Telecom Paris

Capital Fund Management

Professional thesis:

Nowcasting Economic Activity

Romain Legrand

MS Big Data

2019 - 2020

Academic Supervisor:

David Bounie

Professional supervisor:

Agustin Lifschitz

Contents

1 Capital Fund Management

1.1 General presentation

Capital Fund Management (CFM in short) is a global asset management company. It was founded in 1991 by Jean-Pierre Aguilar and Bruno Combier, two former HEC students. In 2000, CFM merged with Science and Finance, a company founded in 1994 by Jean-Philippe Bouchaud, a graduate from the French Ecole Normale Supérieure. After the demise of Jean-Pierre Aguilar in 2009 the company has been collegially managed by Jean-Philippe Bouchaud, Philippe Jordan, Marc Potters, Jacques Saulière and Laurent Laloux.

Though CFM is based in Paris, it also has offices in New York City, London, Tokyo and Sydney. It currently employs more than 270 staff worldwide from 30 countries, most based in Paris. Its activities represent more than $10 billions in asset management, placing it in the top 100 hedge funds in the world.

CFM takes a scientific and academic approach to finance, using quantitative and systematic techniques to develop alternative investment strategies and products for institutional investors and financial advisers. This scientific approach is combined with the latest technology to analyse large quantities of data, identify patterns, then develop and implement trading algorithms.

Academic research is at the core of CFM's activities. In particular, CFM is a pioneer of econophysics. It has innovated by applying research and academic techniques from physics to finance and economics, seeking to create consistent returns not correlated to wider market performance and applying appropriate risk parameters. CFM maintains tight links with the academic world, as demonstrated by the creation of the Capital Fund Management Chair for Econophysics and Complex Systems at Ecole Polytechnique in 2019.

CFM is primarily an investment company. Figure 1.1 summarizes some key figures on its investment activities.

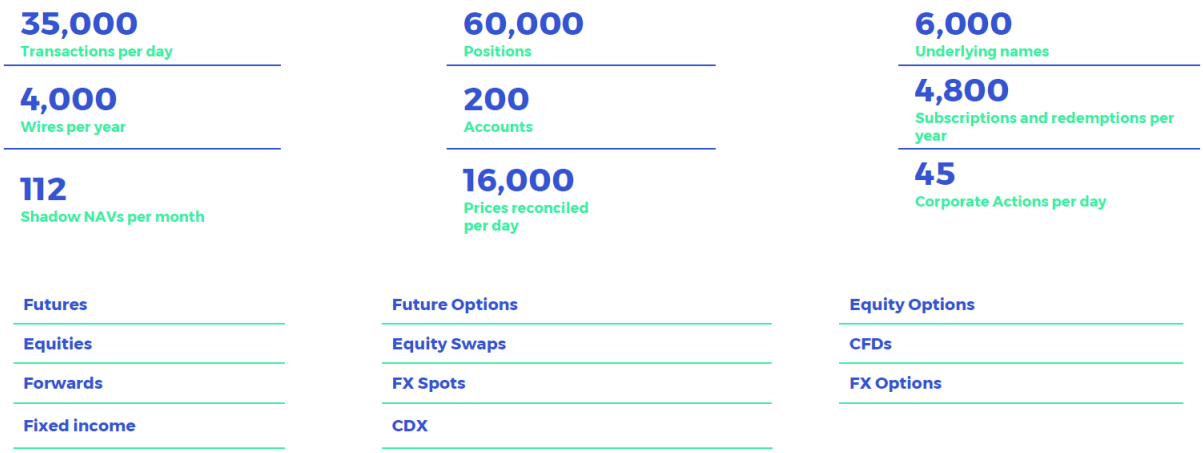


Figure 1.1: Variety and volumes of products traded

In terms of investment strategies, CFM proposes two main directions. The first direction exists since 2002 and represents the range of "hedge fund" strategies revolving around its two main Alpha products: Stratus and Discus. These two funds are highly research-driven, and involve sophisticated proprietary models. They trade at medium frequency (daily to monthly), and involve high frequency technologies for execution. Clients of these products incur premium fees. The second direction is newer (2014) and represents the range of alternative Beta strategies. It relies on two funds: "Long Only ESG" (a topical investment fund with a quantitative touch), and "Systemic Global Macro" (a hybrid product). Strategies on these funds rely on standard models (enhanced versions), and trade more "institutional" products (ISD, ISE, IST, ISB...), at lower frequency. These products involve lower fees.

