappropriate". That is, estimates that are not likely solutions of interest. Specifically, solutions that incorporate a near-singular error term covariance matrix (any eigenvalue less than 0.002), mixing weights such that they are close to zero for almost all t for at least one regime, or mixing weight parameter estimate close to zero (or one). It also filters out estimates with any modulus "bold A" eigenvalues larger than 0.9985, as the solution is near the boundary of the stationarity region and likely not a local maximum. `fitSGMVAR` then returns the solution based on the largest log-likelihood that is not filtered out. Other

solutions can be studied by using the function `alt_gsmvar` as usual.

4.4. Further examination of the estimates

In addition to examining the summary printout, it is often useful to visualize the model by plotting the mixing weights together with the time series and the model's (marginal) stationary density together with a kernel density estimate of the time series. That is exactly what the plot method for GSMVAR models does. The following command creates the time series plot along with estimated mixing weights:

```
R> plot(fit12gs, type="series")
```

The resulting plot is presented in Figure ??.

And the following command creates the stationary density plot:

```
R> plot(fit12gs, type="density")
```

The resulting plot is presented in Figure ?. If the argument `type` is not specified, both of the figures will be plotted.

It is also sometimes interesting to examine the time series of (one-step) conditional means and variances of the process along with the time series the model was fitted to. This can be done conveniently with the function `cond_moment_plot`, where the argument `which_moment` should be specified with `"mean"` or `"variance"` accordingly. In addition to the conditional moment of the process, `cond_moment_plot` also displays the conditional means or variances of the regimes multiplied by the mixing weights. Note, however, that the conditional variance of the process is not generally the same as the weighted sum of regimewise conditional variances, as it includes a component that encapsulates heteroskedasticity caused by variation in the conditional mean.

The variable metric algorithm employed in the final estimation does not necessarily stop at a local maximum point. The algorithm might also stop at a saddle point or near a local maximum, when the algorithm is not able to increase the log-likelihood, or at any point, when the maximum number of iterations has been reached. In the latter case, the estimation function throws a warning, but saddle points and inaccurate estimates need to be detected by the researcher.

It is well known that in a local maximum point, the gradient of the log-likelihood function is zero, and the eigenvalues of the Hessian matrix are all negative. In a local minimum, the eigenvalues of the Hessian matrix are all positive, whereas in a saddle point, some of them are positive and some negative. Nearly numerically singular Hessian matrices occur when the surface of the log-likelihood function is very flat about the estimate in some directions. This particularly happens when the model contains overly large degrees of freedom parameter estimates or the mixing weights $\alpha_{m,t}$ are estimated close to zero for all $t = 1, \dots, T$ for some regime m .

`gmvarKit` provides several functions for evaluating whether the estimate is a local maximum point. The function `get_foc` returns the (numerically approximated) gradient of the log-likelihood function evaluated at the estimate, and the function `get_soc` returns eigenvalues of the (numerically approximated) Hessian matrix of the log-likelihood function evaluated at the

estimate. The numerical derivatives are calculated using a central difference approximation

$$\frac{\partial L(\boldsymbol{\theta})}{\partial \theta_i} \approx \frac{f(\boldsymbol{\theta} + \mathbf{h}^{(i)}) - f(\boldsymbol{\theta} - \mathbf{h}^{(i)})}{2h}, \quad h > 0, \quad (25)$$

where θ_i is the i th element of $\boldsymbol{\theta}$ and $\mathbf{h}^{(i)} = (0, \dots, 0, h, 0, \dots, 0)$ contains h as its i th element. By default, the difference $h = 6 \cdot 10^{-6}$ is used for all parameters except for overly large degrees of freedom parameters, whose partial derivatives are approximated using larger differences. The difference is increased for large degrees of freedom parameters, because the limited precision of the float point presentation induces artificially rugged surfaces to the their profile log-likelihood functions, and the increased differences diminish the related numerical error. On the other hand, as the surface of the profile log-likelihood function is very flat about a large degrees of freedom parameter estimate, large differences work well for the approximation.

For example, the following code calculates the first order condition for the G-StMVAR model `fit12gs`:

```
R> get_foc(fit12gs)
```

```
[1] 1.475392e-03 2.520058e-03 1.593868e-03 -4.325443e-03
[5] 3.426033e-03 1.272343e-03 -7.762371e-04 -4.553748e-03
[9] 3.181280e-03 3.919771e-03 -2.321276e-02 4.170381e-03
[13] -1.782989e-02 3.166647e-03 -3.229744e-03 -3.738130e-03
[17] -9.082465e-03 2.232360e-02 -7.895506e-03 9.746278e-06
```

and the following code calculates the second order condition:

```
R> get_soc(fit12gs)
```

```
[1] -1.329154e-01 -1.389172e+00 -1.273009e+01 -1.865806e+01
[5] -2.067262e+01 -5.037907e+01 -8.094178e+01 -1.070554e+02
[9] -1.715455e+02 -2.124879e+02 -2.769413e+02 -3.371079e+02
[13] -4.467047e+02 -1.104002e+03 -1.130339e+03 -1.261037e+03
[17] -1.820865e+03 -9.262083e+03 -1.302900e+04 -3.456585e+04
```

All eigenvalues of the Hessian matrix are negative, which points to a local maximum, but the gradient of the log-likelihood function seems to somewhat deviate from zero. The gradient might be inaccurate, because it is based on a numerical approximation. It is also possible that the estimate is inaccurate, because it is based on approximative numerical estimation, and the estimates are therefore not expected to be exactly accurate. Whether the estimate is a local maximum point with accuracy that is reasonable enough, can be evaluated by plotting the graphs of the profile log-likelihood functions about the estimate. In **gmvarKit**, this can be done conveniently with the function `profile_logliks`.

The exemplify, the following command plots the graphs of profile log-likelihood functions of the estimated G-StMVAR model `fit12gs`:

```
R> profile_logliks(fit12gs, scale=0.02, precision=200)
```

The resulting plot is presented in Figure ??.

The output shows that the estimate's accuracy is reasonable, as changing any individual parameter value marginally would not increase the log-likelihood much. The argument `scale` can be adjusted to shorten or lengthen the interval shown in the horizontal axis. If one zooms in enough by setting `scale` to a very small number, it can be seen that the estimate is not exactly at the local maximum, but it is so close that moving there would not increase the log-likelihood notably. The argument `precision` can be adjusted to increase the number of points the graph is based on. For faster plotting, it can be decreased, and for more precision, it can be increased. The argument `which_pars` is used to specify the parameters whose profile log-likelihood functions should be plotted. This argument is particularly useful when creating as many plots as there are parameters in the model to a single figure would cause the individual plots to be very small. In such a case, profile log-likelihood functions for subsets of the parameters can be plotted separately by specifying this argument accordingly.

We have discussed tools that can be utilized to evaluate whether the found estimate is a local maximum with a reasonable accuracy. It is, however, more difficult to establish that the estimate is the global maximum. With `gmvarkit`, the best way to increase the reliability that the found estimate is the global maximum, is to run more estimation rounds by adjusting the argument `ncalls` of the estimation function `fitGSMVAR`.

If the model is very large, a very large number of estimation rounds may be required to find the global maximum. If there are two regimes in the model, p is reasonable, and the dimension of the time series at most four, the required number of estimation rounds typically varies from several hundred to several thousand depending on the model and the data. In the simpler models, less estimation rounds are required. In the larger models, and in particular if $M > 2$ or $d > 4$, a significantly large number of estimation rounds may be required obtain the MLE. Another thing that makes the estimation more challenging, are exotic parameter constraints that do not reduce the dimension of the parameter much. Constraints that greatly reduce complexity of the parameter space (such as constraining the autoregressive matrices to be identical in all regimes³), on the other hand, make the estimation easier and reliable estimation of such models thereby require less estimation rounds.

4.5. Estimation of the structural GSMVAR model

As explained, `gmvarkit` currently supports two types of structural models: structural models identified recursively by the lower triangular Cholesky decomposition and structural models identified by conditional heteroskedasticity. Recursive identification is assumed for reduced form models, and thus do not require any further estimation. Generalized impulse response functions and generalized forecast error variance decompositions can be estimated for recursively identified models by using the reduced form models directly in the functions `GIRF` and `GFEVD`. The rest of this section discusses estimation of structural models identified by conditional heteroskedasticity as in Virolainen (2021b), which employ a parametrization different to the reduced form models.

The structural GSMVAR models are estimated similarly to the reduced form version, expect that the model is parametrized with W and λ_{mi} , $m = 2, \dots, M$, $i = 1, \dots, d$ instead of the covariance matrices Ω_m , $m = 1, \dots, M$. The estimation is can be done with the function

³Models constrained in this way can often be reliably estimated with a reasonable number of estimation rounds even when $M > 2$

`fitGSMVAR` but now the argument `structural_pars` needs to be supplied with a list providing the constraints on W (which equally imposes the constraints on the B-matrix), and optionally, linear constraints on the λ_{mi} parameters or constraints restricting λ_{mi} to fixed values.

The list `structural_pars` should contain at least the element `W` which is a $(dx d)$ matrix matrix with its entries imposing constraints on W : NA indicating that the element is unconstrained, a positive value indicating strict positive sign constraint, a negative value indicating strict negative sign constraint, and zero indicating that the element is constrained to zero. The elements named `C_lambda` and `fixed_lambdas` are optional (and alternative to each other).

If `C_lambda` is specified, it should be a $(d(M - 1) \times r)$ constraint matrix that satisfies $(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma$ where $\lambda_m = (\lambda_{m1}, \dots, \lambda_{md})$ and γ is the new $(rx1)$ parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of `C_lambda` must be either positive or zero. Ignore (or set to NULL) if the eigenvalues λ_{mi} should not be constrained. Note that other constraints than constraining some of the λ_{mi} to be identical are not recommended but if such constraints are imposed, the argument `lambda_scale` in the genetic algorithm (see `?GAfit`) should be adjusted accordingly. If some of the λ_{mi} are constrained to be identical, make sure the appropriate zero constraints placed in the W matrix, because otherwise the MLE does not identify and you probably won't obtain any useful estimates (see Virolainen 2021b, Proposition 2).

If `fixed_lambdas` is specified, it should be a $d(M - 1)$ length numeric vector $(\lambda_2, \dots, \lambda_M)$ specifying fixed values for the eigenvalue parameters λ_{mi} . They should be strictly larger than zero. Ignore (or set to NULL) if the eigenvalues λ_{mi} should not be constrained. Note that you cannot use both `C_lambda` and `fixed_lambdas` at the same time.

Reliable estimation of structural GSMVAR models typically requires much more estimation rounds than the estimation of the reduced form models. However, when $M = 2$, every reduced form model has an implied statistically identified structural model, which can be built without any additional estimation (this will be discussed next). We recommend considering this implied model first. Then, if overidentifying constraints are to be imposed on the B-matrix (or equally W), we recommend using the unrestricted estimate to create an initial guess for the constrained parameter vector and pass this to the genetic algorithm as an initial population. See the help page `?GAfit` for the arguments that can be passed by `fitGSMVAR` to the genetic algorithm. Create the initial guess for the parameter vector by using the form given in documentation of the argument `initpop`. If $M \neq 2$, the structural model needs to be estimated in the normal way with `fitGSMVAR`, however.

