FARS: Factor Augmented Regression Scenarios in R

Gian Pietro Bellocca
Universidad Carlos III de Madrid

Ignacio Garrón
Universidad Carlos III de Madrid

C. Vladimir Rodríguez-Caballero
Department of Statistics, ITAM, Mexico.

Esther Ruiz
Universidad Carlos III de Madrid

Abstract

Obtaining realistic scenarios for the distribution of key economic variables is crucial for econometricians, policy-makers, and financial analysts. The **FARS** package provides a comprehensive framework in R for modeling and designing economic scenarios based on distributions derived from multi-level dynamic factor models (ML-DFMs) and factor-augmented quantile regressions (FA-QRs). The package enables users to: (i) extract global and block-specific factors using a flexible multi-level factor structure; (ii) compute asymptotically valid confidence regions for the estimated factors, accounting for uncertainty in the factor loadings; (iii) estimate FA-QRs; (iv) recover full predictive conditional densities from quantile forecasts; and (v) estimate the conditional density when the factors are stressed.

Keywords: Multi-level dynamic factor model, Quantile regression, Scenario analysis, R.

1. Introduction

There is a growing interest in developing new econometric tools to create extreme scenarios for the distribution of economic and financial variables. Constructing such scenarios can help understand the resilience of economic systems by providing early warning signals of what to expect should such conditions materialize in adverse outlooks. In pursuit of this goal, González-Rivera, Rodríguez-Caballero, and Ruiz (2024) propose a methodology to obtain stressed densities of target variables by combining three procedures: i) fitting dynamic factor models, ii) applying subsampling methods, and iii) estimating factor-augmented quantile regressions.

The methodology assumes that underlying economic and/or financial latent factors drive the density of the target variable. A dynamic factor model (DFM) extracts such unobservable components from a large set of potential predictors. The preferred estimation method is Principal Components (PC); see, for example, Bai (2003) and Bai and Ng (2013) for technical details. Over the last few decades, the DFM has been generalized in several directions to accommodate economic and financial applications more effectively. In particular, multi-level DFMs (ML-DFMs) have been used to extract latent factors from predictors grouped into blocks.

The factor structure of the ML-DFM allows for pervasive (or global) factors that are common across all variables in the system, as well as block-specific (or regional) factors associated with one or more blocks. The model can incorporate either non-overlapping blocks of variables, as in Breitung and Eickmeier (2016), or overlapping blocks, as proposed by Rodríguez-Caballero and Caporin (2019). Given its flexible structure, factor extraction in ML-DFMs is often based on the sequential least squares (LS) method, initially proposed by Breitung and Eickmeier (2016).

Once the latent factors have been extracted, the next step involves generating stressed scenarios (or stressed factors) for the conditional densities. To this end, the methodology proposed by González-Rivera, Maldonado, and Ruiz (2019) is employed. Under unexpected and rare circumstances, the factors driving the distribution of the variable of interest are under stress and, thus, deviate substantially from their averages. Stressed factors are probabilistically derived based on their multidimensional distribution, focusing on the observations located in the extreme autocontours of this distribution.

Finally, similarly to Adrian, Boyarchenko, and Giannone (2019), the quantiles of the distribution of the target variable can be estimated by fitting factor-augmented quantile regressions (FA-QRs) with the estimated (stressed or unstressed) factors as regressors. Then, following Azzalini and Capitanio (2003), the corresponding h-step-ahead conditional density is obtained using the estimated quantiles together with a skew-t distribution. This density delivers any quantile of interest in the absence of stress conditions; that is, when the underlying factors are around their averages.

This paper presents the **FARS** package, which provides a comprehensive framework in R for modeling and forecasting conditional densities based on ML-DFM and FA-QRs.[1] The package enables users to: i) extract pervasive, semipervasive, and block-specific factors using a flexible multi-level factor structure; ii) compute asymptotically valid confidence regions for the estimated factors, accounting for uncertainty in the factor loadings; iii) estimate FA-QRs; iv) recover full predictive conditional densities from these quantile forecasts; and v) estimate the density when the factors are stressed. The functionalities of the package are illustrated by building scenarios for the density of U.S. growth, as in González-Rivera *et al.* (2024).

Some alternative implementations of DFMs are available in the R programming language. The **sparseDFM** package implements popular estimation methods for DFMs, including the recent Sparse DFM approach by Mosley, Chan, and Gibberd (2024); see Mosley, Chan, and Gibberd (2023). The **MARSS, KFAS** packages provide a flexible framework for modeling DFMs within state-space structures (Holmes, Ward, Scheuerell, and Wills (2023) and Helske (2017)). Furthermore, the **dfms** package offers a broad suite of DFM estimation techniques under the assumption of independent and identically distributed (i.i.d.) idiosyncratic components (Krantz, Bagdziunas, Tikka, and Holmes 2025). In contrast, implementations of ML-DFM remain scarce. To the best of our knowledge, the only available package in R is **GCCfactor**, which supports model selection, estimation, bootstrap inference, and simulation for the model (see Lin and Shin (2023)). Nevertheless, the case of overlapping-block ML-DFMs based on PC and generalized canonical correlation (CC) estimation techniques is not supported by any existing package to date.

The rest of this paper is organized as follows. The methodology is briefly described in Section 2. Section 3 describes the code. Section 4 is devoted to illustrating the capabilities of the **FARS** package in the context of estimating the conditional density of economic growth in

[1] Version 0.5.0 of the **FARS** package is available on CRAN: `https://CRAN.R-project.org/package=FARS`.

the US as a function of underlying domestic and international factors. Finally, Section 5 concludes with a summary.

2. Methodology

In this section, we provide a brief description of the methodology for obtaining conditional density forecasts of the target variable under standard economic dynamics and stressed scenarios of the underlying factors. The section is structured in three parts. First, we discuss the factor structures involved in our specifications: DFM and ML-DFM with and without overlapping blocks. Second, we explain two methods for estimating the asymptotic multidimensional distribution of the estimated factors, assuming that idiosyncratic components are either cross-sectionally uncorrelated or weakly correlated. Third, we describe the procedure for obtaining full-density forecasts for the target variable under both stressed and non-stressed scenarios using FA-QRs.

2.1. Dynamic Factor Model (DFM)

The DFM has been extensively studied in the literature to reduce the dimensionality of large sets of variables by assuming that they can be represented by a relatively small number of common underlying factors; see, for example, Stock and Watson (2002a,b), Bai (2003), and Bai and Ng (2013). Consider $X_t = (x_{1t}, ..., x_{Nt})'$, the $N \times 1$ vector of weakly stationary variables at time $t = 1, ..., T$. The DFM is given by

$$X_t = PF_t + \epsilon_t, \tag{1}$$

where $P = (p_1', ..., p_N')'$ is the $N \times r$ matrix of factor loadings, $F_t = (F_{1t}, \dots, F_{rt})'$ is an $r \times 1$ vector of latent factors, and $\epsilon_t = (\epsilon_{1t}, ..., \epsilon_{Nt})'$ is the $N \times 1$ vector of idiosyncratic components, which are assumed to be cross-sectionally weakly correlated, and uncorrelated with the common factors F_t. F_t and ϵ_t are weakly stationary processes. Finally, the number of factors, r, is known.

The identification condition in (1) is standard in the literature. It assumes that $\frac{1}{T}F'F = I_r$, and that $\frac{1}{N}P'P$ is a diagonal matrix with distinct elements on the main diagonal, ordered from largest to smallest. Under these restrictions, the estimated factors are identified up to a sign transformation; see Bai and Ng (2013) for further details in the context of PC estimation.

In practice, the factors are often estimated using PC. Let $X = (X_1, \dots, X_T)'$ denote the $T \times N$ matrix of observed data. The PC-estimated factors, $\hat{F}_t$, are obtained as $\sqrt{T}$ times the eigenvectors associated with the r largest eigenvalues of the matrix XX', ordered in decreasing magnitude. The corresponding loading matrix is then estimated by $P' = \frac{1}{T}F'X$.

2.2. Multi-level Dynamic Factor Model (ML-DFM)

In many economic or financial applications, the variables in X_t are naturally grouped into blocks, such as countries, geographical regions, or economic sectors. In some cases, not all variables in X_t load onto all factors in the DFM, which implies the presence of zeros in P. The standard PC approach is suboptimal in this context, as it neglects the block structure. Consequently, when the block structure is known, a more appropriate approach is to extract the factors from a ML-DFM, where the relevant zero restrictions are imposed directly on

P. In what follows, we present two alternative specifications of the ML-DFM, depending on whether the blocks of variables overlap.

ML-DFM without overlapping blocks

Breitung and Eickmeier (2016) propose the following ML-DFM with non-overlapping blocks. Denote by $X_{k,t}$ the $N \times 1$ [2] vector of variables within block $k = 1, ..., K$ such that $X_t = (X_{1,t}, ..., X_{K,t})'$ with a total cross-sectional dimension of $N \times K$. The specification is as follows.

$$\begin{bmatrix} X_{1,\cdot t} \\ \vdots \\ X_{K,\cdot t} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu}_1 & \boldsymbol{\lambda}_1 & 0 & \dots & 0 \\ \boldsymbol{\mu}_2 & 0 & \boldsymbol{\lambda}_2 & \dots & 0 \\ \vdots & & 0 & \ddots & 0 \\ \boldsymbol{\mu}_K & 0 & 0 & \dots & \boldsymbol{\lambda}_K \end{bmatrix} \begin{bmatrix} G_t \\ F_{1,t} \\ F_{2,t} \\ \vdots \\ F_{K,t} \end{bmatrix} + \begin{bmatrix} \epsilon_{1,\cdot t} \\ \vdots \\ \epsilon_{K,\cdot t} \end{bmatrix}, \tag{2}$$

$$X_t = P^* F_t^* + \epsilon_t,$$

where $F_t^* = \left(G_t', F_{1,t}', \dots, F_{K,t}'\right)'$ and $P^* = [M_k, \Lambda_k]$. $G_t = (G_{1,t}, \dots, G_{r_G,t})'$ is the $r_G \times 1$ vector of pervasive factors, which load on all variables in the system while $F_{k,t} = (F_{1,t}, \dots, F_{r_k,t})'$ is the $r_k \times 1$ vector of block-specific factors, which load only within the block $X_{k,t}$. The loading matrix and the idiosyncratic noise are defined conformably; see Breitung and Eickmeier (2016) and Choi, Kim, Kim, and Kwark (2018) for further technical details and identification conditions.

ML-DFM with overlapping blocks

For clarity of exposition of the ML-DFM with overlapping blocks, consider the case with $K = 3$; see Rodríguez-Caballero and Caporin (2019) for a detailed description.[3] Assume the presence of pervasive factors, G_t, and block-specific factors, $F_{k,t} = \left(F_{1,t}', F_{2,t}', F_{3,t}'\right)'$, as described earlier. In addition to these, a general factor structure may also include pairwise (or semipervasive) factors, $F_{kj,t} = \left(F_{12,t}', F_{13,t}', F_{23,t}'\right)'$. For instance, the factor $F_{12,t}$ loads only on the variables in blocks $X_{1,t}$ and $X_{2,t}$; that is, the semipervasive factor captures the commonality only between blocks 1 and 2 without any dependence on block 3. This type of factor structure is illustrated in Figure 1, which visually represents the relationships between pervasive, semipervasive, and block-specific factors.

