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

In the context of macroeconomic/financial time series, the **FARS** package provides a comprehensive framework in R for the construction of conditional densities of the variable of interest based on the factor-augmented quantile regressions (FA-QRs) methodology, with the factors extracted from multi-level dynamic factor models (ML-DFMs) with potential overlapping group-specific factors. Furthermore, the package also allows the construction of measures of risk as well as modeling and designing economic scenarios based on the conditional densities. In particular, the package enables users to: (i) extract global and group-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) obtain estimates of the parameters of the FA-QRs together with their standard deviations; (iv) recover full predictive conditional densities from estimated quantiles; (v) obtain risk measures based on extreme quantiles of the conditional densities; and (vi) estimate the conditional density and the corresponding extreme quantiles when the factors are stressed.

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

1. Introduction

In the context of macroeconomic/financial time series, there is a growing interest in the development of new econometric tools to obtain predictions of the probability densities of specific key variables; see, for example, Granger and Pesaran (2000a) and Granger and Pesaran (2000b), who argue that point forecasts are not sufficient from the perspective of a properly informed decision-maker. In addition to being of interest in themselves, these densities can also serve to obtain measures of macroeconomic vulnerability, which are crucial for the design of resilience policies; see, for example, Delle Monache, De Polis, and Petrella (2024). Furthermore, econometricians, policy makers, and financial analysts are also interested in the construction of realistic scenarios for the distribution of key variables that can help to further understand the resilience of economic systems by providing early warning signals of what to expect should such conditions materialize in adverse outlooks; see, for example, González-Rivera, Rodríguez-Caballero, and Ruiz (2024) and Adrian, Giannone, Lucciani, and West (2024).

To start with, estimation of the conditional density of interest is often based on assuming that underlying economic and/or financial latent factors drive it. As proposed by Bai and Ng (2008) and Ando and Tsay (2011), and popularized by 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 underlying latent factors, which summarize economic and/or financial activity, as regressors. The FA-QR model allows for different impacts of underlying factors on different quantiles of the distribution of the variable of interest, and consequently, for potential asymmetries in the downside and upside risks. After estimating the quantiles, and following Azzalini and Capitanio (2003), the corresponding h-step-ahead conditional density is obtained by fitting a skew-t distribution to them. The skew-t distribution has been shown to be flexible enough to provide an appropriate approximation to the conditional density of a large number of economic variables; see Mitchell, Poon, and Zhu (2024) for alternative estimators of densities, which are shown to outperform the popular skew-t distribution in the unlikely case of multimodal distributions. The estimated conditional density delivers any quantile of interest, and, in particular, extreme quantiles, which are often used as measures of vulnerability as, for example, the Growth at Risk (GaR) proposed by Adrian *et al.* (2019), or the Inflation at Risk (IaR) as in Lopez-Salido and Loria (2024).

The factors needed as regressors for FA-QRs can be extracted from a dynamic factor model (DFM), with the preferred estimation method being Principal Components (PC); see, for example, Bai (2003) and Bai and Ng (2013) for technical details.[1] Over the last few decades, when dealing with large systems of economic variables, it is not unusual to empirically observe that some of the latent factors, which summarize the common movements in the system, only load on particular groups of variables. This block structure may represent economic, geographical, cultural, or other characteristics. In this context, PC may face difficulties. Alternatively, factors can be extracted from Multi-level Dynamic Factor models (ML-DFMs) with the matrix of factor loadings subjected to the adequate blocks of zero restrictions. 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 group-specific factors associated with one or more blocks of variables. The ML-DFM can incorporate non-overlapping or overlapping blocks of variables. The factors of ML-DFMs can be extracted using the sequential Least Squares (LS) estimator proposed by Breitung and Eickmeier (2016) for non-overlapping factors and generalized by Rodríguez-Caballero and Caporin (2019) to overlapping factors. It is also important to note that, when the extracted factors are used as regressors of predictive regressions, obtaining measures of their uncertainty becomes relevant; see, for example, Amburgey and McCracken (2022) and Lewis, Mertens, Stock, and Trivedi (2022). The asymptotic distribution of the factors extracted by sequential LS is established by Choi, Kim, Kim, and Kwark (2018) for DFMs without overlapping factors and by Lu, Jin, and Su (2025) for overlapping factors.

Finally, in order to generate stressed scenarios (or stressed factors) for the conditional densities, the methodology proposed by González-Rivera, Maldonado, and Ruiz (2019) can be used. 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 on its extreme autocontours.

This paper presents the **FARS** package, which provides a comprehensive framework in R

[1]Note that on top of being used as predictors of FA-QRs, there are many other applications in which the factors can be of interest in themselves as, for example, when using them to construct economic/financial indexes or as predictors of diffusion indexes; see the survey on DFMs by Stock and Watson (2011).

for modeling and forecasting conditional densities based on ML-DFMs and FA-QRs.[2] The package enables users to:

1. Use sequential-LS to extract (pervasive, semipervasive, and block-specific) factors based on a flexible specification of the ML-DFM allowing for potential overlapping factors. To the best of our knowledge, there are no alternative published R packages that do this task. The only package available in R designed to extract factors in ML-DFMs with overlapping factors is **GCCfactor** by Lin and Shin (2023), which supports model selection and estimation in the context of a Generalized Canonical Correlation (GCC) estimator, which is closely related to sequential-LS; see Lin and Shin (2022) for a description of the GCC estimator.[3]

2. Compute asymptotically valid confidence regions for the factors extracted using sequential LS, accounting for uncertainty in the factor loadings, and for potential cross-correlations of the idiosyncratic components. As far as we are concerned, the only software in R that allows inference on factors is that mentioned above by Lin and Shin (2023). However, note that the confidence regions obtained in the latter package are based on bootstrap instead of being asymptotic.

3. Estimate the parameters of the FA-QR models and obtain their standard deviations. Recover full predictive conditional densities from these estimated quantiles. Obtain risk measures such as GaR and IaR. To our knowledge, only Lajaunie, Flament, Hurlin, and Kazemi (2025) provides unpublished software to estimate factor-augmented quantile regressions and the corresponding densities.[4]

4. Obtain scenarios for the conditional density and associated risk measures when the factors are stressed.

The functionalities of the **FARS** package are illustrated by extracting the underlying factors of headline inflation observed in a large number of countries in the euro area (EA). We also show how to use the extracted factors to estimate the conditional density of aggregate inflation in a given country and the corresponding risk of large inflation, both when the economy is under business-as-usual conditions and when it is under stress. A second illustration of the **FARS** package considers building scenarios for the density of economic growth in the United States (US), as in González-Rivera *et al.* (2024).

[2]Version 0.6.0 of the **FARS** package is available in CRAN: `https://CRAN.R-project.org/package=FARS`.

[3]Some alternative implementations of DFMs (but not multilevel) are available in the R programming language, although it is not published. 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 idiosyncratic components independently and identically distributed (*i.i.d.*) (Krantz, Bagdziunas, Tikka, and Holmes 2025). Also, there is commercial software that can be used to extract factors from DFMs; see, for example, Solberger and Spanger (2020) for the estimation of the DFM in the context of state-space models.

[4]There is available R software for quantile regressions (including linear, nonlinear, censored, locally polynomial and additive quantile regressions but not factor-augmented regressions) (Koenker 2025), or for factor-augmented regressions but without being regressions for the quantiles (Mevik and Wehrens 2007). Note that the former package has not been published.

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 with two empirical applications, namely, factor extraction and density estimation of aggregate inflation in the EA, and estimating (business-as-usual and stressed) conditional densities of economic growth in 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 extracting underlying factors and obtaining conditional density forecasts of the target variable under standard economic dynamics and stressed scenarios of the underlying factors. First, we discuss the factor structures involved in the DFMs and ML-DFMs (with and without overlapping blocks), and describe the asymptotic distribution of the PC estimated factors, assuming that idiosyncratic components are either cross-sectionally uncorrelated or weakly correlated. Second, we describe how to obtain forecasts of the density of 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 observed 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 weakly stationary latent factors, and $\epsilon_t = (\epsilon_{1t}, ..., \epsilon_{Nt})'$ is the $N \times 1$ vector of idiosyncratic components, which are assumed to be weakly stationary and cross-sectionally weakly correlated, and uncorrelated with the common factors F_t. Finally, the number of factors, r, is known.