Building structural model based on a reduced form model

If the number of regimes is two ($M = 2$), a structural model can be built based on a reduced form model, because the matrix decomposition used in the simultaneous diagonalization of the error term covariance matrices always exists. This can be done with function `gsmvar_to_sgsmvar` which should be supplied with the reduced form model, and it then returns a corresponding structural model. After creating the structural model, the columns of W can be reordered with the function `reorder_W_columns` which also reorders all λ_{mi} accordingly (and hence the resulting model will coincide with the original reduced form model). Also, all signs any column of W can be swapped with the function `swap_W_signs`.

To exemplify, the following code creates statistically identified structural model based on

the reduced form model `fit12gs` and then prints the estimates.

```
R> fit12gss <- gsmvar_to_sgsmvar(fit12gs)
R> fit12gss
```

Structural G-StMVAR model:

```
p = 1, M1 = 1, M2 = 1, d = 2, #parameters = 20, #observations = 244 x 2,
conditional log-likelihood, intercept parametrization, no AR parameter
constraints
```

Regime 1 (GMVAR type)

Mixing weight: 0.17

Regime means: 0.66, 1.67

```
      Y      phi0      A1      Omega      1/2
1 y1 = [ 1.60 ] + [ 0.13 -0.61 ] y1.1 + [ 1.21 -0.04 ]      eps1
2 y2   [ 0.48 ]   [ -0.03  0.72 ] y2.1   [ -0.04  0.14 ]      eps2
```

Regime 2 (StMVAR type)

Mixing weight: 0.83

Regime means: 0.78, 0.54

Df parameter: 7.57

```
      Y      phi0      A1      Omega      1/2
1 y1 = [ 0.55 ] + [ 0.33 -0.04 ] y1.1 + (      [ 0.42 0.00 ] )      eps1
2 y2   [ 0.12 ]   [ 0.05  0.71 ] y2.1   ( ARCH_mt [ 0.00 0.04 ] )      eps2
```

Structural parameters:

```
      W      lamb2
1 [ 0.95 -0.55 ] [ 0.36 ]
2 [ 0.16  0.34 ] , [ 0.28 ]
```

The B-matrix (or equally W) is subject to 0 zero constraints and 2 sign constraints. The eigenvalues `lambda_{mi}` are not subject to linear constraints.

Estimates for the structural parameters, W and the eigenvalues, are printed last.

If there is only one mixture component, i.e., `M == 1`, `gsmvar_to_sgsmvar` returns a symmetric and pos. def. square root matrix of the error term covariance matrix by default. But one may also employ lower triangular Cholesky identification by setting `cholesky = TRUE` in the arguments.

Estimating overidentified structural GSMVAR models

Sometimes is appropriate to impose overidentifying constraints on W (or equally the B-matrix). With preliminary estimates from the just-identified model in the case $M > 1$ or any model in the case $M = 1$, it is convenient to use the function `estimate_sgsmvar`. It takes in a reduced form or structural GSMVAR model as a class 'gsmvar' object and new constraints

in the argument `new_W` as a matrix expressing the sign or zero constraints. Strictly positive or negative elements signify strict sign constraints, zeros zero constraints, and NA values that the element is unconstrained.

`estimate_sgsmvar` then creates preliminary estimate based on the supplied model and the constraints, and then runs the two-phase estimation with settings of the genetic algorithm such that the search is focused on the neighbourhood of the preliminary estimate. Thus, it will lead to the correct ML estimate only if the unconstrained estimate is close to the constrained one in the first place. It is therefore useful for imposing zero constraints for elements that are close to zero in the unrestricted estimate, for instance.

It is important to make sure that supplied model readily satisfies the sign constraints that are imposed. To achieve this, you can swap the signs in each column of the W matrix with the function `swap_W_signs`. If the sign constraints are not readily satisfied, the preliminary estimate switches the signs and will probably lead to incorrect estimate.

`estimate_sgsmvar` can also be used to estimate models that are not identified, i.e., one regime models. If it supplied with a reduced form model, it will first apply the function `gsmvar_to_sgsmvar`, then impose the constraints and finally estimate the model.

4.6. Constrained estimation

Linear constraints on the autoregressive parameters

Imposing linear constraints on the autoregressive parameters of GMVAR model is straightforward in **gmvar**kit. The constraints are expressed in a somewhat general form which allows to impose a wide class of constraints but one needs to take the time to construct the constraint matrix carefully for each particular case.

We consider constraints of form

$$(\phi_1, \dots, \phi_M) = \mathbf{C}\psi, \quad (26)$$

$$\phi_m = (\text{vec}(A_{m,1}), \dots, \text{vec}(A_{m,p})) \quad (pd^2x1), \quad m = 1, \dots, M, \quad (27)$$

\mathbf{C} is known (Mpd^2xq) constraint matrix (of full column rank) and ψ is unknown $(qx1)$ parameter vector.

The parameter vector for constrained model has the size $((M(d + d(d + 1)/2 + 1) + q - 1)x1)$ and the form

$$\boldsymbol{\theta} = (\phi_{1,0}, \dots, \phi_{M,0}, \boldsymbol{\psi}, \alpha_1, \dots, \alpha_{M-1}, \boldsymbol{\nu}), \quad (28)$$

where $\boldsymbol{\psi}$ is the $(qx1)$ parameter vector containing constrained autoregressive parameters. As in the case of regular models, instead of the intercept parametrization that takes use of intercept terms $\phi_{m,0}$, one may use the mean parametrization with regimewise means μ_m instead ($m = 1, \dots, M$).

Examples of linear constraints

Consider the following two common uses of linear constraints: restricting the autoregressive matrices to be the same for all regimes and constraining some AR parameters to zero. Of

course also some other constraints may be useful, but we chose to show illustrative examples of these two, as they are taken use of in [Kalliovirta *et al.* \(2016\)](#).

Restricting AR matrices to be the same for all regimes

To restrict the AR matrices to be the same for all regimes, we want ϕ_m to be the same for all $m = 1, \dots, M$. The parameter vector ψ ($qx1$) then corresponds to any $\phi_m = \phi$, and therefore $q = pd^2$. For the constraint matrix we choose

$$C = [I_{pd^2} : \dots : I_{pd^2}]' \quad (Mpd^2 \times pd^2), \quad (29)$$

that is, M pieces of $(pd^2 \times pd^2)$ diagonal matrices stacked on top of each other, because then

$$C\psi = (\psi, \dots, \psi) = (\phi, \dots, \phi). \quad (30)$$

For instance, if there are two regimes in the model, the appropriate constraint matrix then created with R as

```
R> p <- 1 # Any autoregressive order
R> d <- 2 # Whatever the dimension of the time series is
R> I_pd2 <- diag(p*d^2) # The appropriate diagonal matrix
R> (C1 <- rbind(I_pd2, I_pd2)) # Stack them on top of each other
```

```
      [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    1    0    0
[3,]    0    0    1    0
[4,]    0    0    0    1
[5,]    1    0    0    0
[6,]    0    1    0    0
[7,]    0    0    1    0
[8,]    0    0    0    1
```

The command `fitGSMVAR(gdpdef, p=1, M=2, model="GMVAR", constraints=C1)` would then estimate a GMVAR(1,2) model with the AR matrices constrained to be the same in both regimes. In practice, you might want to adjust the number of CPU cores used, the of estimation rounds, and set seeds. Notably, with the dimension of the time series being only two and $p = 1$ with two regimes, almost all of the estimation rounds end up in the MLE. Also, because model has the AR matrices constrained to be the same for all regimes, the estimation is much easier than with freely estimated models.

Restricting AR parameters to be the same for all regimes and constraining non-diagonal elements of coefficient matrices to be zero

The previous example shows how to restrict the AR parameters to be the same for all regimes, but say we also want to constrain the non-diagonal elements of coefficient matrices $A_{m,i}$ ($m = 1, \dots, M, i = 1, \dots, p$) to be zero. We have the constrained parameter ψ ($qx1$) representing the unconstrained parameters (ϕ_1, \dots, ϕ_M) , where by assumption $\phi_m = \phi =$

$(vec(A_1), \dots, vec(A_p))$ (pd^2x1) and the elements of $vec(A_i)$ ($i = 1, \dots, p$) corresponding to the diagonal are zero.

For illustrative purposes, let's consider a GMVAR model with autoregressive degree $p = 2$, number of mixture components $M = 2$ and number of time series in the system $d = 2$. Then we have

$$\phi = (A_1(1, 1), 0, 0, A_1(2, 2), A_2(1, 1), 0, 0, A_2(2, 2)) \quad (8x1) \quad \text{and} \quad (31)$$

$$\psi = (A_1(1, 1), A_1(2, 2), A_2(1, 1), A_2(2, 2)) \quad (4x1). \quad (32)$$

By a direct calculation, we can see that choosing the constraint matrix

$$C = \begin{bmatrix} \tilde{c} \\ \tilde{c} \end{bmatrix} \quad (Mpd^2x4), \quad (33)$$

where

$$\tilde{c} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (pd^2x4) \quad (34)$$

satisfies $C\psi = (\phi, \dots, \phi)$.

The above constraint matrix can be created with R as

```
R> c_tilde <- matrix(0, nrow=2*2^2, ncol=4)
R> c_tilde[c(1, 12, 21, 32)] <- 1
R> C2 <- rbind(c_tilde, c_tilde)
R> C2
```

```
      [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    0    0    0
[3,]    0    0    0    0
[4,]    0    1    0    0
[5,]    0    0    1    0
[6,]    0    0    0    0
[7,]    0    0    0    0
[8,]    0    0    0    1
[9,]    1    0    0    0
[10,]   0    0    0    0
[11,]   0    0    0    0
[12,]   0    1    0    0
[13,]   0    0    1    0
[14,]   0    0    0    0
[15,]   0    0    0    0
[16,]   0    0    0    1
```

The command `fitGSMVAR(gdpdef, p=2, M=2, model="GMVAR", constraints=C2)` would then estimate a GMVAR(2,2) model with the AR matrices constrained to be the same in both regimes and the off-diagonal elements constrained to zero (again, you may want to adjust the arguments `ncalls`, `ncored`, and `seeds`).

Constraining the unconditional means of some regimes to be the same

In addition to constraining the autoregressive parameters, **gmvarKit** allows to constrain the unconditional means of some regimes to be the same. This feature is, however, only available for models that are parametrized with the unconditional means instead of intercepts (because some of the estimation is always done with mean-parametrization and one cannot generally swap the parametrization when constraints are imposed on means/intercepts). With the mean-parametrization employed (by setting `parametrization="mean"`), one may define groups of regimes that have the same mean parameters using the argument `same_means`. For instance, with three regime model ($M = 3$) the argument `same_means=list(c(1, 3), 2)` sets the unconditional means of the first and third regimes to be the same while allows the second regime to have different mean.