The model is written as

$$x_{k,it} = \mu_{k,i}' G_t + \kappa_{kj_i}' F_{kj,t} + \lambda_{k,i}' F_{k,t} + \epsilon_{k,it},$$

where $k = 1, 2, 3$ indicates the block, index $i = 1, \dots, N$ denotes the $i'th$ cross-section unit of block k, $t = 1, \dots, T$ is the time dimension, and kj means interaction between blocks k and

[2] To simplify notation, we assume that every block of data has the same cross-sectional dimension; nevertheless, in practical situations, such dimensions may vary. In this sense, if N_k denotes the cross-sectional dimensional of the Block k, the total number of cross-sectional units in the model is $N = N_1 + N_2 + \cdots + N_k$.

[3] The ML-DFM in (3) can be extended to more than three blocks. The **FARS** package supports $K > 3$ blocks, including triple-wise (and higher-order) interactions. However, the computational burden naturally increases when the number of blocks and/or the order of interactions increases.

$j \in 1, 2, \ldots, k$ with $k \neq j$. $\mu_{k,i}, \kappa_{kj_i}$, and $\lambda_{k,i}$ are the $\mathbf{r}_G, \mathbf{r}_{F_{kj}}$, and $\mathbf{r}_{F_k}$- dimensional factor loadings. The number of pervasive, pairwise, and block-specific factors can naturally vary in each block k. The idiosyncratic term denoted by $\epsilon_{k,it}$ satisfies the standard assumptions of the DFM previously introduced.

The three-block ML-DFM with overlapping blocks can be rewritten as

$$\begin{bmatrix} X_{1,\cdot t} \\ X_{2,\cdot t} \\ X_{3,\cdot t} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu}_1 & \boldsymbol{\kappa}_{12_1} & \boldsymbol{\kappa}_{13_1} & 0 & \boldsymbol{\lambda}_1 & 0 & 0 \\ \boldsymbol{\mu}_2 & \boldsymbol{\kappa}_{12_2} & 0 & \boldsymbol{\kappa}_{23_2} & 0 & \boldsymbol{\lambda}_2 & 0 \\ \boldsymbol{\mu}_3 & 0 & \boldsymbol{\kappa}_{13_3} & \boldsymbol{\kappa}_{23_3} & 0 & 0 & \boldsymbol{\lambda}_3 \end{bmatrix} \begin{bmatrix} G_t \\ F_{12,t} \\ F_{13,t} \\ F_{23,t} \\ F_{1,t} \\ F_{2,t} \\ F_{3,t} \end{bmatrix} + \begin{bmatrix} \epsilon_{1,\cdot t} \\ \epsilon_{2,\cdot t} \\ \epsilon_{3,\cdot t} \end{bmatrix} , \tag{3}$$

$$X_t = P^* F_t^* + \epsilon_t,$$

where $F_t^* = \left(G_t', F_{12,t}', F_{13,t}', F_{23,t}', F_{1,t}', F_{2,t}', F_{3,t}'\right)'$ and $\Lambda^* = [M_k, K_k, \Lambda_k]$. Note that the total number of unobservable common factors involved in (3) is $\mathbf{r}_G + \mathbf{r}_{F_{12}} + \mathbf{r}_{F_{13}} + \mathbf{r}_{F_{23}} + \mathbf{r}_{F_1} + \mathbf{r}_{F_2} + \mathbf{r}_{F_3}$. Hallin and Liška (2011) and Ergemen and Rodríguez-Caballero (2023) propose a simple methodology based on the inclusion-exclusion principle from set theory to determine the number of pervasive, semipervasive and block-specific factors.

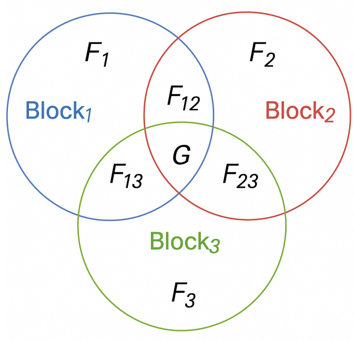


Figure 1: Factor structure formed by three different overlapping blocks of data.

Sequential least squares estimation

Estimation of the ML-DFM is based on the sequential approach proposed by Breitung and Eickmeier (2016) in which the main goal is to minimize the following residual sums of squares (RSS) function:

$$S(\hat{F}_t, \hat{P}) = \sum_{t=1}^{T} \left(X_t - \hat{P}\hat{F}_t\right)' \left(X_t - \hat{P}\hat{F}_t\right), \tag{4}$$

by a sequence of LS regressions. The algorithm can be executed for the general case of K blocks with overlapping factors as follows:

1. Obtain the initial values of the factors as follows:

(a) Employ canonical correlation analysis (CCA) on $X_{k,t}$ to obtain initial estimates of the global factor, $\hat{G}^{(0)} = \left(\hat{G}_1^{(0)}, \hat{G}_2^{(0)}, \dots, \hat{G}_T^{(0)}\right)'$.

(b) Filter out the global component by regressing $X_{k,t}$ on $\hat{G}^{(0)}$, and get the corresponding residuals, $X_{k,t}^{*(0)}$, from each of the K separate regressions.

(c) Employ CCA on $X_{k,t}^{*(0)}$ to obtain the following lower-level factors, selecting the corresponding blocks.

(d) Regress $X_{k,t}^{*(0)}$ on the respective lower-level factors involved and get the residuals.

(e) Steps c) and d) are executed sequentially until the initial estimates of the pairwise block factors are obtained. Denote by $X_{k,it}^{**(0)}$ the residuals after filtering the pairwise factors of each block k.

(f) Run PC on $X_{k,t}^{**(0)}$ to get the specific-block factors $\hat{F}^{(0)} = \left(\hat{F}_{1,t}^{(0)}, \hat{F}_{2,t}^{(0)}, \dots, \hat{F}_{k,t}^{(0)}\right)'$.

(g) The initial matrix of loadings, $\hat{P}^{(0)}$, is estimated through time-series regressions of $X_{k,t}$ on the global factors, $X_{k,t}^{*}$ on the semi-pervasive factors, and $X_{k,t}^{**}$ on the non-pervasive factors.

2. Updated estimates for the unobservable factors $\hat{F}^{(1)}$ are obtained by LS regression of $X_{k,t}$ on $\hat{P}^{(0)}$ as follows $\hat{F}^{(1)} = \left(\hat{P}^{(0)\prime}\hat{P}^{(0)}\right)^{-1}\hat{P}^{(0)\prime}X_{k,t}$.

3. The updated factors $\hat{F}^{(1)}$ are used to obtain the associated loadings matrix, $\hat{P}^{(1)}$, as in Step 1.

4. Steps 2 and 3 are repeated until the RSS converges to a minimum, from which $\hat{F}^{*}$ and $\hat{P}^{*}$ are obtained.

As can be seen, the algorithm does not impose any normalization step. Henceforth, even though the vector of common components $P^{*}F_t^{*}$ is consistently estimated and just-identified, the factor and loading matrices themselves are not separately identified; they can only be estimated consistently up to a rotation of the factor space.

Breitung and Eickmeier (2016) adapt the standard normalization step in PC analysis to separately identify P^{*} and F_t^{*}. The first step requires orthogonalizing the different levels of estimated factors (pervasive, pairwise, and block-specific) with respect to one another. A practical implementation consists of recursively regressing each factor on the previously ordered ones and using the residuals as updated, orthogonalized estimates. For instance, block-specific factors can be regressed on pairwise factors, and the resulting residuals can then be regressed on pervasive factors. Since each regression corresponds to a projection operation, this sequential procedure is equivalent to applying the Gram-Schmidt orthogonalization process to the vector of estimated factors F_t^{*}, following a predetermined ordering.[4] Finally, the normalized pervasive factors are obtained as the top $\mathbf{r}_G$ principal components of

[4]This sequential orthogonalization procedure, though operationally implemented through regressions, reflects the structure of the Gram-Schmidt process and leverages the projection logic underpinning the famous Frisch–Waugh–Lovell (FWL) theorem in regression analysis. While we are not estimating coefficients, the residuals obtained from regressing one factor level on another correspond to their orthogonal components, as in the FWL decomposition.

the estimated common components. These are derived from the nonzero eigenvalues and the corresponding eigenvectors of the matrix

$$\widehat{M}\left(\frac{1}{T}\sum_{t=1}^{T}\widehat{G}_t\widehat{G}_t'\right)\widehat{M}'.$$

The same normalization procedure can be applied to the semipervasive and block-specific factors, using the sample covariance matrices of their respective common components.

As explained earlier, the algorithm requires a suitable initialization of P^* and F_t^*. The **FARS** package provides two initialization options: CCA and, alternatively, PC. While both approaches yield approximately the same estimated common components $P^*F_t^*$, CCA typically leads to faster convergence, requiring fewer iterations to minimize the RSS. However, when the factor structure is highly complex, initializing with PC tends to be computationally more efficient. See also Breitung and Eickmeier (2016) for the small sample properties of the sequential LS estimator with CCA and PC for the two-level DFM.

2.3. Probability distribution of factors

Constructing probabilistic scenarios requires knowledge of the joint distribution of the unobservable factors. The asymptotic distribution of such factors obtained from the DFM by PCA in (1) is derived by Bai (2003). He shows that if $\frac{F'F}{T}=I_r$ and $\frac{\sqrt{N}}{T}\to 0$ when $N,T\to\infty$, the asymptotic distribution of $\widehat{F}_t$, at each moment, t, is given by

$$\sqrt{N}\left(\widehat{F}_t-F_t\right)\xrightarrow{d}N\left(0,\Sigma_P^{-1}\Gamma_t\Sigma_P^{-1}\right),\tag{5}$$

where $\Sigma_P=\lim_{N\to\infty}\frac{P'P}{N}$ and $\Gamma_t=\lim_{N\to\infty}\sum_{i=1}^{N}\sum_{j=1}^{N}p_ip_j'E(\varepsilon_{it}\varepsilon_{jt})$ with p_i and ε_{it} being defined as in the DFM in (1). The finite sample approximation of the asymptotic covariance matrix of $\widehat{F}_t$ can be estimated as follows:

$$MSE_t=\left(\frac{\hat{P}'\hat{P}}{N}\right)^{-1}\frac{\hat{\Gamma}_t}{N}\left(\frac{\hat{P}'\hat{P}}{N}\right)^{-1},\tag{6}$$

where $\hat{\Gamma}_t$ is a consistent estimator for Γ_t. Under the assumption of cross-sectionally uncorrelated idiosyncratic components, Bai and Ng (2006) propose the following estimator:

$$\widehat{\Gamma}_t^{BN}=\frac{1}{N}\sum_{i=1}^{N}\hat{p}_i\hat{p}_i'\hat{\varepsilon}_{it}^2,\tag{7}$$

where $\hat{\varepsilon}_{it}=x_{it}-\hat{p}_i'\hat{F}_t$ are the residuals from the DFM model.