In model (1), the loadings and factors cannot be separately identified. They can only be estimated consistently up to a rotation of the factor space. Consequently, the standard identification restrictions often assumed in the literature are 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, estimated factors are identified up to a sign transformation; see Bai and Ng (2013) for more details about the identification of DFMs in the context of PC estimation.

In practice, 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 highest eigenvalues of the matrix XX', ordered in decreasing magnitude. The corresponding loading matrix is then estimated by $\hat{P}' = \frac{1}{T}\hat{F}'X$.

2.2. Multi-level Dynamic Factor Model

In many economic/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 the matrix of loadings, 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

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

where, for $k = 1, ..., K$, $X_{k,t}$ is the $N_k \times 1$ vector of variables within block k, such that the cross-sectional dimension of $X_t = (X_{1,t}, ..., X_{K,t})'$ is $N = \sum_{k=1}^{K} N_k$. Furthermore, $G_t = (G_{1,t}, \dots, G_{r_G,t})'$ is the $r_G \times 1$ vector of pervasive factors that 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 that load only within the block $X_{k,t}$. The matrix of loadings and the idiosyncratic noise are defined conformably; see Breitung and Eickmeier (2016) and Choi *et al.* (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.[5] 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, 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 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 represents the relationships between pervasive, semipervasive, and block-specific factors, when $K = 3$.

The ML-DFM with overlapping blocks is given by

$$x_{k,it} = \mu'_{k,i} G_t + \kappa'_{kj_i} F_{kj,t} + \lambda'_{k,i} F_{k,t} + \epsilon_{k,it},$$

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

[5] 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.

vectors of 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 in (1).

The vector representation of the three-block ML-DFM with overlapping blocks is given by

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

Note that the total number of unobserved 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 of set theory to determine the number of pervasive, semipervasive and block-specific factors. However, the **FARS** package assumes that this number is known.

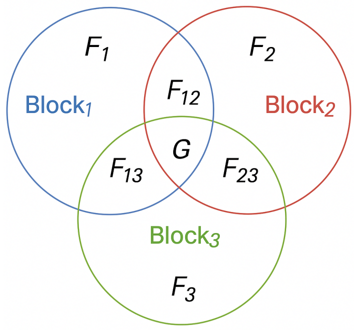


Figure 1: Factor structure of the ML-DFM with three different overlapping blocks of data.

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 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 block-specific factors $\hat{F}^{(0)} = \left(\hat{F}_{1,t}^{(0)}, \hat{F}_{2,t}^{(0)}, \ldots, \hat{F}_{k,t}^{(0)}\right)'$.

(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 unobserved 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.

The algorithm above does not impose any normalization. Henceforth, although the vector of common components $P^*F_t^*$ is consistently estimated, the factors and loading matrices are not identified separately. Consequently, Breitung and Eickmeier (2016) adapt the standard normalization in PC analysis to separately identify P^* and F_t^*. First, the different levels of estimated factors (pervasive, pairwise, and block-specific) are orthogonalized with respect to each other. A practical implementation consists of recursively regressing each factor on the previously ordered ones and using the residuals as updated orthogonalized estimates. For example, 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, $\hat{F}_t^*$, following a predetermined ordering.[6] Finally, the normalized pervasive factors are obtained as the main $\mathbf{r}_G$ components of the estimated common components. These are derived from the nonzero eigenvalues and the corresponding eigenvectors of the matrix $\widehat{M}\left(\frac{1}{T}\sum_{t=1}^{T}\widehat{G}_t\widehat{G}_t'\right)\widehat{M}'$, where $\widehat{M}$ represents the matrix of loadings corresponding to global factors. The same normalization procedure can be applied to the semipervasive and block-specific factors, using the sample covariance matrices of their respective common components.

[6]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. Although we do not estimate coefficients, the residuals obtained by regressing one factor level on another correspond to their orthogonal components, as in the FWL decomposition.

In step 1 of the algorithm, initialization of P^* and F_t^* is carried out using CCA. Alternatively, the **FARS** package provides the alternative of using PC. Although both approaches produce approximately the same estimated common components $\hat{P}^* \hat{F}_t^*$, the convergence of CCA is typically faster, requiring fewer iterations to minimize 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 comparison of the small sample properties of the sequential LS estimator initialized with CCA and PC for the two-level DFM.

2.3. Asymptotic distribution of factors

The construction of probabilistic scenarios for the unobserved factors requires knowledge of their joint distribution. The asymptotic distribution of PC factors obtained from the DFM in (1) is derived by Bai (2003). 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 $\Gamma_t = \lim_{N\to\infty} \sum_{i=1}^N \sum_{j=1}^N p_i p_j' E(\varepsilon_{it}\varepsilon_{jt})$ 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 relax this assumption allowing the idiosyncratic components to be weakly cross-sectionally correlated. Under these 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 $I(\cdot)$ 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.

Regardless of whether Γ_t is obtained from (7) or (8), the estimated asymptotic covariance matrix in (6) does not account for the uncertainty arising from the estimation of the loading matrix. 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:

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

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

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

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

The estimated factors, which summarize the information contained in a large set of predictors X_t, 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:

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

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^*. However, in practice, it is challenging to map these estimates into a probability distribution function because of 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 t

$$\begin{aligned} &\min_{F_t^{(S)}} \hat{q}_{0.05}(y_{t+h}|y_t, F_t^{(S)}) \\ &s.t. \quad g(F_t^{(S)},\alpha) = 0, \end{aligned} \tag{14}$$

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).[7]

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 DFM or ML-DFM, with non-overlapping or overlapping blocks. In

[7]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.

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 PC[8]; 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.
`loadings`	$N \times r$ 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 data stored in the `factors` and `loadings` 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-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. An additional optional argument, `flip`, can be supplied to improve the interpretability of the plots. `flip` is a binary vector (with values `0` or `1`) indicating whether each estimated factor or the corresponding loadings should be sign-flipped before plotting. A value of `1` for a given

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

element reverses its sign, while 0 leaves it unchanged. This is useful when the arbitrary sign indeterminacy of factor models leads to less interpretable visualizations. An optional argument, `fpr`, 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). 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. Specific attributes of the `mldfm` object can accessed using appropriate get functions, `get_factors()`, `get_loadings()` and `get_residuals()`.

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[9]. 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 an object `mldfm_subsample` containing a list of `mldfm` objects, enabling the user to apply standard methods to each of the subsample results. In addition, an optional `seed` argument can be provided to ensure the reproducibility of the results. A `mldfm_subsample` object contains the attributes listed in Table 2 and provides `print()`, `summary()` and `plot()` methods, as well as `get_mldfm_list()` and `get_mldfm_model()` functions to access the entire list or a specific `mldfm` object, respectively.

Attribute	Description
`models`	A list containing the `n_samples` `mldfm` objects.
`n_samples`	The number of subsamples generated.
`sample_size`	The proportion of the sample used for each subsample $\frac{N^*}{N}$.
`seed`	The seed used for random sampling.

Table 2: Attributes of the `mldfm_subsample` object.

The second step involves constructing confidence regions for the factors, as described 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 `subsamples`, which uses the output of `mldfm_subsampling()` to compute the MSE correction as defined

[9]The argument `n_samples` is the number of samples, while `sample_size` is the proportion of the cross-sectional dimension, N, which 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.

in Equation (9). The third is `alpha`, which defines the coverage probability (i.e., the level of stress) of the ellipsoids. An optional argument, `fpr`, 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 `fars_scenario` object whose attributes are presented in Table 3. A `fars_scenario` object is provided with the standard S3 methos (`print()`, `summary()` and `plot()`) and with `get_ellipsoids()` and `get_sigma_list()` functions to access specific attributes. In particular, `get_ellipsoids()` returns a list of T matrices of size $z \times r$ representing the ellipsoid points in r dimensions at each time 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 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.