One can also combine linear constraints on the AR parameters with constraining some of the means to be the same. This allows, for instance, to estimate a model in which only the covariance matrix varies in time. To exemplify, the following code (which is not executed in this vignette) estimates a GMVAR($p = 4, M = 2$) model such that the unconditional means and autoregression matrices are constrained be the same in both regimes. The resulting model thereby has time-varying covariance matrix but otherwise it is linear.

```
R> I_pd2 <- diag(4*2^2) # The appropriate diagonal matrix for the constraint matrix
R> C3 <- rbind(I_pd2, I_pd2) # Stack them on top of each other
R> fit42cm <- fitGSMVAR(gdpdef, p=4, M=2, model="GMVAR", parametrization="mean",
+   same_means=list(1:2), constraints=C3, ncalls=16, ncores=8, seeds=1:16)
```

Using 8 cores for 16 estimations rounds...

Optimizing with a genetic algorithm...

```
|+++++| 100% elapsed=31s
```

Results from the genetic algorithm:

The lowest loglik: -227.251

The mean loglik: -225.743

The largest loglik: -224.648

Optimizing with a variable metric algorithm...

```
|+++++| 100% elapsed=03s
```

Results from the variable metric algorithm:

The lowest loglik: -223.311

The mean loglik: -223.311

The largest loglik: -223.311

Calculating approximate standard errors...

Finished!

Constraining the mixing weight parameters alphas to fixed values

It is also possible to constrain the mixing weight parameters $\alpha_1, \dots, \alpha_{M-1}$ to some fixed values.

To do so, specify the fixed values in the argument `weight_constraints` as a $(M - 1 \times 1)$ vector $(\alpha_1, \dots, \alpha_{M-1})$. Each element should be strictly between zero and one, and the sum of all of them should be strictly less than one. For instance, when $M = 2$, specifying `weight_constraints=0.6` constrains the mixing weights parameters of the first regime as $\alpha_1 = 0.6$ (and hence $\alpha_2 = 0.4$).

4.7. Testing parameter constraints

One way to assess the validity of the imposed constraints is to compare the values of information criteria of the constrained and unconstrained models. **gmvarKit**, however, also provides functions for testing the constraints with the likelihood ratio test, Wald test, and Rao's test, which are applicable as the ML estimator of a GSMVAR model has the conventional asymptotic distribution (as long as the model is correctly specified and one is willing to assume the validity of the required unverified assumptions; see Virolainen 2021b, Theorem 3, and Kalliovirta *et al.* (2016), Theorem 3). For a discussion on the likelihood ratio and Wald tests, see Buse (1982) and the references therein, for example.

The likelihood ratio test considers the null hypothesis that the true parameter value θ_0 satisfies some constraints imposed on these parameters (such that the constrained parameter space is a subset of the parameter space, which is presented in Virolainen 2021b, Assumption 2 for the GSMVAR models). Denoting by \hat{L}_U and \hat{L}_C the (maximized) log-likelihoods based on the unconstrained and constrained ML estimates, respectively, the test statistic takes the form

$$LR = 2(\hat{L}_U - \hat{L}_C). \quad (35)$$

Under the null, the test statistic is asymptotically χ^2 -distributed with the degrees of freedom given by the difference in the dimensions of the unconstrained and constrained parameter spaces. With **gmvarKit**, the likelihood ratio test can be calculated with the function `LR_test`, which takes the unconstrained model (a class `gsmvar` object) as its first argument and the constrained model as the second argument.

gmvarKit implements the Wald test of the null hypothesis

$$A\theta_0 = c, \quad (36)$$

where A is a $(k \times d)$ matrix with full row rank, c is a $(k \times 1)$ vector, θ_0 is the true parameter value, d is the dimension of the parameter space, and k is the number of constraints. The Wald test statistic takes the form

$$W = (A\hat{\theta} - c)'[A\mathcal{J}(\hat{\theta})^{-1}A']^{-1}(A\hat{\theta} - c), \quad (37)$$

where $\mathcal{J}(\hat{\theta})$ is the observed information matrix evaluated at the ML estimate $\hat{\theta}$. Under the null, the test statistic is asymptotically χ^2 -distributed with k degrees of freedom (which is the difference in dimensions of the constrained and unconstrained parameter spaces). With **gmvarKit**, the Wald test can be calculated with function `Wald_test`, which takes the estimated unconstrained model (as a class `gsmvar` object) as the first argument, the matrix A as the second argument, and the vector c as the third argument.

Rao's test is implemented to the function `Rao_test` (see function documentation on how to use it).

Note that the standard tests are not applicable if the number of GMVAR or StMVAR type regimes is chosen too large, as then some of the parameters are not identified, causing the result of the asymptotic normality of the ML estimator to break down. This particularly happens when one tests for the number of regimes in the model, as the under the null some of the regimes are reduced from the model⁴ (see the related discussion in Virolainen 2021b). Similar caution applies for testing whether a regime is of the GMVAR type against the alternative that it is of the StMVAR type: then $\nu_m = \infty$ under the null for the regime m to be tested, which violates the assumption that the parameter value is in the interior of a compact subset of the parameter space (see Virolainen 2021b, Theorem 3 and Assumption 1).

5. Quantile residual based model diagnostics

In the GSMVAR models, the empirical counterparts of the error terms $\varepsilon_{m,t}$ in (??) cannot be calculated, because the regime that generated each observation is unknown, making the conventional residual based diagnostics unavailable. Therefore, **gmvar** utilizes so called quantile residuals, which are suitable for evaluating adequacy of the GSMAR models.

Denote by y_t , $t = 1, 2, \dots$, the time series of interest and \mathcal{F}_{t-1} the σ -algebra generated by the random variables or vectors $\{y_{t-j}, j > 0\}$. Moreover, let θ denote the relevant parameter vector. Kalliovirta (2012) defines univariate quantile residuals as

$$R_{t,\theta} = \Phi^{-1}(F(y_t; \theta \mid \mathcal{F}_{t-1})), \quad (38)$$

where $\Phi(\cdot)^{-1}$ is the standard normal quantile function and $F(\cdot \mid \mathcal{F}_{t-1})$ is the conditional distribution function of the considered model.

Kalliovirta and Saikkonen (2010) define multivariate quantile residuals analogously to the univariate ones but by taking into account the dependence of the component time series from each other. Denote $\mathcal{A}_{j-1} = \sigma(y_{1,t}, \dots, y_{j-1,t})$ and by $f(\cdot \mid \sigma(\mathcal{F}_{t-1}, \mathcal{A}_{j-1})) = f_{j-1,t-1}(\cdot)$ the conditional density function conditional on the σ -algebra $\sigma(\mathcal{F}_{t-1}, \mathcal{A}_{j-1})$.

The conditional density function of the random vector y_t can be expressed in a product form by conditioning to the components y_t in addition to the history \mathcal{F}_{t-1} as

$$f(y_t; \theta \mid \mathcal{F}_{t-1}) = \prod_{j=1}^d f_{j-1,t-1}(y_{j,t}; \theta), \quad (39)$$

where $y_{j,t}$ is the j th component of y_t and $f_{0,t-1}(y_{1,t}; \theta) = f_{1,t-1}(y_{1,t}; \theta)$ is the marginal conditional density function of $y_{1,t}$ conditional on \mathcal{F}_{t-1} .

The conditional distribution functions corresponding to the density functions $f_{j-1,t-1}(\cdot; \theta)$ in (39) are of the form

$$F_{j-1,t-1}(y_{j,t}; \theta) = \int_{-\infty}^{y_{j,t}} f_{j-1,t-1}(u; \theta) du. \quad (40)$$

⁴Meitz and Saikkonen (2021) have, however, recently developed such tests for mixture models with Gaussian conditional densities

The multivariate quantile residuals are then defined as

$$R_{t,\theta} = \begin{bmatrix} R_{1t,\theta} \\ R_{2t,\theta} \\ \vdots \\ R_{dt,\theta} \end{bmatrix} = \begin{bmatrix} \Phi^{-1}(F_{0,t-1}(y_{1,t}; \theta)) \\ \Phi^{-1}(F_{1,t-1}(y_{2,t}; \theta)) \\ \vdots \\ \Phi^{-1}(F_{d-1,t-1}(y_{d,t}; \theta)) \end{bmatrix}, \quad (41)$$

and its empirical counterpart, $r_{t,\hat{\theta}}$, is obtained by replacing the parameter θ with its maximum likelihood estimate $\hat{\theta}$. Closed form expressions for the quantile residuals of the G-StMVAR model (which encompasses GMVAR and StMVAR models as special cases) are derived in Appendix B

For a correctly specified GSMVAR model employing the ML estimator, the empirical counterparts of multivariate quantile residuals are asymptotically multivariate standard normal (Kalliovirta and Saikkonen 2010, Lemma 3). They can therefore be utilized in graphical diagnostic similarly to the conventional Pearson's residuals. For the graphical diagnostics, **gmvar** provides function `diagnostic_plot` which plots the quantile residual time series, auto- and crosscorrelations of the quantile residuals, auto- and crosscorrelations of the squared quantile residuals, and normal quantile-quantile plots as well as histograms of the quantile residuals.

Kalliovirta and Saikkonen (2010) also propose three diagnostic tests for testing normality, autocorrelation, and conditional heteroskedasticity of the quantile residuals. The tests can be based either on the data or on a simulation procedure. If the sample is short, tests based on the data can be too forgiving, so to obtain more reliable test results the simulation procedure is recommended (with sample size of at least several thousand). The tests can be calculated with **gmvar** by using the function `quantile_residual_tests`. The simulation procedure is employed if the argument `nsim` is set larger the number of observations (in each component time series). In this case, `nsim` sets the length of the sample path used in the simulation procedure. If one is concerned about autocorrelation or conditional heteroskedasticity in a specific lag, the (standardized) individual statistics discussed in Kalliovirta and Saikkonen (2010) can be examined. The function `quantile_residual_tests` returns them automatically for the specified lags.

6. Impulse response analysis

6.1. Generalized impulse response function

We consider the generalized impulse response function (GIRF) (Koop *et al.* 1996) defined as

$$\text{GIRF}(n, \delta_j, \mathcal{F}_{t-1}) = E[y_{t+n} | \delta_j, \mathcal{F}_{t-1}] - E[y_{t+n} | \mathcal{F}_{t-1}], \quad (42)$$

where n is the chosen horizon, $\mathcal{F}_{t-1} = \sigma\{y_{t-j}, j > 0\}$ as before, the first term in the RHS is the expected realization of the process at time $t + n$ conditionally on a structural shock of magnitude $\delta_j \in \mathbb{R}$ in the j th element of e_t at time t and the previous observations, and the second term in the RHS is the expected realization of the process conditionally on the previous observations only. GIRF thus expresses the expected difference in the future outcomes when the specific structural shock hits the system at time t as opposed to all shocks being random.

Due to the p -step Markov property of the GSMVAR model, conditioning on (the σ -algebra generated by) the p previous observations $\mathbf{y}_{t-1} \equiv (y_{t-1}, \dots, y_{t-p})$ is effectively the same as conditioning on \mathcal{F}_{t-1} at the time t and later. The history \mathbf{y}_{t-1} can be either fixed or random, but with random history the GIRF becomes a random vector, however. Using fixed \mathbf{y}_{t-1} makes sense when one is interested in the effects of the shock in a particular point of time, whereas more general results are obtained by assuming that \mathbf{y}_{t-1} follows the stationary distribution of the process. If one is, on the other hand, concerned about a specific regime, \mathbf{y}_{t-1} can be assumed to follow the stationary distribution of the corresponding component model.