In many empirical settings, assuming that the idiosyncratic covariance matrix Σ_ϵ is diagonal imposes a stringent restriction that may not hold in practice. Therefore, alternatively, we can relax this assumption and allow the idiosyncratic components to be weakly cross-sectionally correlated. Under those circumstances, Γ_t can be consistently estimated as proposed by Fresoli, Poncela, and Ruiz (2024) by using adaptive thresholding of the sample covariances of the idiosyncratic residuals, $\hat{\sigma}_{ij}$, as follows:

$$\widetilde{\Gamma}^{FPR}=\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{N}\hat{p}_i\hat{p}_j'\frac{1}{T}\sum_{t=1}^{T}\hat{\varepsilon}_{it}\hat{\varepsilon}_{jt}I\left(\mid\hat{\sigma}_{ij}\mid\geq c_{ij}\right),\tag{8}$$

where $I(\dot{})$ is the indicator function that takes value one when the argument is true and zero otherwise, and $c_{ij} = \delta\omega_{NT}\left[\widehat{Var}\left[\hat{\varepsilon}_{it}\hat{\varepsilon}_{jt}\right]\right]^{1/2}$, with $\widehat{Var}\left[\hat{\varepsilon}_{it}\hat{\varepsilon}_{jt}\right] = \frac{1}{T}\sum_{t=1}^{T}\left[\hat{\varepsilon}_{it}\hat{\varepsilon}_{jt} - \hat{\sigma}_{ij}\right]^2$, $\omega_{NT} = \frac{1}{\sqrt{N}} + \sqrt{\frac{\log(N)}{T}}$, and δ chosen as proposed by Qiu and Liyanage (2019).

It is important to note that the estimator of Γ_t in (8) requires stationarity and, consequently, is constant over time. However, the estimator in (7), which does not require stationarity, may not be adequate for moderate levels of cross-sectional idiosyncratic correlation.

The asymptotic covariance matrix estimated as in (6) does not account for the uncertainty arising from the estimation of the loading matrix, regardless of whether Γ_t is obtained from (7) or (8). In this light, Maldonado and Ruiz (2021) propose a correction of the asymptotic MSE based on subsampling in the cross-sectional space subsets of series of size $N^* < N$, with each series in the subsample containing all temporal observations. For each subsample, the loadings and factors are estimated by PC, obtaining $\hat{F}_t^{*(s)}$ and $\hat{P}^{*(s)}$, for $s = 1, ..., S$. The corrected finite sample approximation of the asymptotic MSE of $\hat{F}_t$ can be estimated as follows:

$$MSE_t^* = \frac{1}{N}\left(\frac{\hat{P}'\hat{P}}{N}\right)^{-1}\hat{\Gamma}_t\left(\frac{\hat{P}'\hat{P}}{N}\right)^{-1} + \frac{N^*}{NS}\sum_{s=1}^{S}\left(\left(\hat{F}_t^{*(s)} - \hat{F}_t\right)\left(\hat{F}_t^{*(s)} - \hat{F}_t\right)'\right), \qquad (9)$$

where $\hat{\Gamma}_t$ can be estimated as in (7) or in (8).

Based on the asymptotic normality result in (5), Maldonado and Ruiz (2021) construct confidence ellipsoids for the estimated factors with coverage probability $100 \times \alpha\%$ as follows:

$$g(F_t, \alpha) = \{F_t \in \mathbb{R}^r | (F_t - \hat{F}_t) MSE_t^{*-1}(F_t - \hat{F}_t) \leq \chi^2_{r(\alpha)}\}, \qquad (10)$$

where $\chi^2_{r(\alpha)}$ is the α-quantile of the χ^2 distribution with r degrees of freedom, with r being the number of factors. Each point on the surface of the ellipsoid represents a possible joint realization of all factors in the DFM. These boundary points correspond to extreme, yet plausible, stress conditions.

2.4. Density Forecasts Under Stressed and Non-Stressed Conditions

Estimated factors can be used to summarize the information contained in a large set of predictors X_t, which are used to estimate the temporal evolution of the conditional density of a target variable. In this subsection, we describe how these densities can be obtained under both stressed and non-stressed conditions for the underlying factors.

Let y_t be the observation at time t of the target variable. We start by obtaining h-step-ahead forecasts of the τ^*-quantile of the conditional distribution of y_t by estimating the following FA-QR:

$$q_{\tau^*}(y_{t+h} \mid y_t, F_t) = \mu(\tau^*, h) + \phi(\tau^*, h)y_t + \sum_{k=1}^{r}\beta_k(\tau^*, h)F_{kt}, \qquad (11)$$

where $\mu(\tau^*, h)$, $\phi(\tau^*, h)$, and $\beta_k(\tau^*, h)$ for $k = 1, \ldots, r$, are parameters, and F_t is the $r \times 1$ vector of the underlying unobserved factors at time t. In practice, the underlying factors in (11) are replaced by their estimations, $\hat{F}_t$, obtained as described above.

The parameters of the FA-QR model in (11) are estimated using the algorithm by Koenker and D'Orey (1987), which implements the quantile regression method originally developed by Koenker and Bassett (1978). When the error terms are assumed to be independently distributed according to a Laplace distribution, the estimator coincides with the Maximum Likelihood (ML) estimator; see Ando and Tsay (2011). Bai and Ng (2008) establishes its asymptotic normality.

The FA-QR provides estimates of the quantile function of the target variable, $\widehat{q}_{\tau^*}(y_{t+h}|y_t, F_t)$, for several values of τ^*. In practice, however, it is challenging to map these estimates into a probability distribution function due to approximation errors and estimation noise. Consequently, as in Adrian *et al.* (2019), we use the skew-t distribution proposed by Azzalini and Capitanio (2003) to smooth the quantile function and estimate the conditional density of y_t. The skew-t density depends on four parameters as follows:

$$f(y;\mu,\sigma,\alpha,v) = \frac{2}{\sigma} st\left(\frac{y-\mu}{\sigma}; v\right) sT\left(\alpha \frac{y-\mu}{\sigma}\sqrt{\frac{v+1}{v+\left(\frac{y-\mu}{\sigma}\right)^2}}; v+1\right), \tag{12}$$

where $st(\cdot)$ and $sT(\cdot)$ denote the probability density function and the cumulative distribution function of the Student's t distribution, respectively. The skew-t distribution is specified by its location μ, scale σ, shape α, and fatness v. At each time t, a skew-t distribution is fitted by choosing the parameters that minimize the squared differences between the quantile estimates and the skew-t implied quantiles, $q_{\tau^*}(y; \mu, \sigma, \alpha, v)$, as follows:

$$(\hat{\mu}_{t+h}, \hat{\sigma}_{t+h}, \hat{\alpha}_{t+h}, \hat{v}_{t+h}) = \underset{\mu,\sigma,\alpha,v}{\operatorname{argmin}} \sum_{t=1}^{T-h} (\widehat{q}_{\tau^*}(y_{t+h} \mid y_t, F_t) - q_{\tau^*}(y_t; \mu, \sigma, \alpha, v))^2. \tag{13}$$

The methodology described above estimates the conditional density of y_t under non-stressed conditions. To construct conditional densities based on stressed scenarios, González-Rivera *et al.* (2019) and González-Rivera *et al.* (2024) use the confidence ellipsoids defined in (10), and determine the value of the factors on the α%-contour (stress level of the underlying factors) that minimize (or maximize) a given quantile (τ) of the conditional distribution of the target variable. For instance, consider that we are interested in deriving a stress scenario for $\tau = 0.05$, with the factors stressed at their α% level, **FARS** solves the following optimization problem at each moment:

$$\min_{F_t^{(S)}} \hat{q}_{0.05}(y_{t+h}|y_t, F_t^{(S)}) \tag{14}$$

$$s.t. \quad g(F_t^{(S)}, \alpha) = 0,$$

where $g(F_t^{(S)}, \alpha) = 0$ is a predetermined α-contour of the factors, that is, an ellipsoid that contains F_t with probability α.

The values of $F_t^{(S)}$ on the boundary of the ellipsoid $g(F_t^{(S)}, \alpha) = 0$ represent extreme events of the factors. After solving the optimization problem in (14), these optimized values are plugged into the estimated FA-QRs. The conditional density of y_t under stress is then obtained by smoothing the corresponding quantiles as described in (13).[5]

[5]Note that the stressed scenarios are slightly different from that in González-Rivera *et al.* (2019) and González-Rivera *et al.* (2024), who obtain stressed factors for each quantile of the distribution.

3. The FARS package

In this section, we provide a detailed overview of the **FARS** package functionalities and explain how users can implement the methodology described in Section 2 using the available functions.

3.1. ML-DFM in FARS

We begin by introducing the `mldfm()` function, which provides users with a flexible tool for extracting factors using either DFM or ML-DFM, with either non-overlapping or overlapping blocks. In the case of a simple DFM, the function requires two input arguments. The first is `data`, which contains the N variables from which the factors are extracted, structured as a $T \times N$ matrix. The second argument is `global`, which specifies the number of factors r to be extracted from the data.

In the case of the ML-DFM without overlapping blocks, additional arguments must be provided to the `mldfm` function: i) the argument `blocks` defines the number of blocks K that make up the data sample (the default is 1, corresponding to the DFM case); ii) `block_ind` requires a vector that indicates the indices of the end column for each block k. For example, if $K = 3$ and $N = N_1 + N_2 + N_3$, the argument `block_ind` should contain $[N_1, N_1 + N_2, N_1 + N_2 + N_3]$; iii) the argument `local` is a vector of integers, indicating the number of block-specific factors r_{F_k} to be extracted from each block k; iv) `global` specifies the number of pervasive factors r_G; v) `method` defines the factor initialization strategy for the sequential LS estimation: 0 for the CCA (default) and 1 for PCA[6]; vi) the arguments `tol` and `max_iter` define the tolerance level and the maximum number of iterations allowed for the RSS minimization process, with default values set to 10^{-6} and 1000, respectively.

In the case of the ML-DFM with overlapping blocks, an additional `middle_layer` argument must be provided. `middle_layer` is a named list, where each name is a string specifying a group of overlapping blocks (e.g. kj in the case of pairwise groups), and each value is the number of factors r_{kj} to extract from that group. For example, if we want to extract one pairwise factor from blocks 1 and 3 ($r_{13} = 1$), the list should be defined as `list("1-3" = 1)`.

Regardless of the particular specification of the model, the `mldfm()` function returns an S3 object of class `mldfm` as output. The object is a list containing several attributes described in Table 1.

Attribute	Description
`Factors`	$T \times r$ matrix containing all the extracted factors.
`Lambda`	$N \times r$ matrix of factor loadings with necessary zero restrictions.
`Residuals`	$T \times N$ residual matrix from the model fit.
`Method`	The initialization strategy used (`CCA` or `PCA`).
`Iterations`	Number of iterations performed until convergence (0 in DFM).
`Factors_list`	A summary list indicating the number of factors extracted at each level.

Table 1: Attributes of the `mldfm` object. The r factors in the `Factors` and `Lambda` matrices follow the hierarchical order (from global to local) described in `Factors_list`.