Attribute	**Description**
`ellipsoids`	A list containing T matrices of dimensions $r \times z$.
`center`	$T \times r$ matrix containing all the factors used as center coordinates for the ellipsoids.
`sigma`	A list of T covariance matrices of dimensions $r \times r$.
`periods`	Number of time periods T.
`n_points`	Number of points z used to define each ellipsoid.
`alpha`	Confidence level for the ellipsoids.

Table 3: Attributes of the `fars_scenario` object.

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

In this subsection, 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[10]. 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_t. Second, `factors`, which includes the factors the user wants to add to the quantile regression model.[11] 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

[10]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.

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

`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 `ellipsoids` argument takes the list of ellipsoids from a scenario 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_t. Differently, if `min` value is `FALSE`, the objective is to maximize the quantile of y_t. The output of `compute_fars()` is an S3 object of type `fars`, which contains a set of attributes listed in Table 4.

Attribute	**Description**
`quantiles`	$T \times 5$ matrix containing the estimated quantiles.
`coeff`	$(r+2) \times 5$ matrix containing the estimated coeffcients.
`std_error`	$(r+2) \times 5$ 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 4: 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 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`. In order to access the attributes of a `fars` object, a set of getter functions is available. The function `get_quantiles()` returns either stressed or non-stressed factors, depending on whether the parameter `stressed` is set to `TRUE` or `FALSE` (default). The functions `get_stressed_factors()` and `get_quantile_levels()` return the stressed factors and the list of estimated quantile levels, respectively.

The second step to obtaining a density forecast is to estimate the conditional density of the target variable y_t 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[12]. 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,

[12]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()`.

`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 that provides the attributes listed in Table 5.

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.
`eval_points`	The sequence of evaluation points used to compute the density.

Table 5: 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 each time t. Finally, the `plot()` function generates a 3D plot of the density, with evaluation points (`eval_points`) 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 distribution is accessible through the `get_distribution()` function.

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 `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. Illustration of FARS package functionalities

In this section, we illustrate the functionalities of the **FARS** package by extracting factors, estimating conditional densities and obtaining stress scenarios in the context of: i) aggregate inflation in Europe; and ii) building scenarios for US growth density. Regardless of the particular application, the first step is to install and load the package **FARS**, which is available publicly on CRAN under the GPL-3 license, 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, it should be loaded as follows:

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

4.1. European inflation: Risk in extreme right quantiles.

In the first illustration, we analyze the risk of an inflation increase in Europe. To do this, we collect monthly headline CPI data (Ha, Kose, and Ohnsorge 2023) from January 2005 to December 2024 ($T = 239$) for a set of $N = 38$ European countries. The countries considered are divided into three different blocks, depending on geographical location:

- **West** ($N_1 = 11$)**:** Austria, Belgium, France, Germany, Ireland, Italy, Luxembourg, Portugal, Spain, Switzerland, United Kingdom.
- **East** ($N_2 = 21$)**:** Belarus, Bosnia and Herzegovina, Bulgaria, Croatia, Cyprus, Czech Republic, Estonia, Greece, Hungary, Kosovo, Latvia, Lithuania, Malta, Moldova, Rep., North Macedonia, Poland, Romania, Slovakia, Slovenia, Turkey, Ukraine.
- **North** ($N_3 = 6$)**:** Denmark, Finland, Iceland, The Netherlands, Norway, Sweden.

For each country, CPI prices are transformed into annualized month-on-month (mom) inflation, with each inflation series sequentially cleaned of seasonal effects and outliers. The processed data can be imported using:

```
R> data("inflation_data", package = "FARS")
```

To estimate a ML-DFM through `mldfm()`, we first need to decide how many factors to extract from each block. We extract one global factor common to all N countries, and one block-specific factor common to countries in each of the three blocks. This operation is performed as follows:

```
R> mldfm_result <- mldfm(inflation_data,
+                         blocks = 3,
+                         block_ind = c(11,32,38),
+                         global = 1,
+                         local = c(1,1,1))
```

Since we do not provide 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             : 239
Number of factors             : 4
Number of nodes               : 4
Initialization method         : CCA
Number of iterations to converge: 33

Factor structure:
 - 1-2-3 :  1 factor(s)
 - 1 :  1 factor(s)
 - 2 :  1 factor(s)
 - 3 :  1 factor(s)

Residual diagnostics:
 - Total residual sum of squares (RSS):  4506.23
 - Average RSS per time period       :  18.85
```

Additionally, using `plot()`, it is possible to obtain a graphical representation of the estimated factors, loadings, and residuals. This is performed by calling the plot function three times in sequence. For a more precise result, we provide the plot function with appropriate arrays composed of dates and country names using the optional arguments. Also, we specify that the global factor and the local factors corresponding to blocks 1 and 2 must be flipped in sign.

```
R> plot(mldfm_result, dates = dates, flip = c(1,1,1,0))
R> plot(mldfm_result, which = "loadings", var_names = countries, flip = c(1,1,1,0))
R> plot(mldfm_result_gm, which = "residuals", var_names = countries)
```

The results are plotted in Figures 3, 4 and 5, respectively.

Non-stressed scenario

In order to analyze potential inflation risk in Europe we utilize Germany as an example. To do this, we extract the corresponding inflation series from the data set.

```
R> dep_variable <- as.numeric(inflation_data[[4]])
```

The first step to build the unstressed scenario is to estimate the FA-QRs as follows:[13]

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

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

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

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:  239
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. This is done by applying the non-linear optimization method and providing an appropriate support for the inflation case.

```
R> ns_density <- compute_density(get_quantiles(fars_result),
+                                support = c(-30,30),
+                                seed = 42,
+                                nl=TRUE)

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  : 239
Estimation points  : 512
Random samples     : 5000
Support range      : [ -30 , 30 ]
Optimization       : Non-linear
```

Finally, we estimate the Inflation at Risk (IaR) at qtau = 0.99 applying the `quantile_risk()` function to the non-stressed density.

```
R> IaR <- quantile_risk(ns_density, qtau = 0.99)
```

Stressed scenarios.

As explained in Section 3, the computation of stressed scenarios 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 95% of the countries in each block.

```
R> mldfm_ss_result <- mldfm_subsampling(inflation_data,
+                                        blocks = 3,
+                                        block_ind = c(11,32,38),
+                                        global = 1,
+                                        local = c(1,1,1),
+                                        n_samples = 100,
+                                        sample_size = 0.95,
+                                        seed = 42)

Generating 100 subsamples...
Subsampling completed.
```

Each of the 100 models stored in `mldfm_ss_result` 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(get_mldfm_model(mldfm_ss_result, index = 10))

Summary of Multilevel Dynamic Factor Model (MLDFM)
===================================================
Number of periods              : 239
Number of factors              : 4
Number of nodes                : 4
Initialization method          : CCA
Number of iterations to converge: 23

Factor structure:
 - 1-2-3 :  1 factor(s)
 - 1 :  1 factor(s)
 - 2 :  1 factor(s)
 - 3 :  1 factor(s)

Residual diagnostics:
 - Total residual sum of squares (RSS):  4226.02
 - Average RSS per time period        :  17.68
```

The second step is to generate the stressed scenario by 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, alpha = 0.99
and standard time-varying Gamma...
Scenario construction completed.
```

A summary of the scenario can be displayed as follows:

```
R> summary(scenario)

FARS Scenario Summary
======================
Number of periods    : 239
Ellipsoid dimensions : 4
Points per ellipsoid : 1072
Confidence level     : 99 %
FPR Gamma            : FALSE

Center (factor estimates):
  Mean     : 0
  Std. Dev : 0.9984
  Min      : -5.6607
  Max      : 4.3215

Ellipsoid variability (diagonal of Sigma):
  Mean     : 0.3317
  Std. Dev : 0.5728
  Min      : 0.0079
  Max      : 7.4062
```

Now that we have the ML-DFM under non-stressed conditions and the stressed scenario stored in `mldfm_result` and `scenario` variables, respectively, we can re-estimate the FA-QRs. Since we are interested in Inflation risk, our objective is to maximize the dependent variable for the chosen quantile (`qtau = 0.99`).