In practice, the GIRF and its distributional properties can be approximated with a Monte Carlo algorithm that generates independent realizations of the process and then takes the sample mean for point estimate. If \mathbf{y}_{t-1} is random and follows the distribution G , the GIRF should be estimated for different values of \mathbf{y}_{t-1} generated from G , and then the sample mean and sample quantiles can be taken to obtain the point estimate and confidence intervals. The algorithm implemented in **gmvarkit** is presented in Appendix C.

Because the GSMVAR model allows to associate specific features or economic interpretations for different regimes, it might be interesting to also examine the effects of a structural shock to the mixing weights $\alpha_{m,t}$, $m = 1, \dots, M$. We then consider the related GIRFs $E[\alpha_{m,t+n}|\delta_j, \mathbf{y}_{t-1}] - E[\alpha_{m,t+n}|\mathbf{y}_{t-1}]$ for which point estimates and confidence intervals can be constructed similarly to (42).

In **gmvarkit**, the GIRF can be estimated with the function **GIRF** which should be supplied with the estimated GSMVAR model or a GSMVAR built with hand-specified parameter values using the function **GSMVAR**. If a reduced form model is supplied to **GIRF**, recursive identification by lower-triangular Cholesky decomposition is automatically assumed. The size of the structural shock can be set with the argument **shock_size**. If not specified, the size of one standard deviation is used; that is, the size one. Among other arguments, the function may also be supplied with the argument **init_regimes** which specifies from which regimes' stationary distributions the initial values are generated from (if more than one regime is specified, the initial values will be generated from a mixture of the stationary distributions with the relative mixing proportions given by the mixing weight parameters). If more than one regime is specified, a mixture distribution with weights given by the mixing weight parameters is used. Alternatively, one may specify fixed initial values with the argument **init_values**. Note that the confidence intervals (whose confidence level can be specified with the argument **ci**) reflect uncertainty about the initial value only and not about the parameter estimates.

Due to the nonlinear nature of the model, GIRFs estimated from different starting values, or with different sign or magnitude of the shock, generally move the variables differently. Sometimes it is, however, of interest to scale the impulse responses so that they correspond to movement of some specific sign and magnitude of some specific variable. In **gmvarkit**, this is most conveniently achieved with the arguments **scale** and **scale_type**. The argument **scale** can be specified in order to scale the GIRFs to some of the shocks so that they correspond to a specific magnitude of instantaneous or peak response of some specific variable. For a single shock, it should be a length three vector where the shock of interest is given in the first element (an integer in $1, \dots, d$), the variable according to which the GIRFs should be scaled in the second element (an integer in $1, \dots, d$), and the magnitude of the given variable's instantaneous or peak response in the third element (a non-zero real number). If the GIRFs of multiple shocks should be scaled, provide a matrix which has one column for each of

the shocks with the columns being the length three vectors described above. The argument `scale_type` should be either `"instant"` or `"peak"` specifying whether you want to scale according to the instantaneous movement of peak response. If `"peak"`, the scale is based on the largest magnitude of peak response in absolute value. Scaling according to peak response won't be based on values after the horizon specified in the argument `"scale_horizon"`. Note that if you scale the GIRFs, the scaled GIRFs of mixing weights can be outside the interval from zero to one.

Because estimating the GIRF and their confidence intervals is computationally demanding, parallel computing is taken use of to shorten the estimation time. The number of CPU cores used can be set with the argument `ncores`. The objects returned by the `GIRF` function have their own `plot` and `print` methods. Also, cumulative impulse responses of the specified variables can be obtained directly by specifying the argument `which_cumulative`.

6.2. Generalized forecast error variance decomposition

We consider the generalized forecast error variance decomposition (GFEVD) (Lanne and Nyberg 2016) that is defined for variable i , shock j , and horizon n as

$$\text{GFEVD}(n, y_{it}, \delta_j, \mathcal{F}_{t-1}) = \frac{\sum_{l=0}^n \text{GIRF}(l, \delta_j, \mathcal{F}_{t-1})_i^2}{\sum_{k=1}^d \sum_{l=0}^n \text{GIRF}(l, \delta_k, \mathcal{F}_{t-1})_i^2}, \quad (43)$$

where n is the chosen and $\text{GIRF}(l, \delta_j, \mathcal{F}_{t-1})_i$ is the i th element of the related GIRF (see also the notation described for GIRF in the previous section). That is, the GFEVD is otherwise similar to the conventional forecast error variance decomposition but with GIRFs in the place of conventional impulse response functions. Because the GFEVDs sum to unity (for each variable), they can be interpreted in a similar manner to the conventional FEVD.

In `gmvarkit`, the GFEVD can be estimated with the function `GFEVD`. As with the GIRF, the GFEVD is dependent on the initial values. The type of the initial values is set with the argument `initval_type`, and there are three options:

1. `"data"` which estimates the GIRFs for all possible length p histories in the data and then the GIRFs in the GFEVD are obtained as the sample mean over those GIRFs.
2. `"random"` which generates the initial values from the stationary distribution of the process or from the mixture of the stationary distributions of some specific regime(s) with the relative mixing proportions given by the mixing weight parameters. The initial regimes can be set with the argument `init_regimes`. The GIRFs in the GFEVD are obtained as the sample mean over the GIRFs estimated for the different random initial values.
3. `"fixed"` which estimates the GIRFs for a single fixed initial value that is set with the argument `init_values`.

The shock size is the same for all scalar components of the structural shock and it can be adjusted with the argument `shock_size`. If the GIRFs for some variables should be cumulative before calculating the GFEVD, specify them with the argument `which_cumulative`. Finally, note that the GFEVD objects have their own `plot` and `print` methods.

6.3. Linear impulse response functions

It is also possible to calculate linear impulse response functions (IRF) based on a specific regime of the estimated model by using the function `linear_IRF`. If the autoregressive dynamics of the model are linear (i.e., either $M = 1$ or mean and AR parameters are constrained identical across the regimes), confidence bounds can be estimated based on a type of a fixed-design wild residual bootstrap method. `gmvarKit` implements the method proposed [Herwartz and Lütkepohl \(2014\)](#).

7. Building a GSMVAR model with specific parameter values

The function `GSMVAR` facilitates building GSMVAR models without estimation, for instance, in order to simulate observations from a GSMVAR process with specific parameter values. The parameter vector (of length $M(d + d^2p + d(d + 1)/2 + 2) - M_1 - 1$ for unconstrained reduced form models) has the form $\theta = (\vartheta_1, \dots, \vartheta_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ where

$$\vartheta_m = (\varphi_{m,0}, \text{vec}(A_{m,1}), \dots, \text{vec}(A_{m,p}), \text{vech}(\Omega_m)), \quad m = 1, \dots, M, \quad \text{and} \quad (44)$$

$$\nu = (\nu_{M_1+1}, \dots, \nu_M). \quad (45)$$

In the GMVAR model (when $M_1 = M$), the vector ν is omitted, as the GMVAR model does not contain degrees of freedom parameters. For models imposing additional constraints on the parameters, the parameter vectors are expressed in a different way. They are only presented in the package documentation for brevity, because the hand-specified parameter values can be set to satisfy any constraints as is.

In a structural GSMVAR model, the parameter vector has the form

$$\theta = (\varphi_{1,0}, \dots, \varphi_{M,0}, \text{vec}(\mathbf{A}_1), \dots, \text{vec}(\mathbf{A}_M), \text{vec}(W), \lambda_1, \dots, \lambda_M, \alpha_1, \dots, \alpha_{M-1}, \nu), \quad (46)$$

$$\mathbf{A}_m = (\text{vec}(A_{m,1}), \dots, \text{vec}(A_{m,p})) \quad (47)$$

$$\lambda_m = (\lambda_{m1}, \dots, \lambda_{md}). \quad (48)$$

For constrained structural models (including constraints on the structural parameters), see the documentation of `GSMVAR` (or any other relevant function).

In addition to the parameter vector, `GSMVAR` should be supplied with arguments `p` and `M` specifying the order of the model similarly to the estimation function `fitGSMVAR` discussed in Section 4.3. If one wishes to parametrize the model with the regimewise unconditional means (μ_m) instead of the intercepts ($\varphi_{m,0}$), the argument `parametrization` should be set to `"mean"` in which case the intercept parameters $\varphi_{m,0}$ are replaced with μ_m in the parameter vector. By default, `gmvarKit` uses intercept parametrization.

To exemplify, we build a reduced form StMVAR $p = 1$, $M = 1$, $d = 2$ model. The model has intercept parametrization and parameter values $\varphi_{1,0} = (0, 1)$ $\text{vec}(A_{1,1}) = (0.2, 0.2, 0.2, -0.2)$, $\text{vech}(\Omega_2) = (1, 0.1, 1)$, and $\nu_1 = 3$. After building the model, we use the `print` method to examine it:

```
R> params112 <- c(0, 1, 0.2, 0.2, 0.2, -0.2, 1, 0.1, 1, 3)
R> mod112 <- GSMVAR(p=1, M=1, d=2, params=params112, model="StMVAR")
R> mod112
```

Reduced form StMVAR model:

```
p = 1, M = 1, d = 2, #parameters = 10,
conditional log-likelihood, intercept parametrization, no AR parameter
constraints
```

Regime 1

Mixing weight: 1.00

Regime means: 0.22, 0.87

Df parameter: 3.00

```

      Y      phi0      A1      Omega      1/2
1 y1 = [ 0.00 ] + [ 0.20 0.20 ] y1.1 + ( [ 1.00 0.10 ] ) eps1
2 y2   [ 1.00 ]   [ 0.20 -0.20 ] y2.1 ( ARCH_mt [ 0.10 1.00 ] ) eps2
```

It is possible to include data in the models built with **GSMVAR** by either providing the data in the argument `data` when creating the model or by adding the data afterwards with the function `add_data`. When the model is supplied with data, the mixing weights, one-step conditional means and variances, and quantile residuals can be calculated and included in the model. The function `add_data` can also be used to update data to an estimated **GSMVAR** model without re-estimating the model.

8. Simulation and forecasting

8.1. Simulation

gmvarkit implements the S3 method `simulate` for simulating observations from **GSMVAR** processes (see `?simulate.gsmvar`). The method requires the process to be given as a class `gsmvar` object, which are typically created either by estimating a model with the function `fitGSMVAR` or by specifying the parameter values by hand and building the model with the constructor function `GSMVAR`. The initial values required to simulate the first p observations can be either set by hand (with the argument `init_values`) or drawn from the stationary distribution of the process (by default) or from a (mixture of) stationary distribution(s) of given regime(s). The argument `nsim` sets the length of the sample path to be simulated.

To give an example, the following code sets the random number generator seed to one and simulates 500 observations long sample from the StMVAR process built in Section 7:

```
R> mysim <- simulate(mod112, nsim=500, seed=1)
```

Our implementation of `simulate` returns a list containing the simulated sample path in `$sample`, the mixture component that generated each observation in `$component`, and the mixing weights in `$mixing_weights`.