The `mldfm` object has typical S3 methods: `print()`, `summary()` and `plot()`. The first two functions offer a brief overview of the model estimation outcome, while `plot()` offers pre-

[6] PCA is implemented using the `prcomp()` function from the package **stats**.

configured visualization tools. The call of the `plot` function on a `mldfm` object generates distinct line charts for all estimated factors, each enriched with confidence interval bands that assume cross-sectionally independent and homoskedastic idiosyncratic components. Furthermore, an optional input argument `dates` can be provided. `dates` is a vector of dates to be displayed on the x-axis, replacing the default integer time index ranging from 1 to T. Moreover, using the `plot()` function, it is possible to visualize estimated loadings or residuals, specifying a `which` argument with values `"loadings"` or `"residuals"`. With `"loadings"`, a singular figure is generated, which contains a set of bar charts displaying the estimated loadings along with their corresponding pairwise confidence intervals. Differently, with `"residuals"`, a figure depicting the correlation heatmap of the residuals is produced. In both cases, the user can provide a list of variable names using the optional `var_names` argument. This enables the replacement of the default indexes from VAR 1 to VAR N with the appropriate variable names.

3.2. Probability distribution of factors in FARS

A two-step procedure is implemented in **FARS** to obtain the asymptotic joint probability density of the factors with the subsampling correction.

The first step involves running a subsampling method to extract factors from subsets of N^* variables, selected from the entire data sample. This is implemented using the `mldfm_subsampling()` function. The function iteratively generates `n_samples` subsamples of size `sample_size` and estimates factors using the ML-DFM approach through the `mldfm()` function[7]. This approach offers two main advantages. First, the arguments of `mldfm_subsampling()` are the same as those of `mldfm()`, with the addition of two additional arguments to define the number and size of the subsamples. Second, the function returns a list of `mldfm` objects, enabling the user to apply standard methods such as `summary()`, `print()`, and `plot()` to each of the subsample results. In addition, an optional `seed` argument can be provided to ensure the reproducibility of the results.

The second step involves constructing confidence regions for the factors, as outlined in equation (10). This operation is performed by the `create_scenario()` function, which requires three main arguments. The first is `model`, which contains the result of the `mldfm()` function applied to the full dataset and serves as the center of the ellipsoid. The second is `subsample`, which takes the output of `mldfm_subsampling()`, a list of `mldfm` objects obtained from each subsample, and uses it to compute the MSE correction as defined in Equation (9). The third is `alpha`, which defines the coverage probability (i.e., the level of stress) for the ellipsoids. An optional argument, `atcsr`, can be set to `TRUE` to estimate the asymptotic MSE of the factors using $\widetilde{\Gamma}^{\text{FPR}}$ as defined in equation (8). Differently, the default setup (`FALSE`) uses $\hat{\Gamma}_t^{\text{BN}}$ as described in Equation (7). The output of `create_scenario()` is a list of T matrices of size $z \times r$ representing the ellipsoid points in r dimensions for each time observation t. The number of points z depends on the number of dimensions r. In the case of only one factor ($r = 1$), only a confidence interval is built based on the specified `alpha` level; for this reason, $z = 2$ (i.e., the upper and the lower bounds). In the case of two dimensions ($r = 2$), the 2-D ellipsoid is composed of $z = 300$ points and is built using the **ellipse** package; see

[7]The argument `n_samples` is the number of samples, while `sample_size` is the proportion of the cross-sectional dimension, N, that composes the subsamples (e.g., 0.9 to selected 90% of the original variables). In the case of multiple blocks, the proportion is maintained in all the blocks.

Murdoch and Chow (2023). Lastly, in the case of more than two dimensions ($r > 2$), the r-D ellipsoid is generated through the `hyperellipsoid()` and `hypercube_mesh()` functions from the **SyScSelection** package (Kopfmann 2023). In this case, the number of points composing the ellipsoid depends on the `phi` parameter of the `hypercube_mesh()` function, which defines the scalar fineness of the mesh. In **FARS**, `phi` is set to 8.

3.3. Conditional Density Under Stressed and Non-Stressed Conditions in FARS

In this section, we present the tools provided by **FARS** for obtaining conditional density forecasts in both the non-stressed and stressed scenarios.

The first step is to estimate the FA-QRs[8]. This operation is performed through the `compute_fars()` function, which estimates the parameter of the FA-QR in Equation (11). In the non-stressed setup, the function requires only three arguments to work. First, `dep_variable`, which contains the dependent variable y. Second, `factors`, which includes the factors the user wants to add to the quantile regression model.[9] Third, `h`, which defines the forecast horizon (the default is $h = 1$). The function estimates the FA-QRs for a fixed set of quantiles: 0.05, 0.25, 0.50, 0.75, and 0.95, as these are later used for the skew-t density fit. Alternatively, the user can modify the extreme quantiles by setting an optional `edge` argument. For example, setting `edge = 0.01` forces the edge quantiles to 0.01 and 0.99. The default value is 0.05. In the stressed scenario setup, additional arguments are required. The `scenario` argument takes the list of ellipsoids produced by the `create_scenario()` function. Moreover, the user must define `QTAU` and `min`, which correspond to the quantile that will be minimized or maximized, and the optimization strategy used to compute stressed factors over the ellipsoid points. The default value for `min` is `TRUE`, which means that the objective is to minimize a given quantile of the target variable y. Differently, if `min` value is `FALSE`, the objective is to maximize the quantile of y. The output of `compute_fars()` is an S3 object of type `fars`, which contains a set of attributes listed in Table 2.

Attribute	Description
`Quantiles`	$T \times 5$ matrix containing the estimated quantiles.
`Coeff`	$(r + 2) \times 5$ matrix containing the estimated coeffcients.
`StdError`	$(r + 2) \times 5$ matrix containing the estimated standard errors.
`Pvalue`	$(r + 2) \times 5$ matrix containing the estimated standard P-values.
`Levels`	The list of estimated quantiles.
`QTAU*`	The quantile selected for the min/max procedure.
`Stressed_Factors*`	$T \times r$ matrix containing the stressed factors.
`Stressed_Quantiles*`	$T \times 5$ matrix containing the estimated stressed quantiles.

Table 2: Attributes of the `fars` object. Attributes marked with * are included only if the user provides the necessary argument for the stressed scenario case.

Like the `mldfm` object, the `fars` object has standard S3 methods. The `print()` function

[8]FARS estimate FA-QRs using the **quantreg** package (Koenker, Portnoy, Ng, Zeileis, Grosjean, and Ripley 2025). The standard deviations of the estimated parameters are calculated using the sandwich formula proposed by Powell (1989) under the option `ker`, which is commonly used in practice.

[9]These can be easily accessed through the `Factors` attribute of the `mldfm` object obtained after estimating the ML-DFM by `mldfm()`.

provides a brief overview of the FA-QRs. The `summary()` function returns a detailed summary of quantile regression, including estimated coefficients, standard errors, and p-values for each quantile. Lastly, the `plot()` function generates two line charts: one composed of non-stressed quantiles and the second of stressed scenario quantiles. The function can display customized dates on the x-axis by setting the corresponding optional argument `dates`.

The second step to obtaining a density forecast is to estimate the conditional density of the target variable y by fitting a skewed-t distribution. This operation is performed via the `compute_density()` function, which requires a `quantiles` argument, containing the quantiles estimated by the `compute_fars()` function[10]. Depending on the quantiles provided, `Quantiles` or `Stressed_Quantiles`, the density function returns the non-stressed or the stressed conditional density, respectively. Additional arguments can be provided to `compute_density()`, including `est_points`, which set the number of estimation points (default is 512), `random_samples`, which define the number of random samples to be drawn from the estimated distribution (default is 5000) and `support`, which select the lower and upper bounds of the random variable support (default is `c(-10,10)`). For each period t, `compute_density()` initializes the skewed-t distribution by setting three parameters (location, scale, and shape) using the quantile values provided as input. The function implements two optimization procedures to fit the skew-t distribution. The default is a linear optimization using `optim()` from **stats**, which implements the `L-BFGS-B` method. The second is a non-linear optimization method that can be selected by setting the argument `nl = TRUE`. The non-linear method is from the **nloptr** package and is based on `NLOPT_LN_SBPLX` (Johnson (2007)). In both cases, the theoretical quantiles and the probability distribution function (pdf) of the fitted skewed-t distribution are computed using `qst()` and `dst()` from **sn** (Azzalini (2023)), respectively. Finally, a `seed` argument can be provided to ensure the reproducibility of the results. The `compute_density()` function returns a `fars_density` object which provides the attributes listed in Table 3.

Attribute	**Description**
`density`	The estimated densities at time t.
`distribution`	The random draws from the fitted skew-t distribution at each t.
`optimization`	The optimization method implemented: linear or non-linear.
`x_vals`	The sequence of evaluation points used to compute the density.

Table 3: Attributes of the `fars_density` object. Both `density` and `distribution` are provided in matrix form with one row for each time t.

The `fars_density` object is equipped with standard S3 methods. The `print()` function provides a brief overview of the estimated density. The `summary()` function returns the mean, median, and standard deviation of the distribution at time t. Finally, the `plot()` function generates a 3D plot of the density, with evaluation points (`x_vals`) on the x-axis, time indices on the y-axis, and density values on the z-axis. The function can also display custom dates on the y-axis by setting the optional argument `time_index`.

The final step in obtaining a conditional density forecast is to extract the conditional quantile from the estimated skew-t distribution. This can be performed using the function

[10] If the quantiles computed with `compute_fars()` have been modified via the `edge` argument, the density function must be informed of the correct quantiles levels. This can be done by setting the `levels` argument using the `$Levels` attribute of the `fars` object returned by `compute_fars()`.

`quantile_risk()`. This function requires two parameters: an object of class `fars_density` and the quantile that must be extracted `QTAU`. The quantile extraction is implemented via `quantile()` from **stats**. Depending on the `fars_density` object provided, either a non-stressed or a stressed density, the `quantile_risk()` extracts a non-stressed quantile or a stressed quantile of the target variable (e.g., in the case of GDP growth with `QTAU = 0.05` ($T = 59$), it extracts Growth-at-Risk or Growth-in-Stress).

Figure 2 shows a recap of the **FARS** package workflow for both the non-stressed and the stressed scenarios.

4. An application to conditional density of growth in the US economy

In this section, we illustrate the functionalities of the **FARS** package by building scenarios for US growth as in González-Rivera *et al.* (2024). In this exercise, we implement an ML-DFM using a data sample composed of three blocks. The first block contains $N_1 = 63$ global macroeconomic variables (GDP growth for 63 countries), the second block contains $N_2 = 248$ domestic macroeconomic variables, and the third block contains $N_3 = 208$ global financial variables, observed quarterly from 2005Q3 to 2020Q1. The dependent variable is the annualized quarterly GDP growth for the US. Data are retrieved from the replication files of González-Rivera *et al.* (2024).