```
R> fars_result <- compute_fars(dep_variable,
+                              get_factors(mldfm_result),
+                              ellipsoids = get_ellipsoids(scenario),
+                              h = 1,
+                              qtau = 0.99,
+                              min = FALSE)

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

The updated `fars` object stored in `fars_result` now contains 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 in the non-stressed case, we fit a skew-t distribution using the `fars_result`. This time we need to provide the stressed quantiles matrix to generate the stressed density. Again, we visualize the density with the `plot` function (see Figure 7, panel b).

```
R> s_density <- compute_density(get_quantiles(fars_result, stressed = TRUE),
+                               support = c(-30,30),
+                               seed = 42,
+                               nl=TRUE)

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

The last step is to compute the Inflation in Stress (IiS) for `qtau = 0.99` by feeding `quantile_risk()` with the stressed densities.

```
R> IiS <- quantile_risk(s_density, qtau = 0.99)
```

In Figure 8, we plot the final IaR and IiS estimates along with the dependent variables for the period spanning from March 2005 to December 2024. We observe that IiS is higher than IaR. This worse outcome would be neglected if we only estimated IaR, which assumes that factors evolve according to an average scenario.

4.2. Economic growth in the US: Risk in extreme left quantiles.

In our second illustration, we follow González-Rivera *et al.* (2024) and construct densities for annualized quarterly GDP growth in the US with the underlying factors extracted in the context of a ML-DFM using a data sample composed of three blocks. The first block contains $N_1 = 63$ international macroeconomic variables (GDP growth for 63 countries), the second block contains $N_2 = 248$ domestic macroeconomic variables, and the third block contains $N_3 = 208$ international financial variables. All variables are observed quarterly from 2005Q3 to 2020Q1.[14] The dataset, composed of $N = N_1 + N_2 + N_3 = 519$ variables and the US GDP growth can be imported using:

```
R> data("mf_data", package = "FARS")
R> data("dep_variable", package = "FARS")
```

We extract one global factor common to all N variables, a pairwise factor common to all international variables (international macroeconomic and international financial blocks), and one block-specific factor common to the variables in each of the three blocks. Then, we check the summary of the model.

[14]Data are retrieved from the replication files of González-Rivera *et al.* (2024).

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

Factor structure:
 - 1-2-3 :  1 factor(s)
 - 1-3 :  1 factor(s)
 - 1 :  1 factor(s)
 - 2 :  1 factor(s)
 - 3 :  1 factor(s)

Residual diagnostics:
 - Total residual sum of squares (RSS):  15215.67
 - Average RSS per time period        :  257.89
```

To build the stressed scenario we implement the same two-step procedure using the `mldfm_subsampling()` and `create_scenario()` functions. As for the inflation case, we generate 100 samples by extracting 95% of the variables from each block and consider the highest stress level of `alpha = 0.99` with default $\hat{\Gamma}_t^{\mathrm{BN}}$.

```
R> mldfm_ss_result <- mldfm_subsampling(mf_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 = 0.95,
+                                       seed = 42)

Generating 100 subsamples...
Subsampling completed.

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

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

Regarding the FA-QRs, since we are interested in GDP growth risk, our objective is to minimize the dependent variable for the chosen low quantile (`qtau = 0.01`).

```
R> fars_result <- compute_fars(dep_variable,
+                              get_factors(mldfm_result),
+                              ellipsoids = get_ellipsoids(scenario),
+                              h = 1,
+                              qtau = 0.01)
```

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

The output stored in `fars_result` contains both the non-stressed and stressed quantiles, which can be visualized, with appropriate dates, by calling the `plot` function (see Figure 9).

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

Now that we have the quantiles and the stressed quantiles, so that we can fit two skew-t distributions to generate the non-stressed and the stressed densities, which we visualize with the `plot` function (see Figure 10, panels a) and b). For this exercise, we implement the linear optimization method.

```
R> ns_density <- compute_density(get_quantiles(fars_result),
+                                support = c(-30,10),
+                                seed = 42)
R> s_density <- compute_density(get_quantiles(fars_result, stressed = TRUE),
+                               support = c(-30,10),
+                               seed = 42)
```

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

The final step is to compute GaR and GiS for `qtau = 0.01` by the feeding `quantile_risk()` function with the appropriate densities.