8.2. Simulation based forecasting

Deriving multiple-steps-ahead point predictions and prediction intervals analytically for the **GSMVAR** models is very complicated, so **gmvarkit** employs the following simulation-based

method. By using the last p observations of the data up to the date of forecasting as initial values, a large number of sample paths for the future values of the process are simulated. Then, sample quantiles from the simulated sample paths are calculated to obtain prediction intervals, and the median or mean is used for point predictions. A similar procedure is also applied to forecast future values of the mixing weights, which might be of interest because the researcher can often associate specific characteristics to different regimes.

Forecasting is most conveniently done with the `predict` method (see `?predict.gsmvar`). The available arguments include the number of steps ahead to be predicted (`n_ahead`), the number sample paths the forecast is based on (`nsim`), possibly multiple confidence levels for prediction intervals (`pi`), prediction type (`pred_type`), and prediction interval type (`pi_type`). The prediction type can be either `median`, `mean`, or for one-step-ahead forecasts also the exact conditional mean, `cond_mean`. The prediction interval type can be any of `"two-sided"`, `"upper"`, `"lower"`, or `"none"`.

To exemplify, the following code forecasts the two-dimensional time-series of U.S. GDP and GDP deflator growth using the G-StMVAR(1,1,1) model `fit12gs` estimated in Section 4.3. The forecast is 10 steps (quarters in this case) ahead, based on 10000 Monte Carlo repetitions with the point forecast based on the mean of those repetitions. The prediction intervals are two-sided with confidence levels 0.95 and 0.90. Finally, the argument `mix_weights` states that also future values of the the mixing weights should be forecasted. After completing the forecast, the function plots the results by default.

```
R> mypred <- predict(fit12gs, n_ahead=10, nsim=10000, pred_type="mean",
+                   pi_type="two-sided", pi=c(0.95, 0.90),
+                   mix_weights=TRUE)
```

The resulting plot is presented in Figure ??.

9. Summary

Mixture vector autoregressive models are a valuable tool in modeling multivariate time series in which the data generating dynamics vary in time. We described the R package **gmvar**, which accommodates the GMVAR model (Kalliovirta *et al.* 2016), the StMVAR model (Virolainen 2021b), and the G-StMVAR model (Virolainen 2021b) - an appealing family of mixture vector autoregressive models that call the GSMVAR models. We discussed several features provided by **gmvar** for GSMVAR modeling: unconstrained and constrained maximum likelihood estimation of the model parameters, hypothesis testing, quantile residual based model diagnostics, estimation of generalized impulse response function and generalized forecast error variance decomposition, simulation, forecasting, and more. For convenience, we have collected some useful functions in **gmvar** to Table 1. For all the exported functions and their usage, see the reference manual.

Computational details

The results in this paper were obtained using R 4.1.2 and **uGMAR** 3.4.1 package running on MacBook Pro 14", 2021, with Apple M1 Pro processor, 16 Gt of unified RAM, and macOS Monterey 12.1 operating system.

Related to	Name	Description
Estimation	<code>fitGSMVAR</code>	Estimate a GSMVAR model.
	<code>alt_gsmvar</code>	Build a GSMVAR model based on results from any estimation round.
	<code>stmvar_to_gstmvar</code>	Estimate a G-StMVAR model based on a StMVAR (or G-StMVAR) model with large degrees of freedom parameters.
	<code>gsmvar_to_sgsmvar</code>	Obtain a structural GSMVAR model identified by heteroskedasticity based on a reduced form model.
	<code>iterate_more</code>	Run more iterations of the variable metric algorithm for a preliminary estimated GSMVAR model.
Estimates	<code>summary</code> (method)	Detailed printout of the estimates.
	<code>plot</code> (method)	Plot the series with the estimated mixing weights and a kernel density estimates of the (marginal) series with the (marginal) stationary densities of the model.
	<code>get_foc</code>	Calculate numerically approximated gradient of the log-likelihood function evaluated at the estimate.
	<code>get_soc</code>	Calculate eigenvalues of numerically approximated Hessian of the log-likelihood function evaluated at the estimate.
	<code>profile_logliks</code>	Plot the graphs of the profile log-likelihood functions.
	<code>cond_moment_plot</code>	Plot the model implied one-step conditional means or variances.
	<code>quantile_residual_tests</code>	Calculate quantile residual tests.
Diagnostics	<code>diagnostic_plot</code>	Plot quantile residual diagnostics.
Forecasting	<code>predict</code> (method)	Forecast future observations and mixing weights of the process.
Simulation	<code>simulate</code> (method)	Simulate from a GSMVAR process.
Create model	<code>GSMVAR</code>	Construct a GSMVAR model based on specific parameter values.
Hypothesis testing	<code>LR_test</code>	Calculate likelihood ratio test.
	<code>Wald_test</code>	Calculate Wald test.
	<code>Rao_test</code>	Calculate Rao's test.
Impulse response analysis	<code>GIRF</code>	Estimate generalized impulse response functions.
	<code>GFEVD</code>	Estimate generalized forecast error variance decomposition.
	<code>linear_IRF</code>	Estimate linear impulse response functions.
Other	<code>add_data</code>	Add data to a GSMVAR model
	<code>swap_parametrization</code>	Swap between mean and intercept parametrizations

Table 1: Some useful functions in **gmvar** sorted according to their usage. The note "method" in parentheses after the name of a function signifies that it is an S3 method for a class **gsmvar** object (often generated by the function `fitGSMVAR` or `GSMVAR`).

Some of the estimation results (and thereby everything that is calculated based on the estimates) may vary slightly when running the code on different computers. This is likely due to the numerical error caused by the limited precision of the float point representation.

References

- Anderson H, Vahid F (1998). “Testing multiple equation systems for common nonlinear components.” *Journal of Econometrics*, **84**(1), 1–36. doi:[10.1016/S0304-4076\(97\)00076-6](https://doi.org/10.1016/S0304-4076(97)00076-6).
- Angelini G, Bacchiocchi E, Caggiano G, Fanelli L (2019). “Uncertainty across volatility regimes.” *Journal of Applied Econometrics*, **34**(3), 437–455. doi:[10.1002/jae.2672](https://doi.org/10.1002/jae.2672).
- Aomoto K, Kita M (2011). *Theory of Hypergeometric Functions*. 1st edition. Springer-Verlag, Tokyo. doi:[10.1007/978-4-431-53938-4](https://doi.org/10.1007/978-4-431-53938-4).
- Bacchiocchi E, Castelnovo E, Fanelli L (2016). “Gimme a Break! Identification and Estimation of the Macroeconomic Effects of Monetary Policy Shocks in the U.S.” *Melbourne Institute Working Paper No. 31/16*. doi:[10.2139/ssrn.2851316](https://doi.org/10.2139/ssrn.2851316).
- Bacchiocchi E, Fanelli L (2015). “Identification in Structural Vector Autoregressive models with Structural Changes, with an Application to US Monetary policy.” *Oxford Bulletin of Economics and Statistics*, **77**(6), 761–779. doi:[10.1111/obes.12092](https://doi.org/10.1111/obes.12092).
- Blondel V, Nesterov Y (2005). “Computationally Efficient Approximations of the Joint Spectral Radius.” *SIAM Journal on Matrix Analysis and Applications*, **27**(1), 256–272. doi:[10.1137/040607009](https://doi.org/10.1137/040607009).
- Burgard J, Neuenkirch M, Nöckel M (2019). “State-Dependent Transmission of Monetary Policy in the Euro Area.” *Journal of Money, Credit and Banking*, **51**(7), 2053–2070. doi:[10.1111/jmcb.12592](https://doi.org/10.1111/jmcb.12592).
- Buse A (1982). “The Likelihood Ratio, Wald, and Lagrange Multiplier Tests: An Expository Note.” *The American Statistician*, **36**(3a), 153–157. doi:[10.1080/00031305.1982.10482817](https://doi.org/10.1080/00031305.1982.10482817).
- Chang CT, Blondel V (2013). “An experimental study of approximation algorithms for the joint spectral radius.” *Numerical Algorithms*, **64**, 181–202. doi:[10.1007/s11075-012-9661-z](https://doi.org/10.1007/s11075-012-9661-z).
- Ding P (2016). “On the Conditional Distribution of the Multivariate t Distribution.” *The American Statistician*, **70**(3), 293–295. doi:[10.1080/00031305.2016.1164756](https://doi.org/10.1080/00031305.2016.1164756).
- Hankin R, Clausen A, Murdoch D (2006). “Special functions in R: introducing the gsl package.” *R News*, **6**(4).
- Herwartz H, Lütkepohl H (2014). “Structural vector autoregressions with Markov switching: Combining conventional with statistical identification of shocks.” *Journal of Econometrics*, **183**(1), 104–116. doi:[doi:doi.org/10.1016/j.jeconom.2014.06.012](https://doi.org/10.1016/j.jeconom.2014.06.012).
- Hubrich K, Teräsvirta T (2013). “Thresholds and Smooth Transitions in Vector Autoregressive Models.” *CREATES Research Paper 2013-18*, Aarhus University. URL https://pure.au.dk/ws/files/54473123/rp13_18.pdf.
- Jungers R (2023). *The JSR toolbox* (<https://www.mathworks.com/matlabcentral/fileexchange/33202-the-jsr-toolbox>), MATLAB Central File Exchange.

- Kalliovirta L (2012). “Misspecification tests based on quantile residuals.” *The Econometrics Journal*, **15**(2), 358–393. doi:[10.1111/j.1368-423X.2011.00364.x](https://doi.org/10.1111/j.1368-423X.2011.00364.x).
- Kalliovirta L, Meitz M, Saikkonen P (2016). “Gaussian mixture vector autoregression.” *Journal of Econometrics*, **192**(2), 465–498. doi:[10.1016/j.jeconom.2016.02.012](https://doi.org/10.1016/j.jeconom.2016.02.012).
- Kalliovirta L, Saikkonen P (2010). “Reliable Residuals for Multivariate Nonlinear Time Series Models.” *Unpublished revision of HECER discussion paper No. 247*. URL <https://blogs.helsinki.fi/lkvaisan/files/2010/08/ReliableResiduals.pdf>.
- Kheifets I, Saikkonen P (2020). “Stationarity and ergodicity of vector STAR models.” *Econometric Reviews*, **39**(407–414), 1311–1324. doi:[10.1080/07474938.2019.1651489](https://doi.org/10.1080/07474938.2019.1651489).
- Kilian L, Lütkepohl H (2017). *Structural Vector Autoregressive Analysis*. 1st edition. Cambridge University Press, Cambridge. doi:[10.1017/9781108164818](https://doi.org/10.1017/9781108164818).
- Koop G, Pesaran M, Potter S (1996). “Impulse response analysis in nonlinear multivariate models.” *Journal of Econometrics*, **74**(1), 119–147. doi:[10.1016/0304-4076\(95\)01753-4](https://doi.org/10.1016/0304-4076(95)01753-4).
- Lanne M, Nyberg H (2016). “Generalized Forecast Error Variance Decomposition for Linear and Nonlinear Multivariate Models.” *Oxford Bulletin of Economics and Statistics*, **78**(4), 595–603. doi:[10.1111/obes.12125](https://doi.org/10.1111/obes.12125).
- Lütkepohl H (2005). *New Introduction to Multiple Time Series Analysis*. 1st edition. Springer, Berlin. doi:[10.1007/978-3-540-27752-1](https://doi.org/10.1007/978-3-540-27752-1).
- Lütkepohl H, Netšunajev A (2017). “Structural vector autoregressions with smooth transitions in variances.” *Journal of Economic Dynamics & Control*, **84**, 43–57. doi:[10.1016/j.jedc.2017.09.001](https://doi.org/10.1016/j.jedc.2017.09.001).
- Meitz M, Preve D, Saikkonen P (2023). “A mixture autoregressive model based on Student’s *t*-distribution.” *Communications in Statistics - Theory and Methods*, **52**(2), 499–515. doi:[10.1080/03610926.2021.1916531](https://doi.org/10.1080/03610926.2021.1916531).
- Meitz M, Saikkonen P (2021). “Testing for observation-dependent regime switching in mixture autoregressive models.” *Journal of Econometrics*, **222**(1), 601–624. doi:[10.1016/j.jeconom.2020.04.048](https://doi.org/10.1016/j.jeconom.2020.04.048).
- Muirhead R (1982). *Aspects of Multivariate Statistical Theory*. 1st edition. John Wiley & Sons, Hoboken, New Jersey. doi:[10.1002/9780470316559](https://doi.org/10.1002/9780470316559).
- Nash J (1990). *Compact Numerical Methods for Computers. Linear Algebra and Function Minimization*. 2nd edition. Adam Hilger, Bristol and New York. doi:[10.1201/9781315139784](https://doi.org/10.1201/9781315139784).
- Parrilo P, Jadbabaie A (2008). “Approximation of the joint spectral radius using sum of squares.” *Linear Algebra and its Applications*, **428**(10), 2385–2402. doi:[10.1016/j.laa.2007.12.027](https://doi.org/10.1016/j.laa.2007.12.027).
- R Core Team (2022). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL <https://www.R-project.org/>.