The first step is to install and load the package. **FARS** is available to the public on CRAN under the GPL-3 license and can be downloaded as follows:

```
R> install.packages("FARS")
```

The development version is available on GitHub at `https://github.com/GPEBellocca/FARS`. This can be downloaded using the **devtools** package with the following command:

```
R>  devtools::install_github("GPEBellocca/FARS")
```

After installing the package from CRAN or GitHub, the package should be loaded as follows:

```
R> library(FARS)
```

The dataset, composed of $N = N_1 + N_2 + N_3 = 519$ variables, is stored in the `data` matrix, while the US GDP growth is stored in `dep_variable`. To estimate an ML-DFM through `mldfm()`, we first need to decide how many factors to extract from each block. Following González-Rivera *et al.* (2024), we extract a global factor from the three blocks, a pairwise factor from the global macroeconomic and global financial blocks (1 and 3), and one local factor from each of the three blocks. This operation is performed as follows:

```
R> mldfm_result <- mldfm(data,
+                        blocks = 3,
+                        block_ind = c(63,311,519),
+                        global = 1,
+                        local = c(1,1,1),
+                        middle_layer = list("1-3" = 1))
```

Since we are not providing any `method`, `tol`, and `max_iter`, the default values are enforced. The `mldfm` object returned is stored in the `mldfm_result` variable. After completion, the function `summary()` can be used to display an overview of the estimated ML-DFM, including the number of factors extracted at each level of the hierarchical structure used in the Sequential LS estimation.

```
R> summary(mldfm_result)

Summary of Multilevel Dynamic Factor Model (MLDFM)
===================================================
Number of periods:  59
Number of factors:  5
Number of nodes:  5
Initialization method:  CCA
Number of iterations to converge:  47

Number of factors per node:
 - 1-2-3 :  1 factor(s)
 - 1-3 :  1 factor(s)
 - 1 :  1 factor(s)
 - 2 :  1 factor(s)
 - 3 :  1 factor(s)

Residual sum of squares (RSS):  15215.6724
Average RSS per period:  257.8928
```

Additionally, using `plot()`, it is possible to obtain a graphical representation of the estimated factors, loadings, and residuals. For illustration, we extract two factors from the global macroeconomic block of variables as follows:

```
R> mldfm_result_gm <- mldfm(data = data[, 1:63], blocks = 1, global = 2)
```

Then, we call the plot function three times to plot the estimated factors, loadings, and residuals, in sequence. For a more precise result, we provide the plot function with appropriate arrays composed of dates and variable names using the optional arguments.

```
R> plot(mldfm_result_gm, dates = dates)
R> plot(mldfm_result_gm, which = "loadings", var_names = var_names)
R> plot(mldfm_result_gm, which = "residuals", var_names = var_names)
```

The results are plotted in Figures 3, 4 and 5, respectively. After estimating the ML-DFM we can now build the non-stressed and stressed scenarios following the steps depicted in Figure 2.

4.1. Non-stressed scenario

The first step to build the unstressed scenario is to estimate the FA-QRs as follow:[11]

[11] For this task, we consider the simplest case with `h=1`

```
R> fars_result <- compute_fars(dep_variable, mldfm_result$Factors, h = 1)

Running Factor-Augmented Quantile Regressions (FA-QRs)...
Completed
```

After this, we can plot the quantiles for the non-stressed scenario (see Figure 6, panel a) and print a recap of the FA-QRs.

```
R> plot(fars_result,dates=dates)
R> print(fars_result)

Factor-Augmented Quantile Regressions (FARS)
===========================================
Forecasted quantiles:
 - Number of periods:  59
 - Quantile levels:  0.05 0.25 0.50 0.75 0.95

Stressed quantiles: NO
```

The results stored in `fars_result` are then used to fit a skew-t distribution, generating the density for the non-stressed scenario based on `fars_result$Quantiles`. This is done by applying the default linear optimization method and providing an appropriate support for our GDP growth case.

```
R> ns_density <- compute_density(fars_result$Quantiles, support = c(-30,10), seed = 42)

Estimating skew-t densities from forecasted quantiles...
Completed
```

The generated `fars_density` object can be used to plot the non-stressed density (see Figure 7, panel a) and visualize an overview of the density estimation.

```
R> plot(ns_density, time_index = dates)
R> print(ns_density)

FARS Density
====================
Time observations  : 59
Estimation points  : 512
Random samples     : 5000
Support range      : [ -30 , 10 ]
Optimization       : Linear
```

Finally, we estimate the GaR at QTAU = 0.01 applying the `quantile_risk()` function to the non-stressed density.

```
R> GaR <- quantile_risk(ns_density, QTAU = 0.01)
```

4.2. Stressed scenario

As explained in Section 3, the computation of the stressed scenario can be performed in two steps. First, we need to obtain the asymptotic distribution of the factors. For this goal, we implement the subsampling procedure using the appropriate function. In our case, we generate 100 samples by extracting 94% of the variables from each block.

```
R> mldfm_ss_result <- mldfm_subsampling(data,
+                                       blocks = 3,
+                                       block_ind = c(63,311,519),
+                                       global = 1,
+                                       local = c(1,1,1),
+                                       middle_layer = list("1-3" = 1),
+                                       n_samples = 100,
+                                       sample_size = 0.94,
+                                       seed = 42)

Generating 100 subsamples...

Subsampling completed.
Number of subsamples generated: 100
```

Each of the 100 elements stored in `mldfm_ss_result` list can be manipulated as a distinct `mldfm` object. For example, we can visualize the summary of the ML-DFM estimated for sample number 10.

```
R> summary(mldfm_ss_result[[10]])

Summary of Multilevel Dynamic Factor Model (MLDFM)
===================================================
Number of periods:  59
Number of factors:  5
Number of nodes:  5
Initialization method:  CCA
Number of iterations to converge:  53

Number of factors per node:
 - 1-2-3 :  1 factor(s)
 - 1-3 :  1 factor(s)
 - 1 :  1 factor(s)
 - 2 :  1 factor(s)
 - 3 :  1 factor(s)

Residual sum of squares (RSS):  14212.7382
Average RSS per period:  240.8939
```

The second step to generate the stressed scenario is calling the `create_scenario()` function. For this exercise, we consider the highest stress level of `alpha = 0.99` and default $\hat{\Gamma}_t^{\mathrm{BN}}$.

```
R> scenario <- create_scenario(model = mldfm_result,
+                              subsample = mldfm_ss_result,
+                              alpha=0.99)
```

```
Constructing scenario using 100 subsamples and alpha = 0.99
Using standard time-varying Gamma
...
Scenario construction completed.
```

Now that we have both our ML-DFM in non-stressed conditions and our stressed scenario, respectively stored in `mldfm_result` and `scenario` variables, we can re-estimate the FA-QRs. Since we are interested in GDP growth risk, our objective is to minimize the dependent variable for the chosen quantile (`QTAU = 0.01`).

```
R> fars_result <- compute_fars(dep_variable,
+                              mldfm_result$Factors,
+                              scenario = scenario,
+                              h = 1,
+                              QTAU = 0.01)
```

```
Running Factor-Augmented Quantile Regressions (FA-QRs)...
Completed
```

The updated `fars` object stored in `fars_result` now contains both the non-stressed and stressed quantiles, which can be visualized by calling the `plot` function (see Figure 6).

```
R> plot(fars_result,dates=dates)
```

As with the non-stressed case, we fit a skew-t distribution using the `fars_result$Stressed_Quantiles` matrix to generate the stressed density, which we visualize with the `plot` function (see Figure 7, panel b).

```
R> s_density <- compute_density(fars_result$Stressed_Quantiles, seed = 42)
Estimating skew-t densities from forecasted quantiles...
Completed
```

The last step is to compute the GiS for `QTAU = 0.01` by feeding `quantile_risk()` with the stressed densities.

```
R> GiS <- quantile_risk(s_density, QTAU = 0.01)
```

In Figure 8, we plot the final GaR and GiS estimates along with the dependent variables. As in González-Rivera *et al.* (2024), we observe that GiS is more negative than GaR. This negative outcome would be neglected if we only estimated GaR, which assumes that factors evolve according to an average scenario.

To conclude our replication exercise, we have repeated the analysis at different levels of stress, `alpha`, and step-ahead, `h`. The results obtained for the out-of-sample forecast is reported in Table 4. Once again, we observe that the GaR estimates are more conservative than the GiS estimates across all time horizons.

	$h = 1$	$h = 2$	$h = 3$	$h = 4$
	2020Q2	2020Q3	2020Q4	2021Q1
Observed	-31.20	33.89	4.50	6.30
GaR	-24.90	-11.05	-1.19	2.28
GiS(70%)	-27.41	-12.41	-2.33	1.97
GiS(95%)	-28.28	-12.89	-2.73	1.87
GiS(99%)	-28.85	-13.20	-2.99	1.80

Table 4: US growth risk (in annualized percentage over previous quarter). The table reports h-step-ahead forecasts of the 1% quantile of growth with information up to 2020Q1 and computed by GaR (without stressing the underlying factors) and by GiS (with factors stressed at 70%, 95% and 99%).

5. Summary and discussion

The **FARS** package offers a suite of tools in R for modeling and designing economic scenarios based on conditional densities derived from multi-level dynamic factor models and factor-augmented quantile regressions. These tools allow researchers to generate both non-stressed and stressed scenarios for target variables, such as the US growth density (see, González-Rivera *et al.* 2024). The **FARS** package is available on the Comprehensive R Archive Network (CRAN) at https://CRAN.R-project.org/package=FARS, including the `data` matrix retrieved from the replication files of González-Rivera *et al.* (2024).

Acknowledgments

Financial support from the Spanish Government grant PID2022-139614NB-C22/AIE/10.13039/501100011033 (MINECO/FEDER) is gratefully acknowledged by all authors.

References

Adrian T, Boyarchenko N, Giannone D (2019). "Vulnerable growth." *American Economic Review*, **109**, 1263–1289.

Ando T, Tsay RS (2011). "Quantile regression models with factor-augmented predictors and information criterion." *Econometrics Journal*, **14**(1), 1–24.

Azzalini A, Capitanio A (2003). "Distributions generated by perturbation of symmetry with emphasis on a multivariate skew t-distribution." *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, **65**, 367–389.

Azzalini AA (2023). *The R package* **sn**: *The skew-normal and related distributions such as the skew-t and the SUN (version 2.1.1).* Università degli Studi di Padova, Italia. Home page: http://azzalini.stat.unipd.it/SN/, URL https://cran.r-project.org/package=sn.

Bai J (2003). "Inferential Theory for Factor Models of Large Dimensions." *Econometrica*, **71**(1), 135–171.

Bai J, Ng S (2006). "Confidence intervals for diffusion index forecasts and inference for factor-augmented regressions." *Econometrica*, **74**, 1133–1150.

Bai J, Ng S (2008). "Forecasting economic time series using targeted predictors." *Journal of Econometrics*, **146**, 304–317.

Bai J, Ng S (2013). "Principal components estimation and identification of static factors." *Journal of Econometrics*, **176**, 18–29.

Breitung J, Eickmeier S (2016). "Analyzing international business and financial cycles using multi-level factor models: A comparison of alternative approaches." In *Advances in Econometrics*, volume 35.

Choi I, Kim D, Kim YJ, Kwark NS (2018). "A multilevel factor model: Identification, asymptotic theory and applications." *Journal of Applied Econometrics*, **33**, 355–377.

Ergemen YE, Rodríguez-Caballero CV (2023). "Estimation of a dynamic multi-level factor model with possible long-range dependence." *International Journal of Forecasting*, **39**(1), 405–430.

Fresoli D, Poncela P, Ruiz E (2024). "Dealing with idiosyncratic cross-correlation when constructing confidence regions for PC factors." URL https://arxiv.org/pdf/2407.06883v1.

González-Rivera G, Maldonado J, Ruiz E (2019). "Growth in stress." *International Journal of Forecasting*, **35**, 948–966.

González-Rivera G, Rodríguez-Caballero CV, Ruiz E (2024). "Expecting the unexpected: Stressed scenarios for economic growth." *Journal of Applied Econometrics*, **39**, 926–942.

Hallin M, Liška R (2011). "Dynamic factors in the presence of blocks." *Journal of Econometrics*, **163**(1), 29–41.