```
R> GaR <- quantile_risk(ns_density, qtau = 0.01)
R> GiS <- quantile_risk(s_density, qtau = 0.01)
```

In Figure 11, we plot the in-sample 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.

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 ML-DFMs and FA-QRs. 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.

Adrian T, Giannone D, Lucciani M, West M (2024). "Scenario synthesis and macroeconomic risk." *arXiv:2505.05193v1[econ.EN].*

Amburgey A, McCracken M (2022). "On the real-time predictive content of financial condition indices for growth." *Journal of Applied Econometrics*, **38**, 137–163.

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.

Delle Monache D, De Polis A, Petrella I (2024). "Modeling and forecasting macroeconomic downside risk." *Journal of Business & Economic Statistics*, **42**(3), 1010–1025.

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.

Granger C, Pesaran M (2000a). *Decision Theoretic Approach to Forecast Evaluation*. World Scientific.

Granger C, Pesaran M (2000b). "Economic and statistical measures of forecast accuracy." *Journal of Forecasting*, **19**, 537–560.

Ha J, Kose MA, Ohnsorge F (2023). "One-stop source: A global database of inflation." *Journal of International Money and Finance*, **137**, 102896.

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 (2025). *quantreg: Quantile Regression*. R package version 6.1, URL https://CRAN.R-project.org/package=quantreg.

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.

Lajaunie Q, Flament G, Hurlin C, Kazemi S (2025). *at Risk*. R package version 0.2.0, URL https://CRAN.R-project.org/package=atRisk.

Lewis D, Mertens K, Stock J, Trivedi M (2022). "Measuring real activity using a weekly economic index." *Journal of Applied Econometrics*, **37**, 667–687.

Lin R, Shin Y (2022). "Generalised canonical correlation estimation of the multilevel factor model." *SSRN Electronic Journal*. doi:10.2139/ssrn.4295429.

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.

Lopez-Salido D, Loria F (2024). "Inflation at risk." *Journal of Monetary Economics*, **145, Supplement**, 103570.

Lu X, Jin S, Su L (2025). "Three-dimensional factor models with global and local factors." *Econometric Theory*.

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

Mevik BH, Wehrens R (2007). "The PLS package: Principal Components and partial least sqaures regression in R." *Journal of Statistical Software*, **18**, 1–23.

Mitchell J, Poon A, Zhu D (2024). "Constructing density forecasts from quantile regressions: Multimodality in macrofinancial dynamics." *Journal of Applied Econometrics*, **39**, 790–812.