- Rigobon R (2003). “Identification through Heteroskedasticity.” *The Review of Economics and Statistics*, **85**(4), 777–792. doi:[10.1162/003465303772815727](https://doi.org/10.1162/003465303772815727).
- Solymos P, Zawadzki Z (2020). **pbapply**: Adding Progress Bar to ‘*apply’ Functions. R package version 1.4-3, URL <https://CRAN.R-project.org/package=pbapply>.
- Teräsvirta T (1994). “Specification, Estimation, and Evaluation of Smooth Transition Autoregressive Models.” *Journal of the American Statistical Association*, **89**(425), 208–2187. doi:[10.2307/2291217](https://doi.org/10.2307/2291217).
- Tsay R (1998). “Testing and Modeling Multivariate Threshold Models.” *Journal of the American Statistical Association*, **93**(443), 1188–1202. doi:[10.1080/01621459.1998.10473779](https://doi.org/10.1080/01621459.1998.10473779).
- Virolainen S (2018a). *gmvarKit: Estimate Gaussian and Student’s t Mixture Vector Autoregressive Models*. R package version 2.1.0 available at CRAN: <https://CRAN.R-project.org/package=gmvarKit>, URL <https://CRAN.R-project.org/package=gmvarKit>.
- Virolainen S (2018b). *uGMAR: Estimate Univariate Gaussian and Student’s t Mixture Autoregressive Models*. R package version 3.4.2 available at CRAN: <https://CRAN.R-project.org/package=uGMAR>, URL <https://CRAN.R-project.org/package=uGMAR>.
- Virolainen S (2021a). “Gaussian and Student’s *t* mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks in the Euro area.” *Unpublished working paper, available as arXiv:2109.13648*. doi:[10.48550/arXiv.2109.13648](https://doi.org/10.48550/arXiv.2109.13648). URL <https://doi.org/10.48550/arXiv.2109.13648>.
- Virolainen S (2021b). “Structural Gaussian Mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks.” *Unpublished working paper, available as arXiv:2007.04713*. URL <https://arxiv.org/abs/2007.04713>.
- Virolainen S (2022a). *Essays on mixture autoregressive models with applications to macroeconomics and finance*. Ph.D. thesis, University of Helsinki. URL <http://hdl.handle.net/10138/350319>.
- Virolainen S (2022b). “A mixture autoregressive model based on Gaussian and Student’s *t*-distributions.” *Studies in Nonlinear Dynamics & Econometrics*, **26**(4), 559–580. doi:[10.1515/snde-2020-0060](https://doi.org/10.1515/snde-2020-0060).
- Virolainen S (2024). *sstvars: Toolkit for Structural Smooth Transition Vector Autoregressive models*. R package version 0.1.0 available at GitHub: <https://github.com/saviviro/sstvars>, URL <https://github.com/saviviro/sstvars>.

A. Properties of multivariate Gaussian and Student's t distribution

Denote a d -dimensional real valued vector by y . It is well known that the density function of a d -dimensional Gaussian distribution with mean μ and covariance matrix Σ is

$$n_d(y; \mu, \Sigma) = (2\pi)^{-d/2} \det(\Sigma)^{-1/2} \exp \left\{ -\frac{1}{2} (y - \mu)' \Sigma^{-1} (y - \mu) \right\}. \quad (49)$$

Similarly to ? but differing from the standard form, we parametrize the Student's t -distribution using its covariance matrix as a parameter together with the mean and the degrees of freedom. The density function of such a d -dimensional t -distribution with mean μ , covariance matrix Σ , and $\nu > 2$ degrees of freedom is

$$t_d(y; \mu, \Sigma, \nu) = C_d(\nu) \det(\Sigma)^{-1/2} \left(1 + \frac{(y - \mu)' \Sigma^{-1} (y - \mu)}{\nu - 2} \right)^{-(d+\nu)/2}, \quad (50)$$

where

$$C_d(\nu) = \frac{\Gamma\left(\frac{d+\nu}{2}\right)}{\sqrt{\pi^d (\nu - 2)^d} \Gamma\left(\frac{\nu}{2}\right)}, \quad (51)$$

and $\Gamma(\cdot)$ is the gamma function. We assume that the covariance matrix Σ is positive definite for both distributions.

Consider a partition $X = (X_1, X_2)$ of either Gaussian or t -distributed (with ν degrees of freedom) random vector X such that X_1 has dimension $(d_1 \times 1)$ and X_2 has dimension $(d_2 \times 1)$. Consider also a corresponding partition of the mean vector $\mu = (\mu_1, \mu_2)$ and the covariance matrix

$$\Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma'_{12} & \Sigma_{22} \end{bmatrix}, \quad (52)$$

where, for example, the dimension of Σ_{11} is $(d_1 \times d_1)$. In the Gaussian case, X_1 then has the marginal distribution $n_{d_1}(\mu_1, \Sigma_{11})$ and X_2 has the marginal distribution $n_{d_2}(\mu_2, \Sigma_{22})$. In the Student's t case, X_1 has the marginal distribution $t_{d_1}(\mu_1, \Sigma_{11}, \nu)$ and X_2 has the marginal distribution $t_{d_2}(\mu_2, \Sigma_{22}, \nu)$ (see, e.g., [Ding \(2016\)](#), also in what follows).

When X has Gaussian distribution, the conditional distribution of the random vector X_1 given $X_2 = x_2$ is

$$X_1 \mid (X_2 = x_2) \sim n_{d_1}(\mu_{1|2}(x_2), \Sigma_{1|2}(x_2)), \quad (53)$$

where

$$\mu(x_2) \equiv \mu_{1|2}(x_2) = \mu_1 + \Sigma_{12} \Sigma_{22}^{-1} (x_2 - \mu_2) \quad \text{and} \quad (54)$$

$$\Omega \equiv \Sigma_{1|2}(x_2) = \Sigma_{11} - \Sigma_{12} \Sigma_{22}^{-1} \Sigma'_{12}. \quad (55)$$

When X has t -distribution, the conditional distribution of the random vector X_1 given $X_2 = x_2$ is

$$X_1 \mid (X_2 = x_2) \sim t_{d_1}(\mu_{1|2}(x_2), \Sigma_{1|2}(x_2), \nu + d_2), \quad (56)$$

where

$$\mu(x_2) = \mu_{1|2}(x_2) = \mu_1 + \Sigma_{12} \Sigma_{22}^{-1} (x_2 - \mu_2) \quad \text{and} \quad (57)$$

$$\Omega(x_2) \equiv \Sigma_{1|2}(x_2) = \frac{\nu - 2 + (x_2 - \mu_2)' \Sigma_{22}^{-1} (x_2 - \mu_2)}{\nu - 2 + d_2} (\Sigma_{11} - \Sigma_{12} \Sigma_{22}^{-1} \Sigma'_{12}). \quad (58)$$

In particular, we have

$$n_d(x; \mu, \Sigma) = n_{d_1}(x_1; \mu_{1|2}(x_2), \Sigma_{1|2}(x_2)) t_{d_2}(x_2; \mu_2, \Sigma_{22}) \quad \text{and} \quad (59)$$

$$t_d(x; \mu, \Sigma, \nu) = t_{d_1}(x_1; \mu_{1|2}(x_2), \Sigma_{1|2}(x_2), \nu + d_2) t_{d_2}(x_2; \mu_2, \Sigma_{22}, \nu). \quad (60)$$

B. Quantile residuals of the G-StMVAR model

The conditional density function of the d -dimensional G-StMVAR process y_t conditional on \mathcal{F}_{t-1} is

$$f_{t-1}(y_t; \theta) = \sum_{m=1}^{M_1} \alpha_{m,t} n_d(y_t; \mu_{m,t}, \Omega_m) + \sum_{m=M_1+1}^M \alpha_{m,t} t_d(y_t; \mu_{m,t}, \Omega_{m,t}, \nu_m + dp), \quad (61)$$

where $n_d(\cdot; \mu_{m,t}, \Omega_m, \nu_m + dp)$ is the density function of d -dimensional normal distribution with mean $\mu_{m,t}$ and covariance matrix Ω_m ; and $t_d(\cdot; \mu_{m,t}, \Omega_{m,t}, \nu_m + dp)$ is the density function of d -dimensional t -distribution with mean $\mu_{m,t}$, covariance matrix $\Omega_{m,t}$, and $\nu_m + dp$ degrees of freedom.