Helske J (2017). "KFAS: Exponential family state space models in R." *Journal of Statistical Software*, **78**, 1–39.

Holmes EE, Ward EJ, Scheuerell MD, Wills K (2023). ***MARSS**: Multivariate Autoregressive State-Space Modeling*. R package version 3.11.9, URL https://CRAN.R-project.org/package=MARSS.

Johnson SG (2007). "The NLopt nonlinear-optimization package." https://github.com/stevengj/nlopt.

Koenker R, Bassett G (1978). "Regression Quantiles." *Econometrica*, **46**, 33.

Koenker R, Portnoy S, Ng PT, Zeileis A, Grosjean P, Ripley BD (2025). ***quantreg**: Quantile Regression*. R package version 6.1, URL https://CRAN.R-project.org/package=quantreg.

Koenker RW, D'Orey V (1987). "Computing Regression Quantiles." *Journal of the Royal Statistical Society Series C: Applied Statistics*, **36**, 383–393.

Kopfmann M (2023). ***SyScSelection**: Systematic Scenario Selection for Stress Testing*. R package version 1.0.2, URL https://CRAN.R-project.org/package=SyScSelection.

Krantz S, Bagdziunas R, Tikka S, Holmes E (2025). ***dfms**: Dynamic Factor Models*. R package version 0.3.0, URL https://CRAN.R-project.org/package=dfms.

Lin R, Shin Y (2023). ***GCCfactor**: GCC Estimation of the Multilevel Factor Model*. R package version 1.0.1, URL https://CRAN.R-project.org/package=GCCfactor.

Maldonado J, Ruiz E (2021). "Accurate Confidence Regions for Principal Components Factors*." *Oxford Bulletin of Economics and Statistics*, **83**, 1432–1453.

Mosley L, Chan TS, Gibberd A (2023). ***sparseDFM**: Estimate Dynamic Factor Models with Sparse Loadings*. R package version 1.0, URL https://CRAN.R-project.org/package=sparseDFM.

Mosley L, Chan TST, Gibberd A (2024). "The sparse dynamic factor model: a regularised quasi-maximum likelihood approach." *Statistics and Computing*, **34**, 1–19.

Murdoch D, Chow ED (2023). ***ellipse**:Functions for Drawing Ellipses and Ellipse-Like Confidence Regions*. R package version 0.5.0, URL https://CRAN.R-project.org/package=ellipse.

Powell J (1989). "Estimation of monotonic regression models under quantile restrictions." *In Barnett, W.A., J.L. Powell and G.E. Tauchen (eds.), Nonparametric and Semiparametric Methods in Econometric and Statistics: Proceedings of the Fith International Symposium in Economic Theory and Econometrics.*

Qiu Y, Liyanage J (2019). "Threshold selection for covariance estimation." *Biometrics*, **75**, 895–905.

Rodríguez-Caballero CV, Caporin M (2019). "A multilevel factor approach for the analysis of CDS commonality and risk contribution." *Journal of International Financial Markets, Institutions and Money*, **63**, 101144.

Stock JH, Watson MW (2002a). "Forecasting using principal components from a large number of predictors." *Journal of the American Statistical Association*, **97**, 1167–1179.

Stock JH, Watson MW (2002b). "Macroeconomic forecasting using diffusion indexes." *Journal of Business and Economic Statistics*, **20**(2), 147–162.

Affiliation:

Gian Pietro Bellocca, Ignacio Garrón, Esther Ruiz
Department of Statistics
Universidad Carlos III de Madrid
E-mail: gbellocc@est-econ.uc3m.es, igarron@est-econ.uc3m.es, ortega@est-econ.uc3m.es

C. Vladimir Rodríguez-Caballero
Department of Statistics
Instituto Tecnológico Autónomo de México
E-mail: vladimir.rodriguez@itam.mx

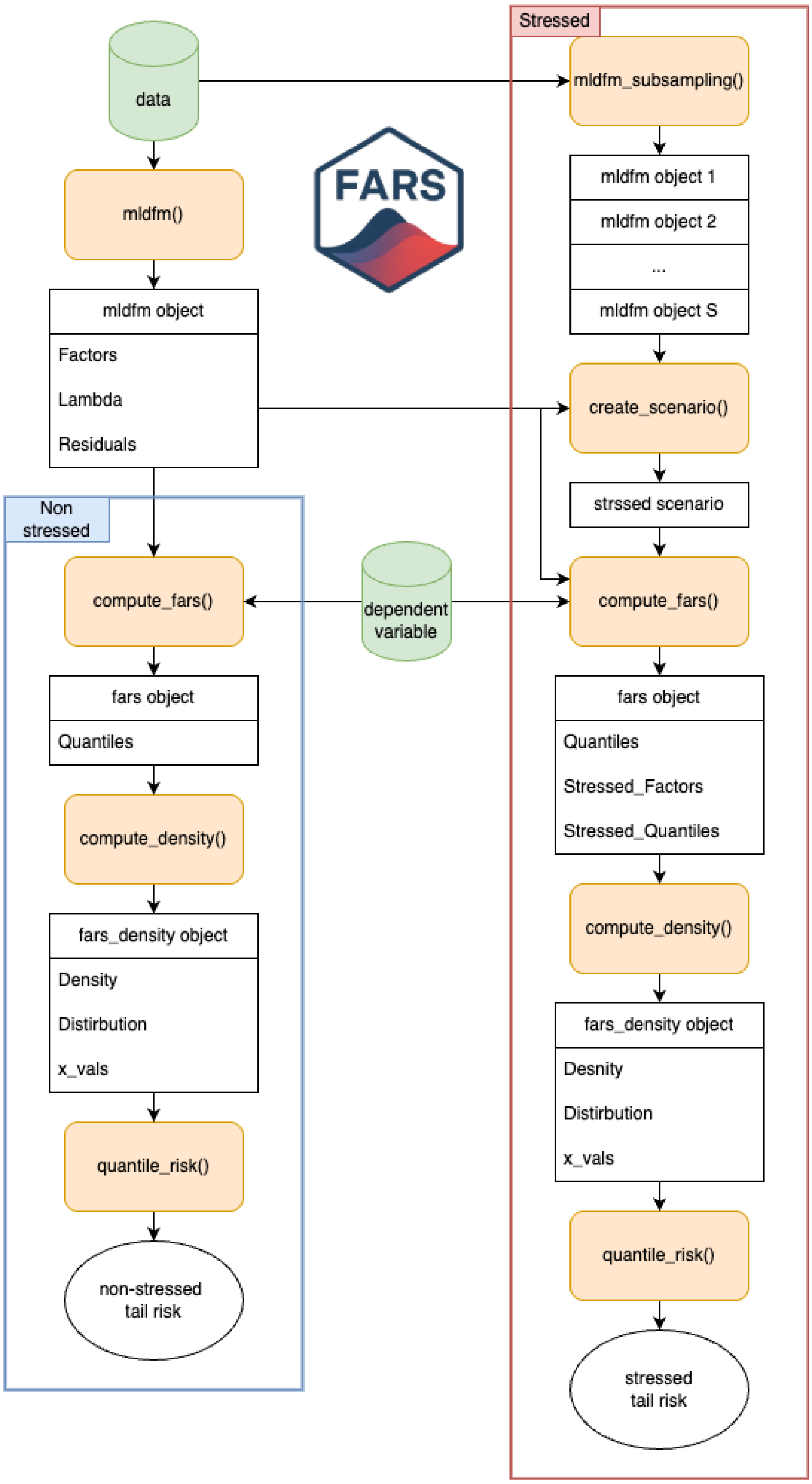
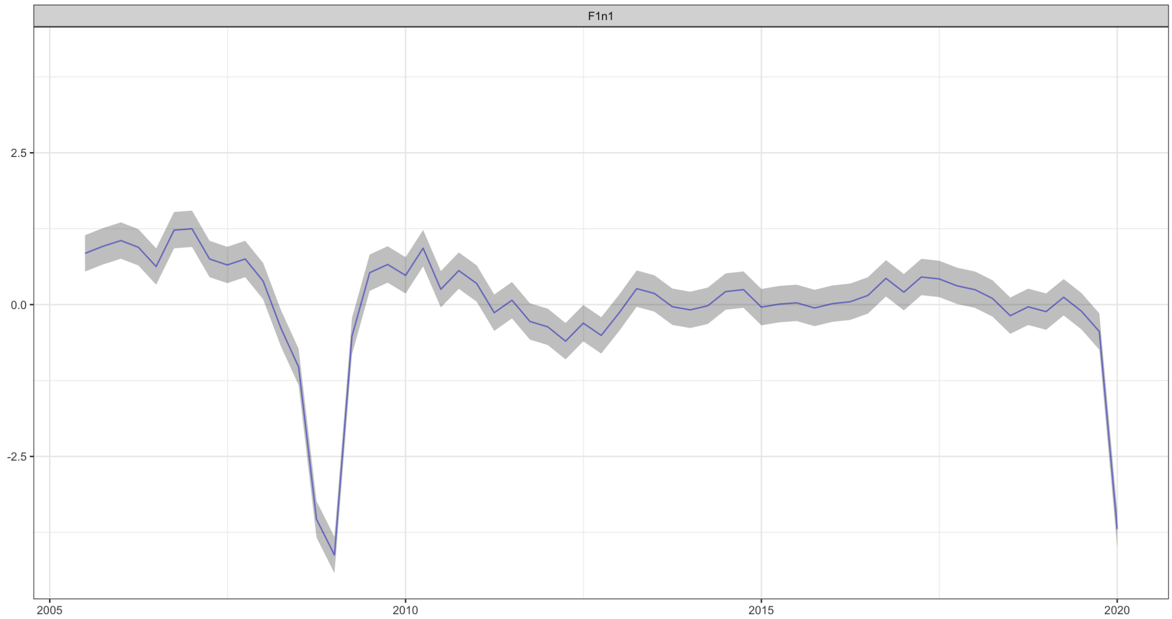
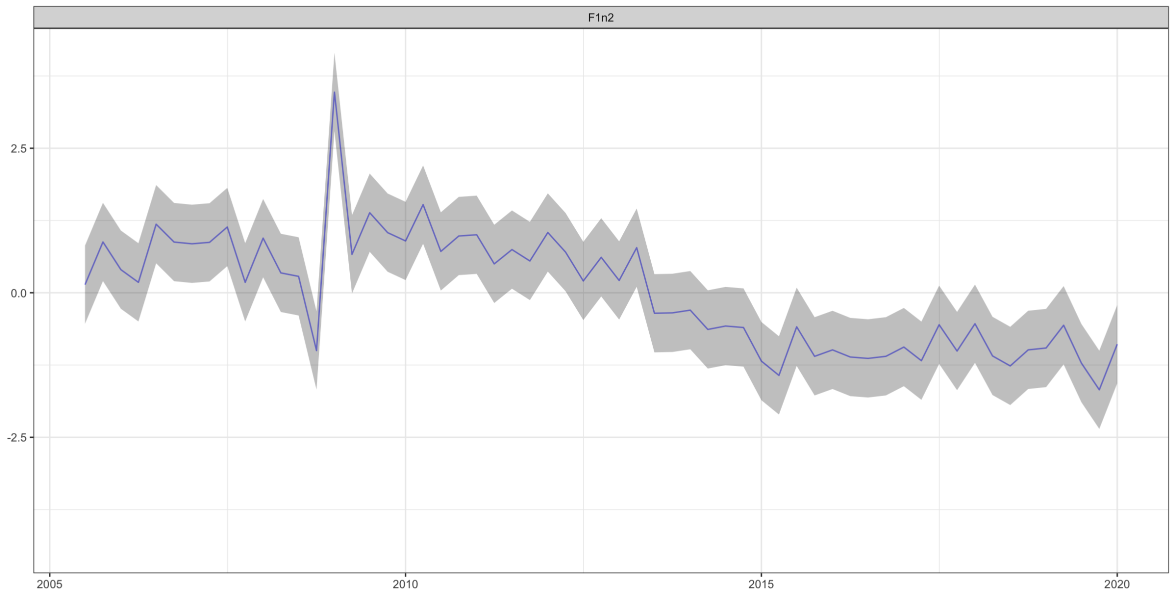


Figure 2: **FARS** package workflow for both non-stressed and stressed scenarios.