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.

Solberger M, Spanger E (2020). "Estimating a dynamic factor model in EWiews using the Kalman filter and smoother." *Computational Economics*, **55**, 875–900.

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.

Stock JH, Watson MW (2011). "Dynamic factor models." *The Oxford Handbook of Economic Forecasting.*

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

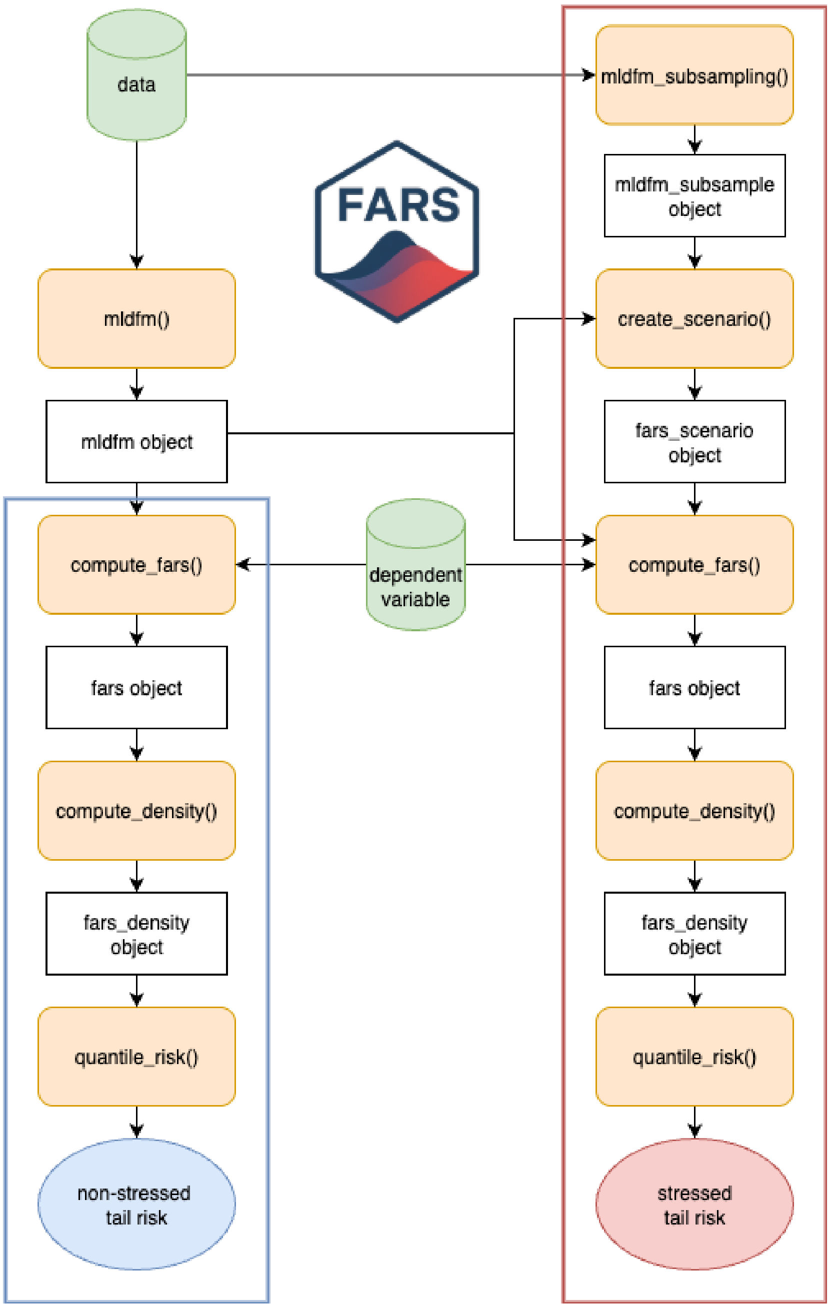
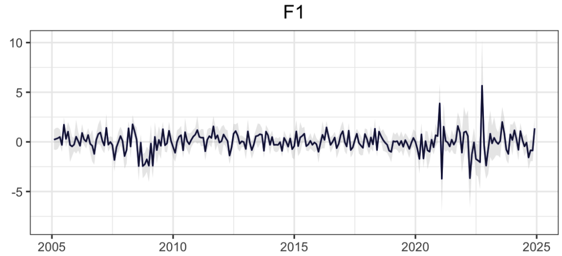
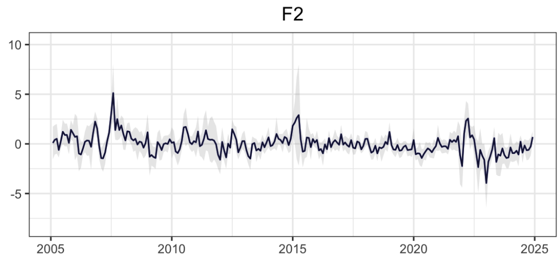


Figure 2: **FARS** package workflow for both non-stressed and stressed scenarios.

(a) Global factor



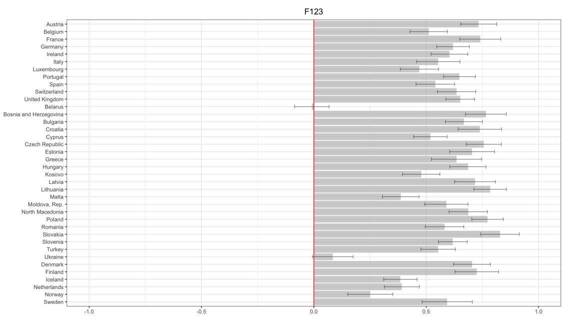
(b) Block 1 factor



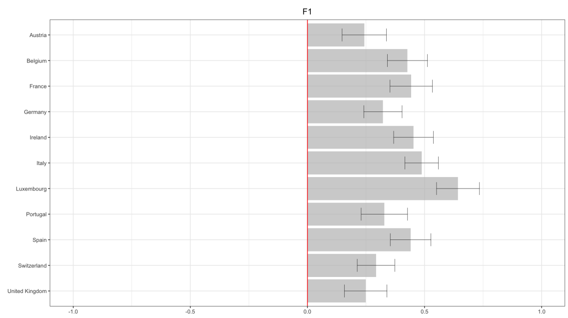
(c) Block 2 factor

(d) Block 3 factor

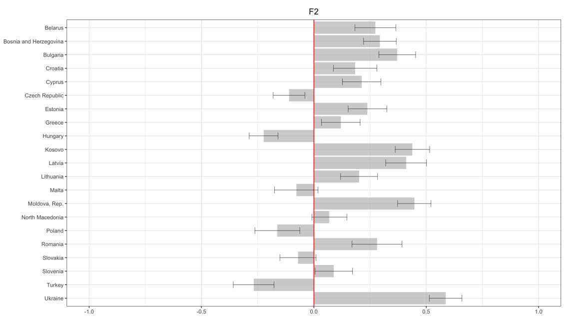
Figure 3: Estimated factors of headline inflation in Europe together with 95% confidence bounds.



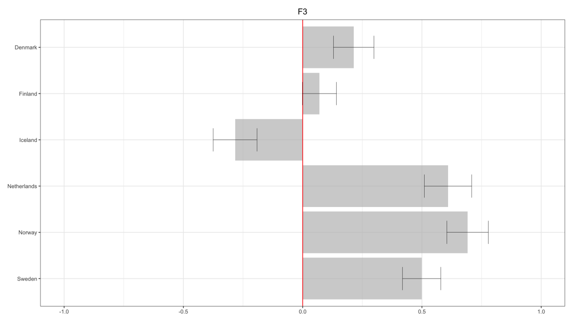
(a) Global factor loadings



(b) Block 1 factor loadings



(c) Block 2 factor loadings



(d) Block 3 factor loadings

Figure 4: Estimated factor loadings of headline inflation in Euroep together with 95% confidence bounds.

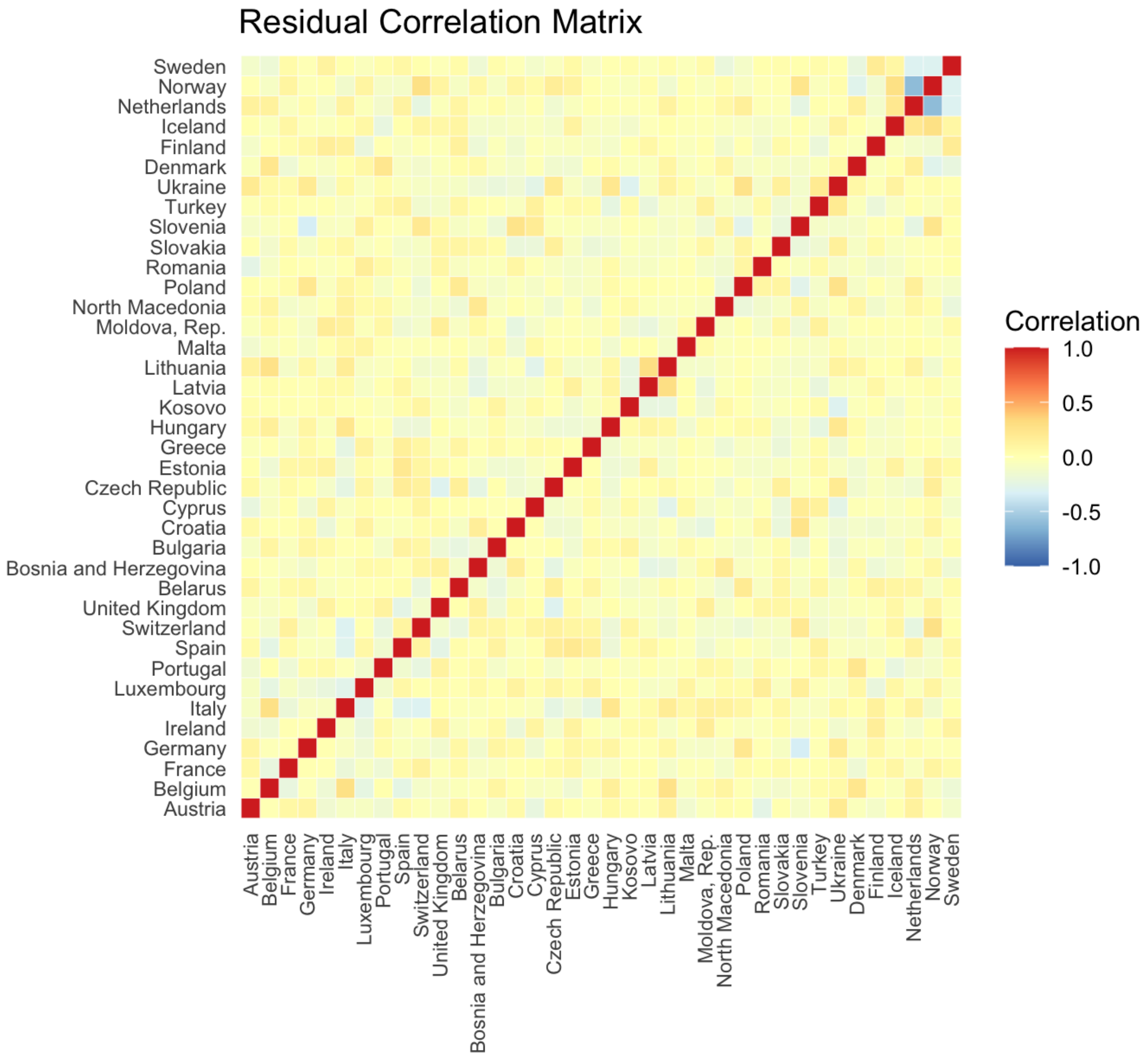
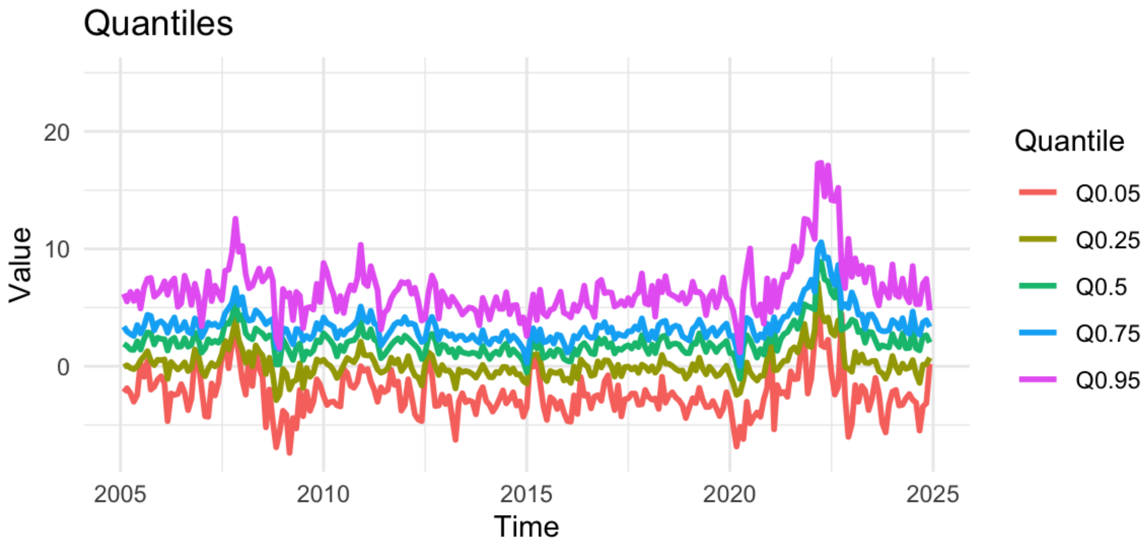
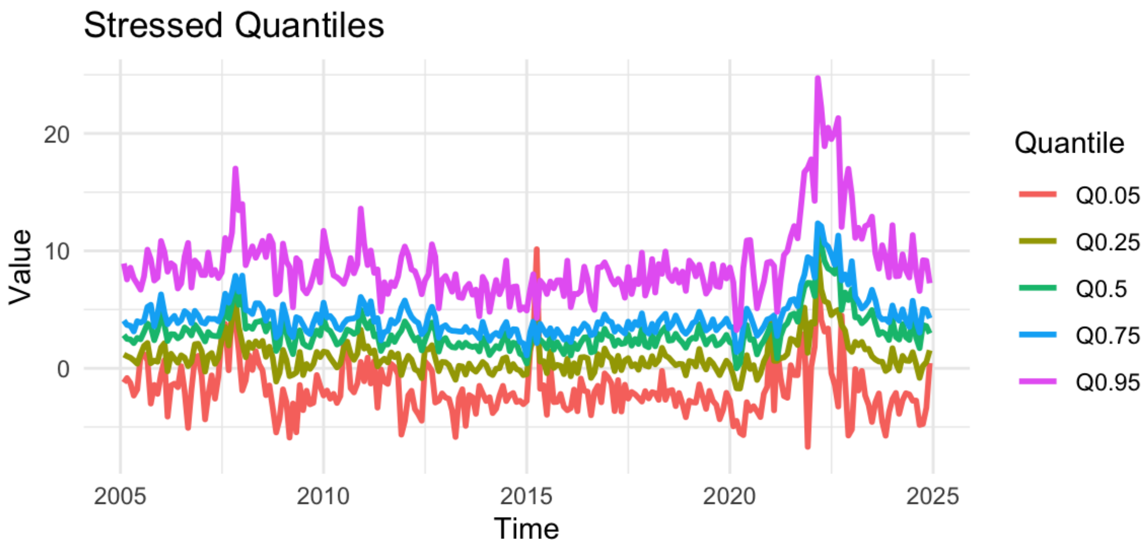