In short, CFM is a leading asset management company trading a wide variety and a high volume of products, using sophisticated models derived from academic research and powerful, state-of-the-art technological platforms.

1.2 Data Analytics

Within the IT department of CFM, the Data division is in charge of collecting, processing, and extracting information from the data. The data considered at CFM in general is quite diverse: market data (real time prices and trading orders, representing more than 10000 products and

several To of data a day); macroeconomic indicators (national and regional statistics, closely monitoring the calendar of data release); corporate fundamentals (operating accounts); and alternative data (texts, graphs, forecasts...).

Within the Data division, The Data Analytics team consists of data engineers and data scientists working on the data pipelines that eventually lead to investment decisions. The Data Analytics team works mainly with two types of data. The first type is time-series, mainly for prediction purposes. Traditional machine learning models are used, along with tools like Shap or Eli5 for model understanding and Dash for data visualisation. The second type is alternative data like text or graphs, for which specialized libraries are used (TensorFlow or Torch for NLP, Neo4J or NetworkX for networks).

The projects in the team follow a flow from exploration to production, along 4 main steps: data engineering (access to providers and creation of data flow); data characterization (data history and detection of bias); value extraction (prediction with machine learning models); production and support (to create programmes that are well-documented, tested and sustainable).

Macro-financial datasets and time series enter the scope of the Data Analytics Division. In this respect, the Data Analytics team proposed in 2020 an internship more specifically oriented towards predicting (or rather, “nowcasting”, see below) economic activity. This project constitutes the object or the present report.

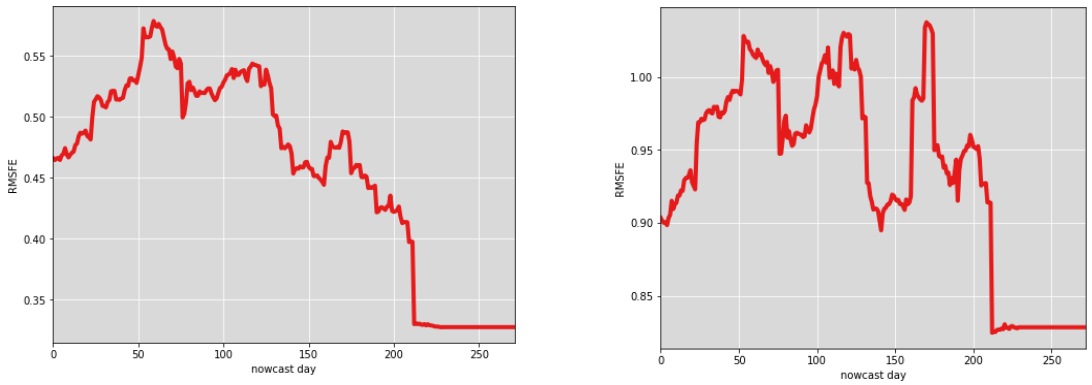
2 The project

2.1 The Nowcasting.com trial

To design inovative investment strategies, CFM is continuously testing alternative datasets and new data providers. In 2020, CFM benefited from a trial proposed by the company Now-casting.com. Founded by former ECB economist Lucrezia Reichlin, Now-Casting.com is an online service delivering high frequency, short-term forecasts for the world's major economies, in real time. These forecasts (or "nowcasts") are generated by a dynamic factor model initially developed by Giannone et al. (2008). Specifically, the model uses a large dataset of several dozens of monthly macroeconomic data to predict the lower frequency, quarterly real GDP growth rate. The services of Now-casting.com cover a wide range of countries and geographic areas such as the United States, the Euro Area, Japan, Germany, Italy, Spain or the United Kingdom. The econometric methodologies they use permit not only to predict the growth rate of real GDP, but also to forecast any monthly feature involved in the model. These features include key macroeconomic and financial variables like industrial production, business sales, consumer prices, house constructions, federal funds rates, or mortgage rates.