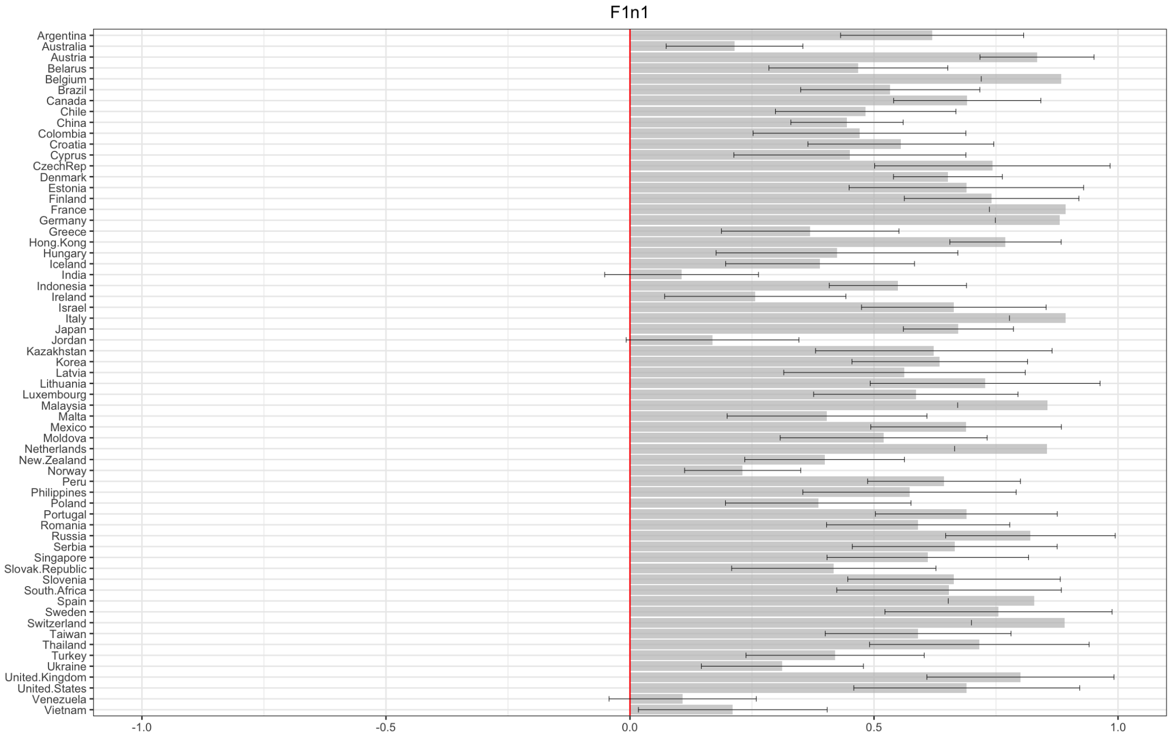


(a) Factor 1

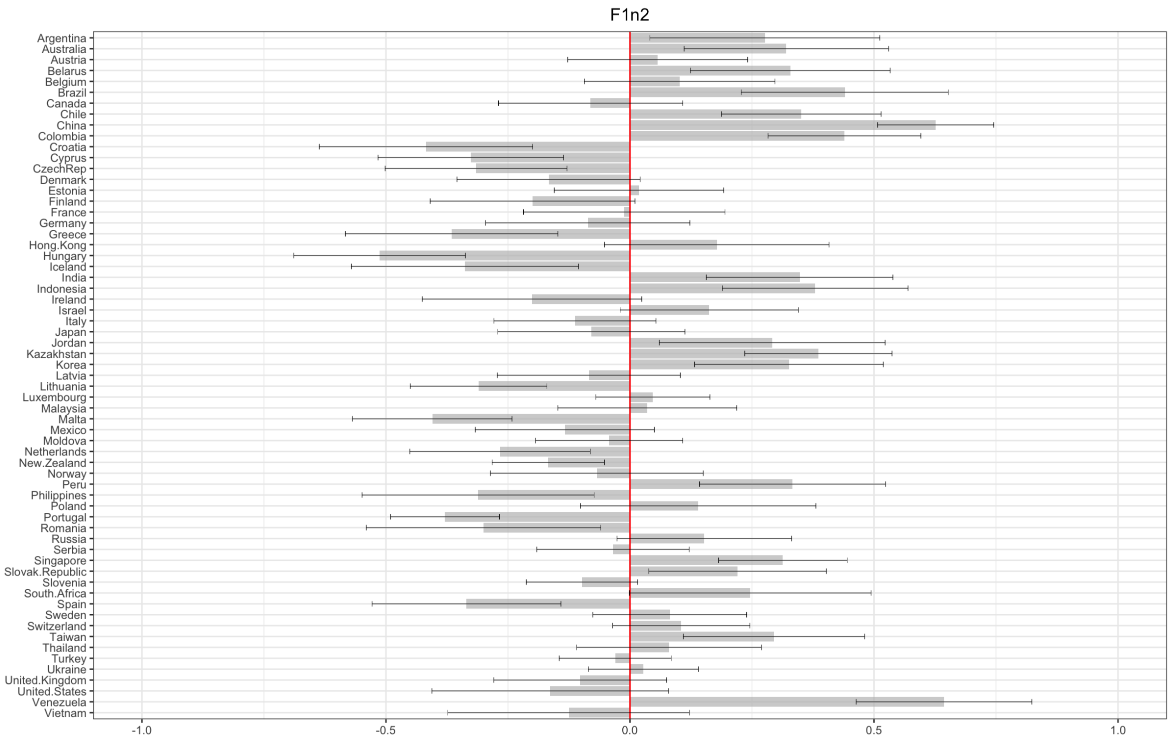


(b) Factor 2

Figure 3: Line chart of extracted factors.



(a) Loadings for factor 1



(b) Loadings for factor 2

Figure 4: Bar chart of factor loadings.

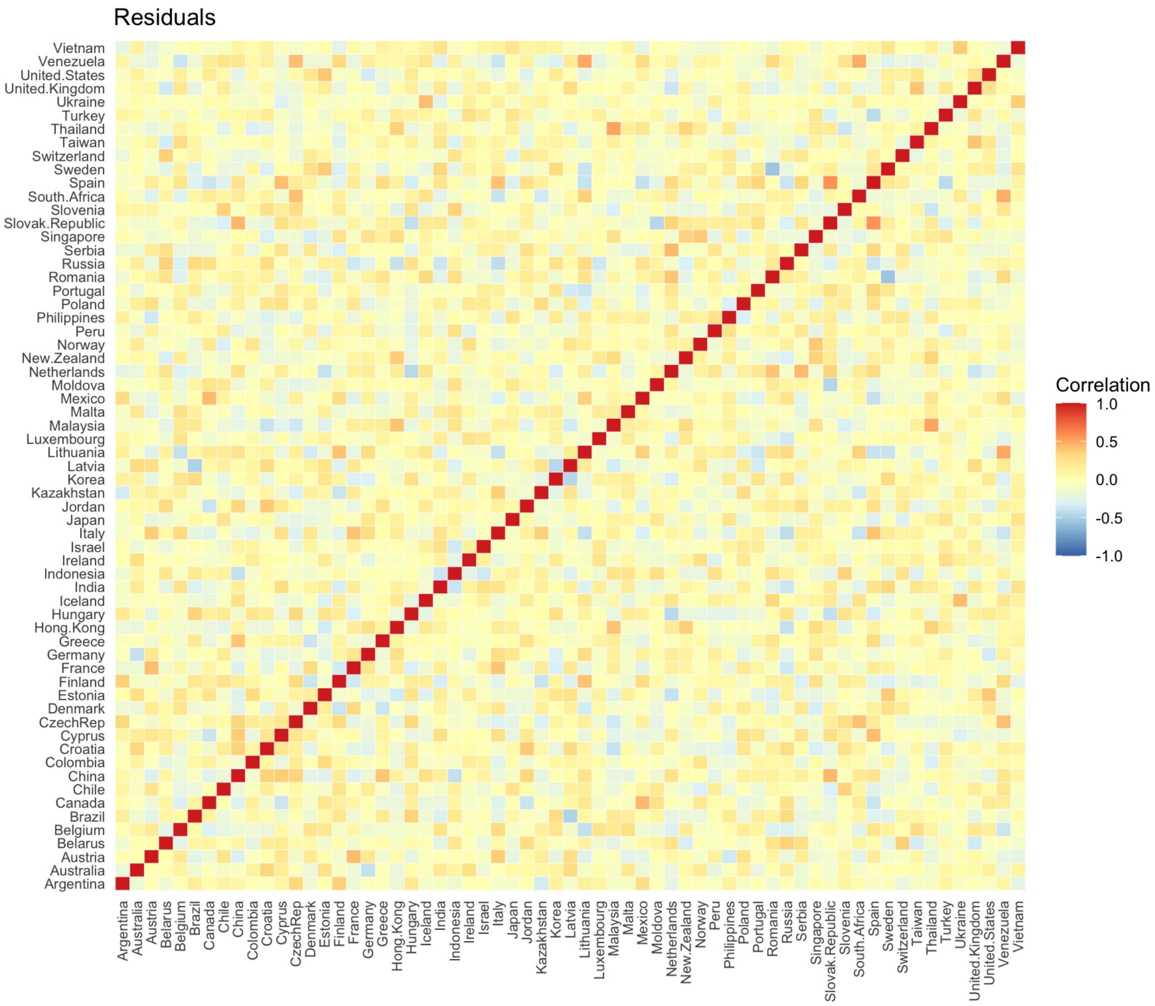
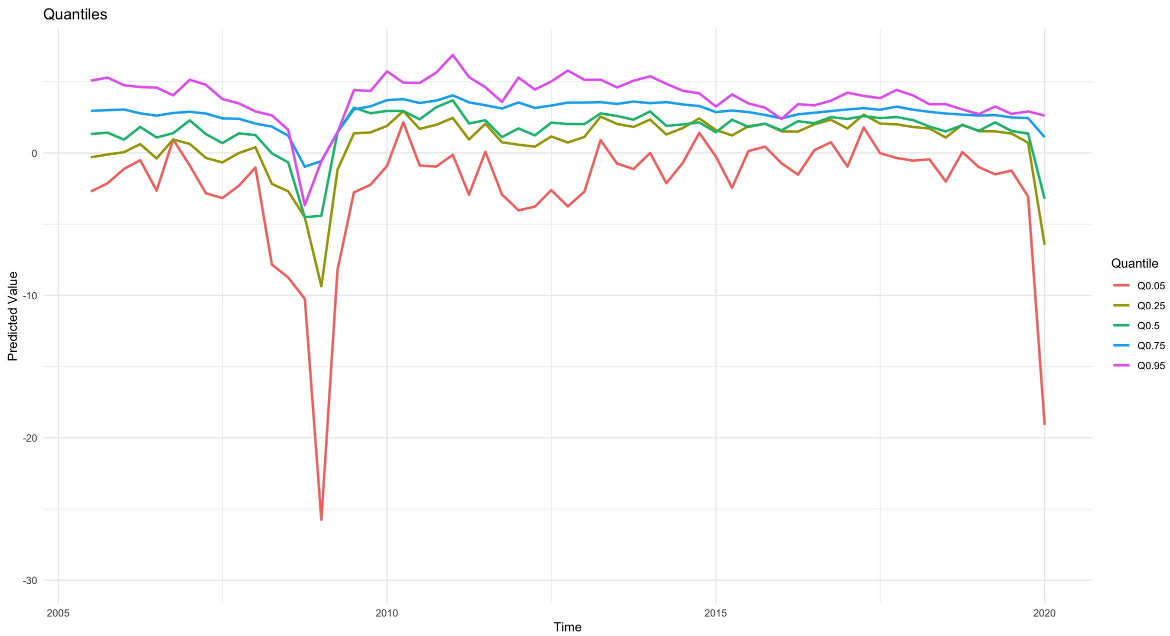
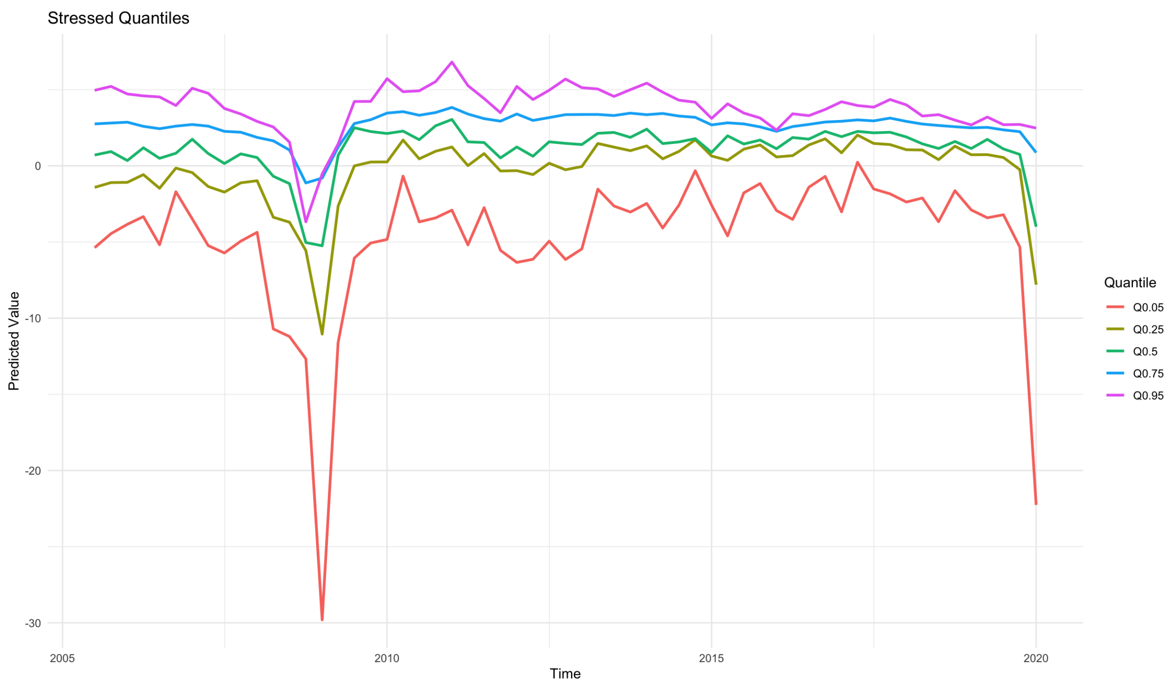