Denote $y_t^{(k)} = (y_{1,t}, \dots, y_{k,t})$ ($k \times 1$), $k \leq d$, $\mu_{m,t}^{(k)} = (\mu_{1,m,t}, \dots, \mu_{k,m,t})$ ($k \times 1$), $k \leq d$, and by $\Omega_{m,t}^{(k)}$ ($\Omega_m^{(k)}$) the upper left ($k \times k$) block matrix of $\Omega_{m,t}$ (Ω_m). Then, the properties of the marginal distributions of multivariate Gaussian and t -distributions (see Appendix A) show that conditional on \mathcal{F}_{t-1} , the random vectors $y_t^{(j)}$, $j = 1, \dots, d$, follow the distribution that is a mixture M_1 j -dimensional normal distributions (with means $\mu_{m,t}^{(j)}$ and covariance matrices $\Omega_m^{(j)}$) and $M_2 \equiv M - M_1$ j -dimensional t -distributions (with means $\mu_{m,t}^{(j)}$, covariance matrices $\Omega_{m,t}^{(j)}$, and $\nu_m + dp$ degrees of freedom). The mixing weights $\alpha_{m,t}$ are not affected, as they are \mathcal{F}_{t-1} -measurable. Therefore, the marginal density function of $y_t^{(j)}$ is

$$f_{t-1}(y_t^{(j)}; \theta) = \sum_{m=1}^{M_1} \alpha_{m,t} n_j(y_t^{(j)}; \mu_{m,t}^{(j)}, \Omega_m^{(j)}) + \sum_{m=M_1+1}^M \alpha_{m,t} t_j(y_t^{(j)}; \mu_{m,t}^{(j)}, \Omega_{m,t}^{(j)}, \nu_m + dp), \quad (62)$$

The conditional density function $f_{0,t-1}(y_{1,t}; \theta)$ in (39) is obtained from (62) by choosing $j = 1$. For $j = 2, \dots, d$, the conditional density functions $f_{j-1,t-1}(y_{j,t}; \theta)$ are obtained by substituting the equation (62) to the formula of conditional density function:

$$f_{j-1,t-1}(y_{j,t}; \theta) = \frac{f_{t-1}(y_t^{(j)}; \theta)}{f_{t-1}(y_t^{(j-1)}; \theta)} = \frac{\sum_{m=1}^{M_1} \alpha_{m,t} n_j(y_t^{(j)}; \mu_{m,t}^{(j)}, \Omega_m^{(j)}) + \sum_{m=M_1+1}^M \alpha_{m,t} t_j(y_t^{(j)}; \mu_{m,t}^{(j)}, \Omega_{m,t}^{(j)}, \nu_m + dp)}{\sum_{n=1}^{M_1} \alpha_{n,t} n_{j-1}(y_t^{(j-1)}; \mu_{n,t}^{(j-1)}, \Omega_n^{(j-1)}) + \sum_{n=M_1+1}^M \alpha_{n,t} t_{j-1}(y_t^{(j-1)}; \mu_{n,t}^{(j-1)}, \Omega_n^{(j-1)}, \nu_n + dp)}. \quad (63)$$

It follows from the properties of the conditional distributions of multivariate normal distribution that we may express the j -dimensional normal distributions as

$$n_j(y_t^{(j)}; \mu_{m,t}^{(j)}, \Omega_m^{(j)}) = n_1(y_{j,t}; \mu_{m,t,j|j-1}, \Omega_{m,j|j-1}) n_{j-1}(y_t^{(j-1)}; \mu_{m,t}^{(j-1)}, \Omega_m^{(j-1)}), \quad (64)$$

where $\mu_{m,t,j|j-1}$ and $\Omega_{m,j|j-1}$ are the conditional mean and covariance matrix of $y_{j,t}$ conditional on $\sigma(\mathcal{A}_{j-1}, \mathcal{F}_{t-1})$. Likewise, it follows from the properties of the conditional distributions of multivariate t -distribution that we may express the j -dimensional t -distributions as

$$t_j(y_t^{(j)}; \mu_{m,t}^{(j)}, \Omega_{m,t}^{(j)}, \nu_m + dp) = t_1(y_{j,t}; \mu_{m,t,j|j-1}, \Omega_{m,t,j|j-1}, \nu_m + dp + j - 1) \times t_{j-1}(y_t^{(j-1)}; \mu_{m,t}^{(j-1)}, \Omega_{m,t}^{(j-1)}, \nu_m + dp), \quad (65)$$

where $\mu_{m,t,j|j-1}$ and $\Omega_{m,t,j|j-1}$ are the conditional mean and covariance matrix of $y_{j,t}$ conditional on $\sigma(\mathcal{A}_{j-1}, \mathcal{F}_{t-1})$.

By denoting

$$\beta_{m,t,j} \equiv \frac{\alpha_{m,t} n_{j-1}(y_t^{(j-1)}; \mu_{m,t}^{(j-1)}, \Omega_m^{(j-1)})}{\sum_{n=1}^{M_1} \alpha_{n,t} n_{j-1}(y_t^{(j-1)}; \mu_{n,t}^{(j-1)}, \Omega_n^{(j-1)}) + \sum_{n=M_1+1}^M \alpha_{n,t} t_{j-1}(y_t^{(j-1)}; \mu_{n,t}^{(j-1)}, \Omega_{n,t}^{(j-1)}, \nu_n + dp)} \quad (66)$$

for $m = 1, \dots, M_1$, $j = 2, \dots, d$, and

$$\beta_{m,t,j} \equiv \frac{\alpha_{m,t} t_{j-1}(y_t^{(j-1)}; \mu_{m,t}^{(j-1)}, \Omega_{m,t}^{(j-1)}, \nu_m + dp)}{\sum_{n=1}^{M_1} \alpha_{n,t} n_{j-1}(y_t^{(j-1)}; \mu_{n,t}^{(j-1)}, \Omega_n^{(j-1)}) + \sum_{n=M_1+1}^M \alpha_{n,t} t_{j-1}(y_t^{(j-1)}; \mu_{n,t}^{(j-1)}, \Omega_{n,t}^{(j-1)}, \nu_n + dp)} \quad (67)$$

for $m = M_1 + 1, \dots, M$, $j = 2, \dots, d$, and using the expressions (64) and (65), we can express the conditional density function (63) as

$$f_{j-1,t-1}(y_{j,t}; \theta) = \sum_{m=1}^{M_1} \beta_{m,t,j} n_1(y_{j,t}; \mu_{m,t,j|j-1}, \Omega_{m,j|j-1}) + \sum_{m=M_1+1}^M \beta_{m,t,j} t_1(y_{j,t}; \mu_{m,t,j|j-1}, \Omega_{m,t,j|j-1}, \nu_m + dp + j - 1), \quad j = 2, \dots, d. \quad (68)$$

For $m = 1, \dots, M_1$, the conditional means $\mu_{m,t,j|j-1}$ and covariance matrices $\Omega_{m,j|j-1}$ are as in (54) and (55) when for each $j = 2, \dots, d$ and $m = 1, \dots, M$ we consider the partition $y_t^{(j)} = (y_t^{(j-1)}, y_{j,t})$, $\mu_{m,t}^{(j)} = (\mu_{m,t}^{(j-1)}, \mu_{j,m,t})$, and

$$\Omega_m^{(j)} = \begin{bmatrix} \Omega_m^{(j-1)} & \Omega_{(j-1),j,m} \\ \Omega'_{(j-1),j,m} & \Omega_m(j,j) \end{bmatrix}, \quad (69)$$

where $\Omega_m(j,j)$ is the j th element of Ω_m and $\Omega_{(j-1),j,m}$ $((j-1) \times 1)$ consists of the rows $1, \dots, j-1$ of the j th column of Ω_m . In particular, we have

$$\mu_{m,t,j|j-1} = \mu_{j,m,t} + \Omega'_{(j-1),j,m} (\Omega_m^{(j-1)})^{-1} (y_t^{(j-1)} - \mu_{m,t}^{(j-1)}), \quad (70)$$

$$\Omega_{m,j|j-1} = \Omega_m(j,j) - \Omega'_{(j-1),j,m} (\Omega_m^{(j-1)})^{-1} \Omega_{(j-1),j,m}. \quad (71)$$

For $m = M_1 + 1, \dots, M$, the conditional means $\mu_{m,t,j|j-1}$ and covariance matrices $\Omega_{m,t,j|j-1}$ are as in (57) and (58) when for each $j = 2, \dots, d$ and $m = 1, \dots, M$ we consider the partition $y_t^{(j)} = (y_t^{(j-1)}, y_{j,t})$, $\mu_{m,t}^{(j)} = (\mu_{m,t}^{(j-1)}, \mu_{j,m,t})$, and

$$\Omega_{m,t}^{(j)} = \begin{bmatrix} \Omega_{m,t}^{(j-1)} & \Omega_{(j-1),j,m,t} \\ \Omega'_{(j-1),j,m,t} & \Omega_{m,t}(j,j) \end{bmatrix}, \quad (72)$$

where $\Omega_{m,t}(j, j)$ is the jj th element of $\Omega_{m,t}$ and $\Omega_{(j-1),j,m,t}$ $((j-1) \times 1)$ consists of the rows $1, \dots, j-1$ of the j th column of $\Omega_{m,t}$. In particular, taking use of the relation $\Omega_{m,t} = \omega_{m,t}\Omega_m$ (where $\omega_{m,t}$ is scalar), we have

$$\begin{aligned}\mu_{m,t,j|j-1} &= \mu_{j,m,t} + \Omega'_{(j-1),j,m,t}(\Omega_{m,t}^{(j-1)})^{-1}(y_t^{(j-1)} - \mu_{m,t}^{(j-1)}) \\ &= \mu_{j,m,t} + \Omega'_{(j-1),j,m}(\Omega_m^{(j-1)})^{-1}(y_t^{(j-1)} - \mu_{m,t}^{(j-1)}),\end{aligned}\quad (73)$$

and

$$\begin{aligned}\Omega_{m,t,j|j-1} &= \frac{\nu_m + dp + (y_t^{(j-1)} - \mu_{m,t}^{(j-1)})'(\Omega_{m,t}^{(j-1)})^{-1}(y_t^{(j-1)} - \mu_{m,t}^{(j-1)})}{\nu_m + dp + j - 3} \tilde{\Omega}_{m,t,j|j-1} \\ &= \frac{\nu_m + dp + \omega_{m,t}^{-1}(y_t^{(j-1)} - \mu_{m,t}^{(j-1)})'(\Omega_m^{(j-1)})^{-1}(y_t^{(j-1)} - \mu_{m,t}^{(j-1)})}{\nu_m + dp + j - 3} \tilde{\Omega}_{m,t,j|j-1},\end{aligned}\quad (74)$$

where

$$\begin{aligned}\tilde{\Omega}_{m,t,j|j-1} &\equiv \Omega_{m,t}(j, j) - \Omega'_{(j-1),j,m,t}(\Omega_{m,t}^{(j-1)})^{-1}\Omega_{(j-1),j,m,t} \\ &= \omega_{m,t}(\Omega_m(j, j) - \Omega'_{(j-1),j,m}(\Omega_m^{(j-1)})^{-1}\Omega_{(j-1),j,m}).\end{aligned}\quad (75)$$

It then remains to find expressions for the conditional distribution functions $F_{j-1,t-1}(y_{j,t}; \boldsymbol{\theta})$, $j = 1, \dots, d$, in (41). For notational convenience, we write

$$\begin{aligned}f_{j-1,t-1}(y_{j,t}; \boldsymbol{\theta}) &= \sum_{m=1}^{M_1} \beta_{m,t,j} n_1(y_{j,t}; \mu_{m,t,j|j-1}, \Omega_{m,j|j-1}) \\ &\quad + \sum_{m=M_1+1}^M \beta_{m,t,j} t_1(y_{j,t}; \mu_{m,t,j|j-1}, \Omega_{m,t,j|j-1}, \nu_m + dp + j - 1)\end{aligned}\quad (76)$$

for all $j = 1, \dots, d$ by defining $\beta_{m,t,1} \equiv \alpha_{m,t}$, $\mu_{m,t,1|0} \equiv \mu_{m,t}^{(1)}$, $\Omega_{m,1|0} \equiv \Omega_m^{(1)}$, and $\Omega_{m,t,1|0} \equiv \Omega_{m,t}^{(1)}$. For $j = 2, \dots, d$, these quantities are defined in (67), (70), (71), (73), and (74). Then,

$$\begin{aligned}F_{j-1,t-1}(y_{j,t}; \boldsymbol{\theta}) &= \sum_{m=1}^{M_1} \beta_{m,t,j} \int_{-\infty}^{y_{j,t}} n_1(u; \mu_{m,t,j|j-1}, \Omega_{m,j|j-1}) du \\ &\quad + \sum_{m=M_1+1}^M \beta_{m,t,j} \int_{-\infty}^{y_{j,t}} t_1(u; \mu_{m,t,j|j-1}, \Omega_{m,t,j|j-1}, \nu_m + dp + j - 1) du,\end{aligned}\quad (77)$$

where we seek to solve the integrals inside the sums.