Figure 5: Correlation heatmap of estimated idiosyncratic components of headline inflation in Europe.

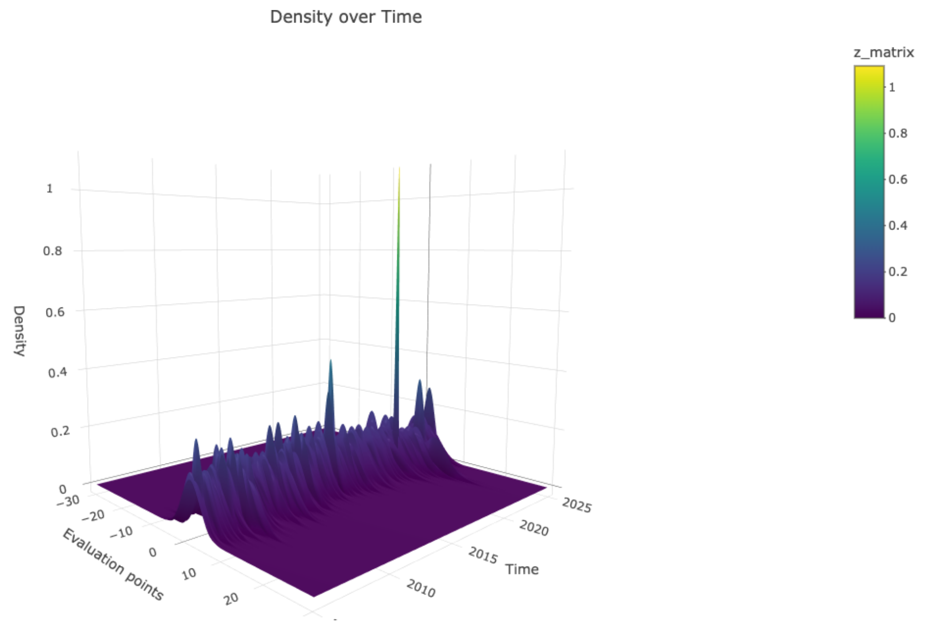


(a) Non-stressed scenario

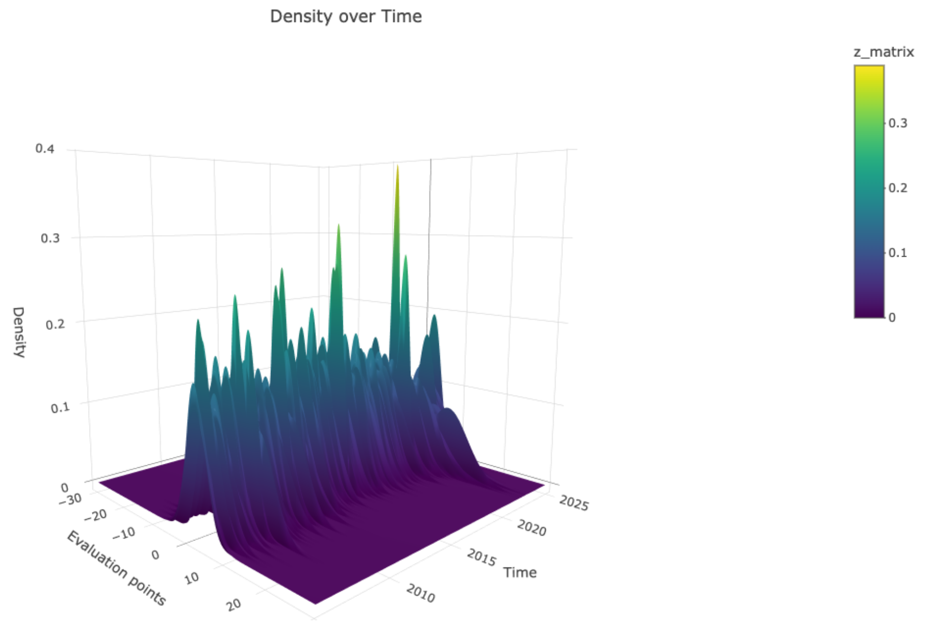


(b) Stressed scenario

Figure 6: Non-stressed (top panel) and stressed scenario (bottom panel) quantiles for Germany headline inflation.



(a) Non-stressed density



(b) Stressed density

Figure 7: Non-stressed (top panel) and stressed (bottom panel) densities for Germany headline inflation.

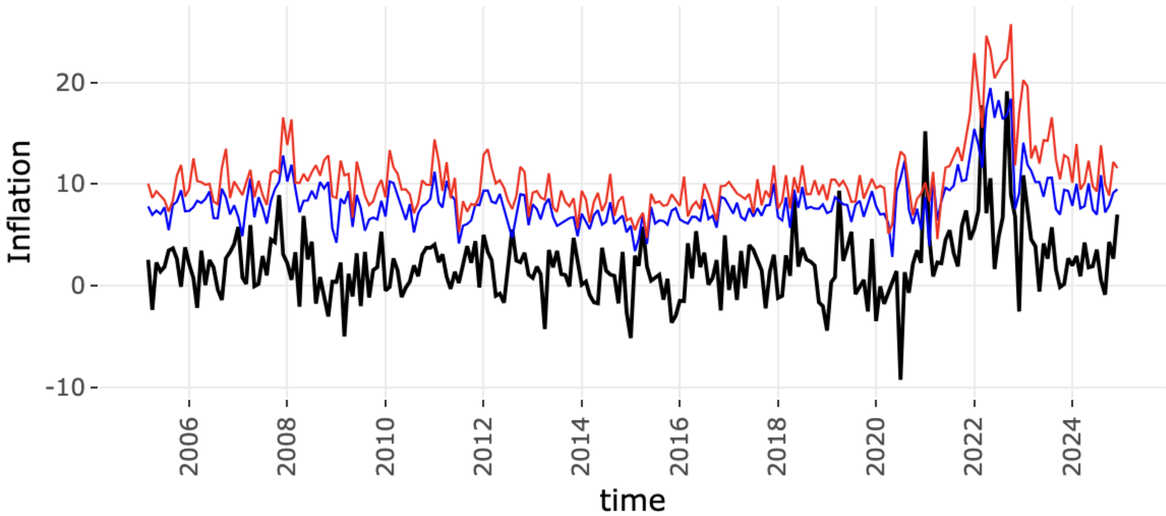
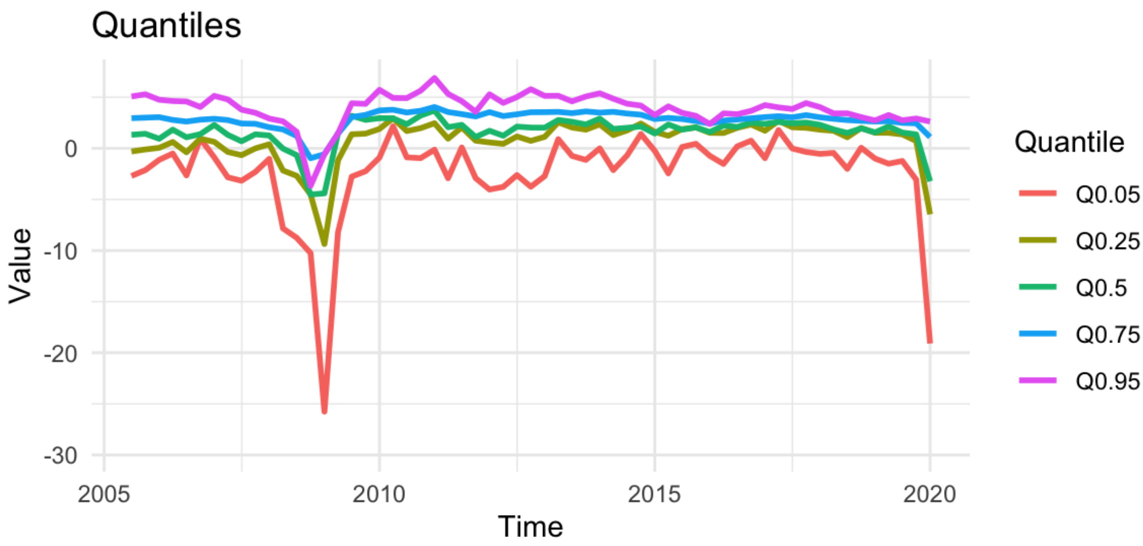
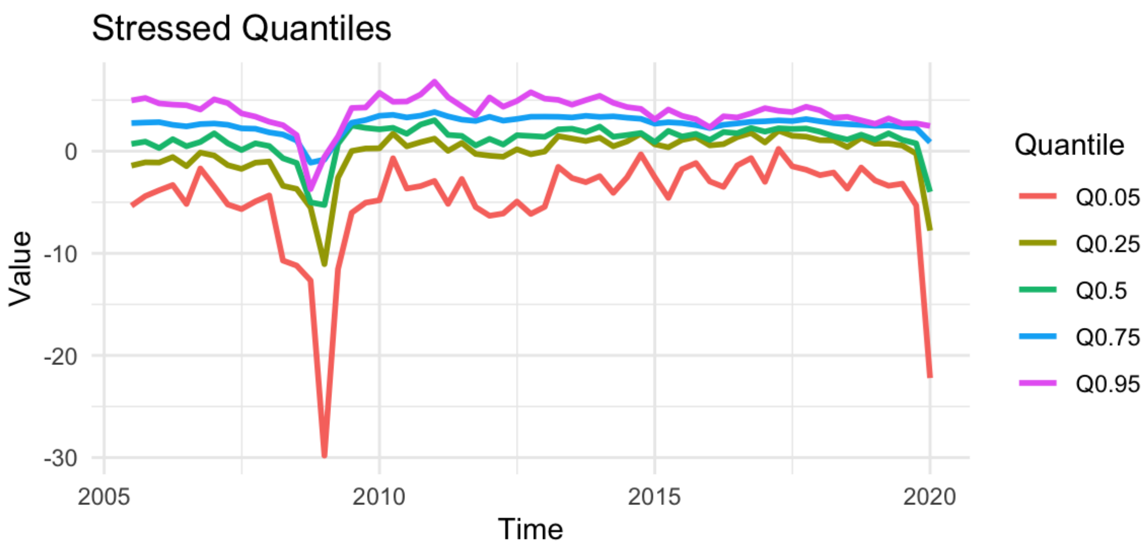


Figure 8: Germany monthly mom headline inflation (black lines), together with 99% IaR (blue) and 99% IiS stressed with $\alpha = 99\%$ (red).

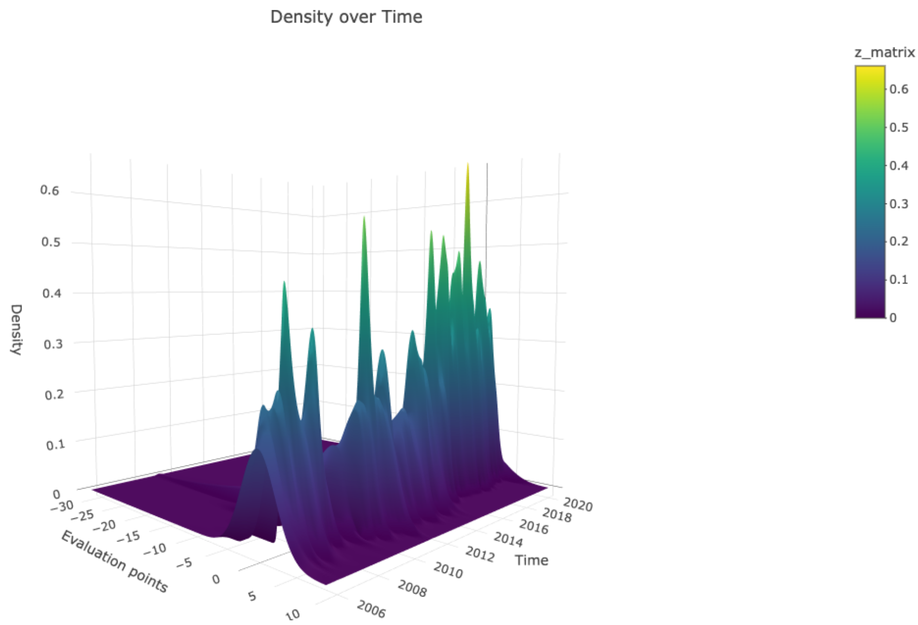


(a) Non-stressed scenario

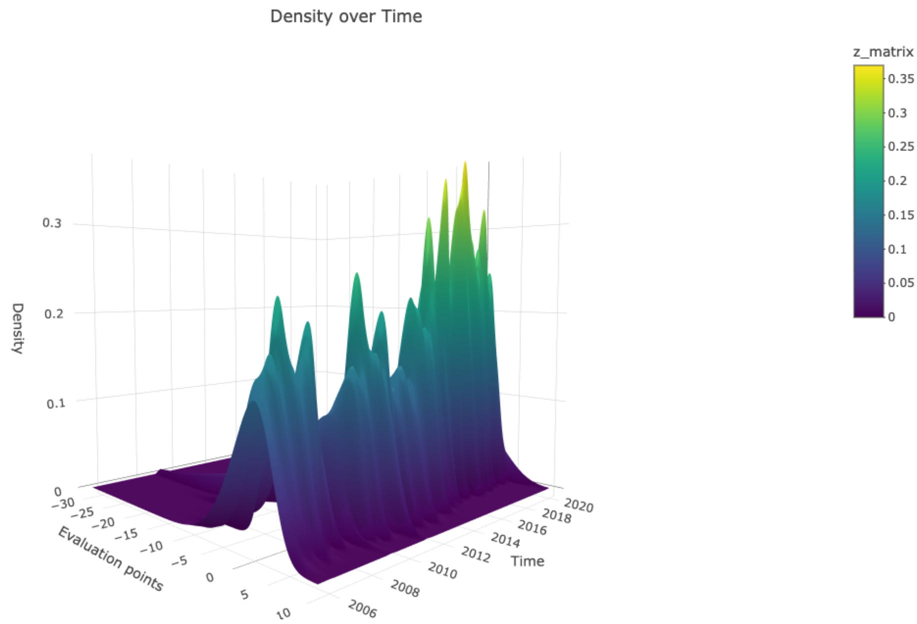


(b) Stressed scenario

Figure 9: Non-stressed (top panel) and stressed scenario (bottom panel) quantiles for US GDP Growth.



(a) Non-stressed density



(b) Stressed density

Figure 10: Non-stressed (top panel) and stressed (bottom panel) densities for US GDP Growth.

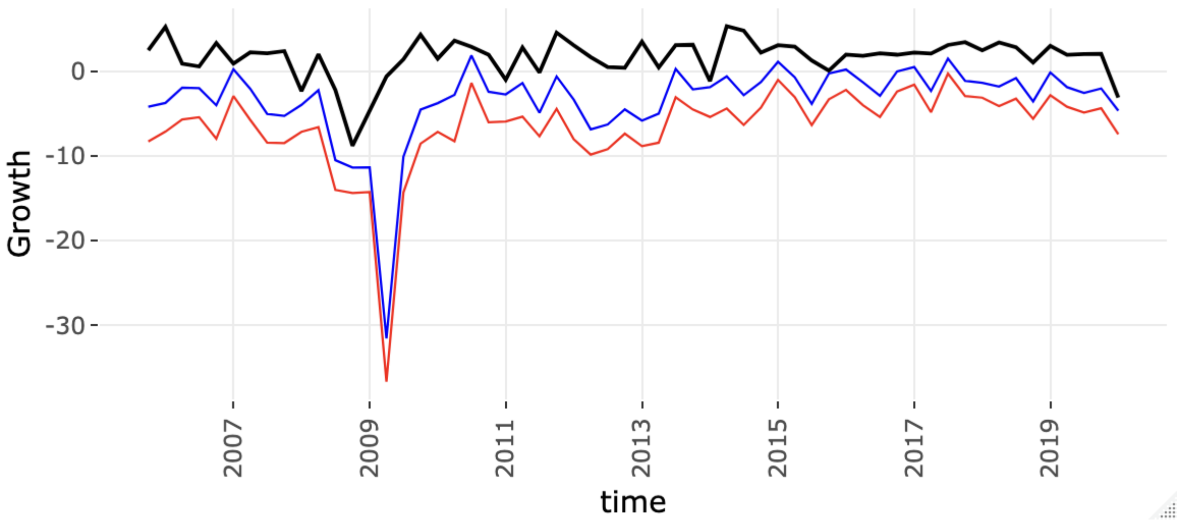


Figure 11: US quarterly growth: observed annualized rates in black, 1% GaR in blue and 1% GiS stressed with $\alpha = 99\%$ in red.