Figure 5: Residual correlation heatmap.

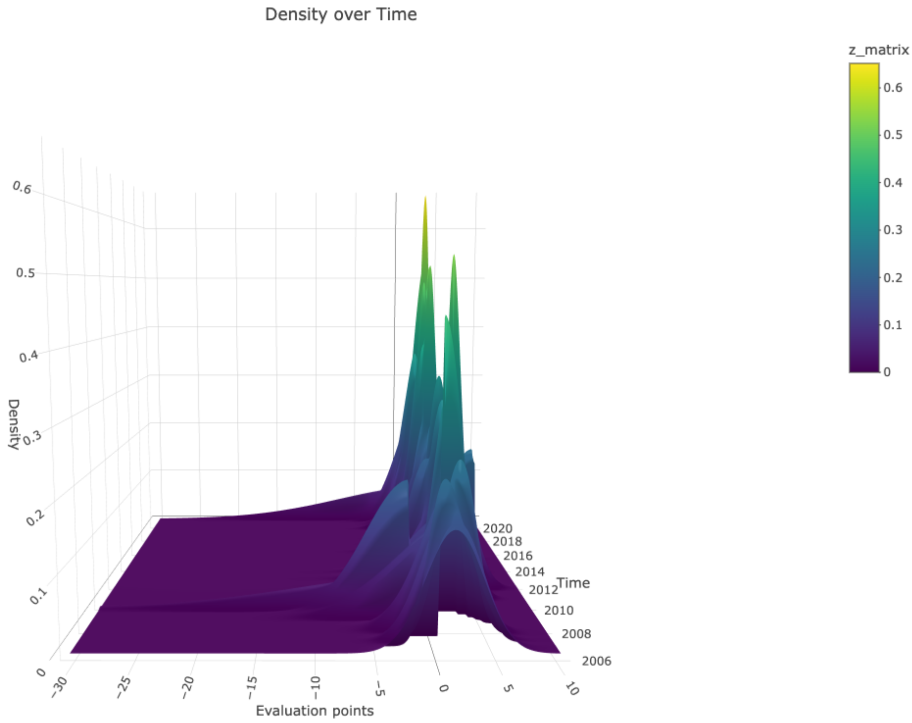


(a) Non-stressed scenario

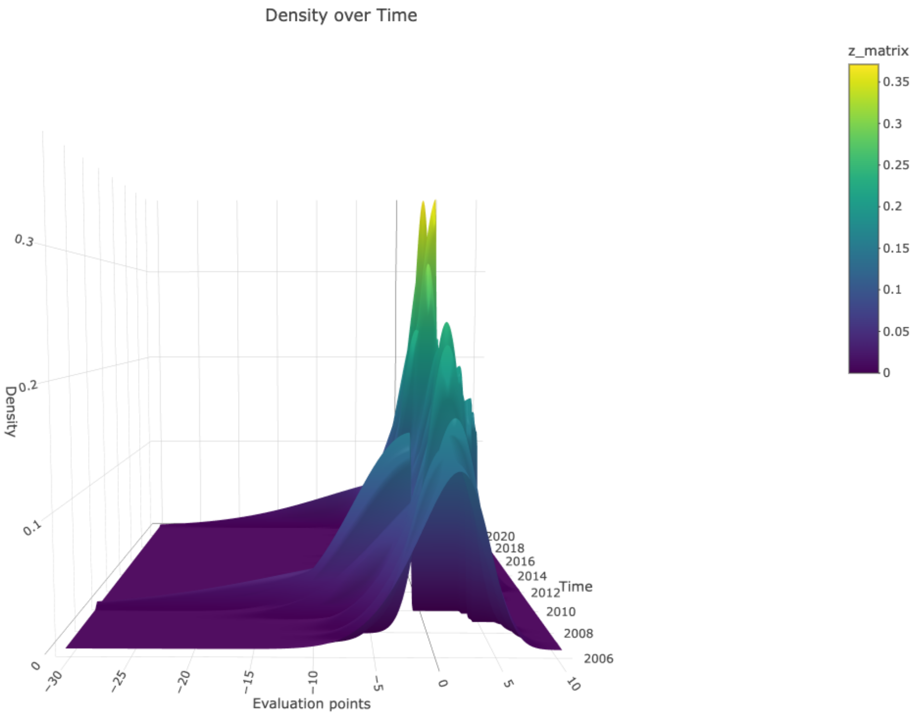


(b) Stressed scenario

Figure 6: Non-stressed and stressed scenario quantiles.



(a) Non-stressed density



(b) Stressed density

Figure 7: Non-stressed and stressed densities.

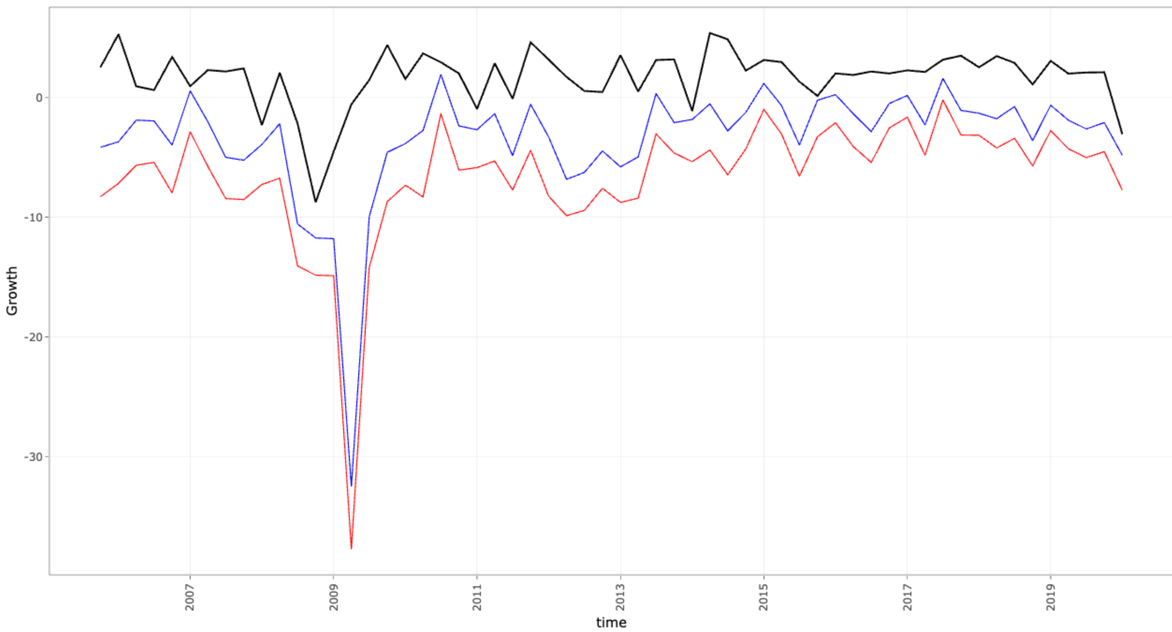


Figure 8: US quarterly growth: observed annualized rates in black, 1% GaR in blue and 1% GiS stressed with $\alpha = 99\%$ in red.