Regarding the first sum, for $m = 1, \dots, M_1$, it is easy to see that the integrals can be expressed using the standard normal distribution function $\Phi(\cdot)$ as

$$\int_{-\infty}^{y_{j,t}} n_1(u; \mu_{m,t,j|j-1}, \Omega_{m,j|j-1}) du = \Phi\left(\frac{y_{j,t} - \mu_{m,t,j|j-1}}{\sqrt{\Omega_{m,j|j-1}}}\right).\quad (78)$$

Next, consider the second sum, $m = M_1 + 1, \dots, M$. By taking use of the symmetry of the t -distribution about its mean, we obtain

$$\begin{aligned}&\int_{-\infty}^{y_{j,t}} t_1(u; \mu_{m,t,j|j-1}, \Omega_{m,t,j|j-1}, \nu_m + dp + j - 1) du \\ &= \frac{1}{2} + \int_{\mu_{m,t,j|j-1}}^{y_{j,t}} t_1(u; \mu_{m,t,j|j-1}, \Omega_{m,t,j|j-1}, \nu_m + dp + j - 1) du.\end{aligned}\quad (79)$$

By applying the change of variables $\tilde{u}_{m,t,j} = u - \mu_{m,t,j|j-1}$ in the integral, the RHS of (79) can be expressed as

$$\frac{1}{2} + \frac{\Gamma\left(\frac{\nu_m + dp + j}{2}\right)}{\sqrt{\pi(\nu_m + dp + j - 3)}\Gamma\left(\frac{\nu_m + dp + j - 1}{2}\right)} \Omega_{m,t,j|j-1}^{-1/2} \int_0^{\tilde{y}_{m,t,j}} \left(1 + \frac{\tilde{u}_{m,t,j}^2}{a_{m,t,j}}\right)^{-b_{m,j}} d\tilde{u}_{m,t,j}, \quad (80)$$

where $\tilde{y}_{m,t,j} \equiv y_{j,t} - \mu_{m,t,j|j-1}$, $a_{m,t,j} \equiv (\nu_m + dp + j - 3)\Omega_{m,t,j|j-1}$, and $b_{m,j} \equiv (\nu_m + dp + j)/2$. Then, by applying the change of variables $z_{m,t,j} = \tilde{u}_{m,t,j}^2/\tilde{y}_{m,t,j}$, we can express the integral in (80) as

$$\int_0^{\tilde{y}_{m,t,j}} \left(1 + \frac{\tilde{u}_{m,t,j}^2}{a_{m,t,j}}\right)^{-b_{m,j}} d\tilde{u}_{m,t,j} = \frac{1}{2} \int_0^{\tilde{y}_{m,t,j}} \left(\frac{\tilde{y}_{m,t,j}}{z_{m,t,j}}\right)^{1/2} \left(1 + \frac{z_{m,t,j}\tilde{y}_{m,t,j}}{a_{m,t,j}}\right)^{-b_{m,j}} dz_{m,t,j}. \quad (81)$$

By applying the third change of variables $x_{m,t,j} = z_{m,t,j}/\tilde{y}_{m,t,j}$ and using the properties of the gamma function, the RHS of (81) can be expressed as

$$\frac{\tilde{y}_{m,t,j}}{2} \int_0^1 x_{m,t,j}^{-1/2} \left(1 - x_{m,t,j} \left(-\frac{\tilde{y}_{m,t,j}^2}{a_{m,t,j}}\right)\right)^{-b_{m,j}} dx_{m,t,j} = \tilde{y}_{m,t,j} \times {}_2F_1\left(\frac{1}{2}, b_{m,j}, \frac{3}{2}; -\frac{\tilde{y}_{m,t,j}^2}{a_{m,t,j}}\right), \quad (82)$$

where the hypergeometric function is defined as (Aomoto and Kita 2011, Section 1.3.1)

$${}_2F_1(a, b, c; x) = \frac{\Gamma(c)}{\Gamma(a)\Gamma(c-a)} \int_0^1 s^{a-1} (1-s)^{c-a-1} (1-sx)^{-b} ds, \quad (83)$$

when $|x| < 1$, $a > 0$, and $c - a > 0$ (when $a, c \in \mathbb{R}$).

Using the above result, we have

$$\begin{aligned} & \int_{-\infty}^{y_{j,t}} t_1(u; \mu_{m,t,j|j-1}, \Omega_{m,t,j|j-1}, \nu_m + dp + j - 1) du \\ &= \frac{1}{2} + \frac{\Gamma\left(\frac{\nu_m + dp + j}{2}\right)}{\sqrt{\pi(\nu_m + dp + j - 3)}\Gamma\left(\frac{\nu_m + dp + j - 1}{2}\right)} \Omega_{m,t,j|j-1}^{-1/2} \tilde{y}_{m,t,j} \times {}_2F_1\left(\frac{1}{2}, b_{m,j}, \frac{3}{2}; -\frac{\tilde{y}_{m,t,j}^2}{a_{m,t,j}}\right) \end{aligned} \quad (84)$$

whenever $\left| -\frac{\tilde{y}_{m,t,j}^2}{a_{m,t,j}} \right| < 1$. That is, the closed form expression (84) exists when

$$|y_{j,t} - \mu_{m,t,j|j-1}| < \sqrt{(\nu_m + dp + j - 3)\Omega_{m,t,j|j-1}}. \quad (85)$$

If this condition does not hold, the quantile residual are obtained by numerically integrating the conditional density function $t_1(u; \mu_{m,t,j|j-1}, \Omega_{m,t,j|j-1}, \nu_m + dp + j - 1)$. For the hypergeometric function, **gmvarkit** uses the package **gsl** (Hankin, Clausen, and Murdoch 2006).

C. Monte Carlo algorithm

We present a Monte Carlo algorithm that produces point estimates and with random initial value $\mathbf{y}_{t-1} = (y_{t-1}, \dots, y_{t-p})$ confidence intervals for the generalized impulse response function

defined in (42). Our algorithm is adapted from Koop *et al.* (1996, pp. 135-136) and Kilian and Lütkepohl (2017, pp. 601-602). We assume that the history \mathbf{y}_{t-1} follows a known distribution G , which may be such that it produces a single outcome with probability one (corresponding to a fixed \mathbf{y}_{t-1}), or it can be the stationary distribution of the process or of a specific regime. In the following, $y_{t+h}^{(i)}(\delta_j, \mathbf{y}_{t-1})$ denotes a realization of the process at time $t+h$ conditional on the structural shock of magnitude δ_j in the j th element of \mathbf{e}_t hitting the system at time t and on the p observations $\mathbf{y}_{t-1} = (y_{t-1}, \dots, y_{t-p})$ preceding the time t , whereas $y_{t+h}^{(i)}(\mathbf{y}_{t-1})$ denotes an alternative realization conditional on the history \mathbf{y}_{t-1} only.

The algorithm proceeds with the following steps.

0. Decide the horizon H , the numbers of repetitions R_1 and R_2 , and the magnitude δ_j for the j th structural shock that is of interest.
1. Draw an initial value \mathbf{y}_{t-1} from G .
2. Draw $H+1$ independent realizations of a shock ε_t from $N(0, I_d)$. If the models contains Student's t regime, the Gaussian shocks are used obtain shocks from the appropriate Student's t distributions (in this case, $H+1$ independent realizations of shocks from $\chi_{\nu_m+dp}^2$ -distributions are also drawn for the t -regimes). Also, draw an initial regime $m \in \{1, \dots, M\}$ according to the probabilities given by the mixing weights $\alpha_{1,t}, \dots, \alpha_{M,t}$ and compute the reduced form shock u_t . If identification by heteroskedasticity is used, $u_t = W\Lambda_m^{1/2}\varepsilon_t$, where $\Lambda_1 = I_d$. If the shocks are, instead, identified recursively, $u_t = L_t\varepsilon_t$, where L_t is the lower triangular matrix obtained from the Cholesky decomposition $\Omega_t = L_tL_t'$ and Ω_t is the conditional covariance matrix of the process. Then, compute the structural shock $e_t = B_t^{-1}u_t$ and impose the size δ_j on its j th element to obtain e_t^* . Finally, calculate the modified reduced form shock $u_t^* = B_te_t^*$.
3. Use the modified reduced form shock u_t^* and the rest H standard normal shocks ε_t obtained from Step 2 to compute realizations $y_{t+h}^{(i)}(\delta_j, \mathbf{y}_{t-1})$ for $n = 0, 1, \dots, H$, iterating forward so that in each iteration the regime m that generates the observation is first drawn according to the probabilities given by the mixing weights. At $h = 0$, the initial regime and the modified reduced form shock u_t^* calculated from the structural shock in Step 2 is used. From $h = 1$ onwards, the $h+1$ th standard normal shock ε_t is used to calculate the reduced form shock $u_{t+h} = W\Lambda_m^{1/2}\varepsilon_{t+h}$, where $\Lambda_1 = I_d$ and m is the selected regime (note that only the reduced form shocks are of interest with $h > 0$, so the same formula can be used for recursively identified shocks here).
4. Use the reduced form shock u_t and the rest H the standard normal shocks ε_t obtained from Step 2 to compute realizations $y_{t+h}^{(i)}(\mathbf{y}_{t-1})$ for $h = 0, 1, \dots, H$, so that the reduced form shock u_t (calculated in Step 2) is used to compute the time $h = 0$ realization. Otherwise proceed similarly to the previous step.
5. Calculate $y_{t+h}^{(i)}(\delta_j, \mathbf{y}_{t-1}) - y_{t+h}^{(i)}(\mathbf{y}_{t-1})$.
6. Repeat Steps 2-5 R_1 times and calculate the sample mean of $y_{t+h}^{(i)}(\delta_j, \mathbf{y}_{t-1}) - y_{t+h}^{(i)}(\mathbf{y}_{t-1})$ for $h = 0, 1, \dots, H$ to obtain an estimate of the GIRF($h, \delta_j, \mathbf{y}_{t-1}$).
7. Repeat Steps 1-6 R_2 times to obtain estimates of GIRF($h, \delta_j, \mathbf{y}_{t-1}$) with different starting values \mathbf{y}_{t-1} generated from the distribution G . Then, take the sample mean and

sample quantiles over the estimates to obtain point estimate and confidence intervals for the GIRF with random initial value.

Notice that if a fixed initial value \mathbf{y}_{t-1} is used, Step 7 is redundant.

Affiliation:

Savi Virolainen

Faculty of Social Sciences

University of Helsinki

P. O. Box 17, FI-0014 University of Helsinki, Finland

E-mail: savi.virolainen@helsinki.fi