If these predictions are accurate, the dataset certainly carries useful information for investment decisions, and CFM might find added value in using the service. During the internship, the first task thus consisted in investigating the dataset, exploring a number of countries and features, and comparing the predictions with the ground truth. This resulted in a brief report (the complete report can be found in Appendix A.1).

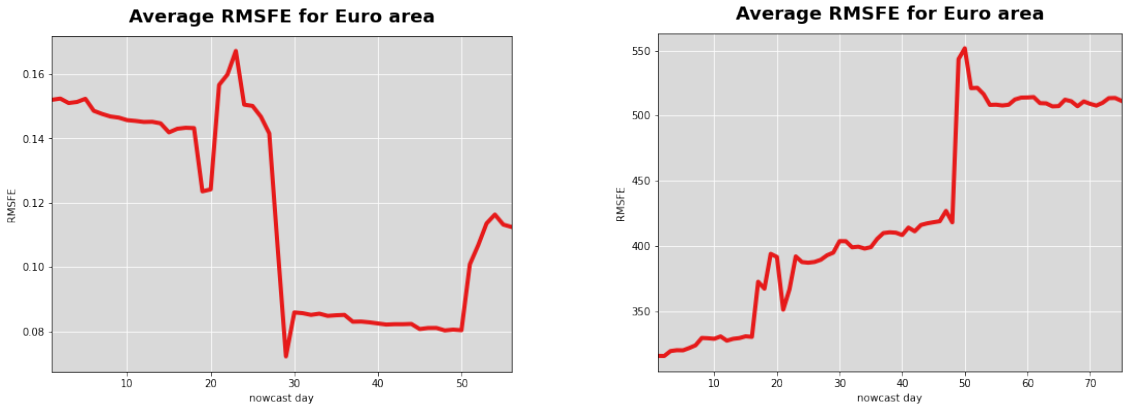
Overall the conclusions were mixed, both regarding the accuracy of the predictions and the model capacity to improve its predictions while new information obtains. Certain countries like the Euro area seemed to produce fairly good GDP nowcasts, the latter effectively improving while more information was added to the model. Other countries like Italy displayed really poor performance, with prediction errors switching back and forth until the release date of GDP. This is illustrated in Figure 2.1.



(a) Euro Area (b) Italy

Figure 2.1: Average RMSE of Nowcasting.com on predictions for GDP growth

For the other features, the conclusions were overall similar. The predictions ranged from fair (Business confidence index for the Euro area) to absurd, with RMSE continuously increasing as more information becomes available (new car registrations in the Euro Area). This is demonstrated by Figure 2.2.



(a) Business confidence index (b) New car registrations

Figure 2.2: Average RMSE of Nowcasting.com on feature predictions

These mediocre performances were not fully convincing. This proved especially true for the United States, which yet represents the main country for macro-financial decision making. The significant price of the service then led to a simple consideration: could it be possible to replicate internally the dynamic factor model used by Now-casting.com, and achieve performances that prove equal - or better - than the ones they propose? As a first step, the United States sounded like a natural candidate since is one of the key countries for global macroeconomic developments, and because long series of data are easily and publicly available.

2.2 Project design

The starting point of the project only consisted in replicating the dynamic factor model used by Now-casting.com for the United States. Quickly however, the idea emerged that the project could be more comprehensive and cover the wider question of nowcasting in general. Indeed, the prediction of key economic variables such as GDP at short term or very short term represents a major challenge for the finance industry. This problem, formally known as nowcasting (a contraction of "now" and "forecasting") has attracted the attention of economists and econometrician for more or less two decades. The data science industry on the other hand has remained more agnostic regarding this issue, and has simply applied regular machine learning methods to obtain short-term predictions. Today, it is still unclear what is the best approach to predict macroeconomic variables in the short term. The question is rendered more difficult by the lack of exchange between the different fields, machine learning models being often unknown to econometrician, and vice versa, data scientists often having a limited culture in econometric models.

The approach of the project would thus consist in testing a range of different methodologies from different fields, in an attempt to assess which one appears most suitable for the purpose of nowcasts. Concretely, three fields were considered as candidates: nowcasting models, econometrics, and machine learning.

Nowcasting models are statistical models that are specifically tailored for short term predictions. In particular, these models account for two specificities of nowcasting models: mixed frequencies, and missing values. Indeed, nowcasting typically involves the prediction of some low frequency variable (for instance quarterly GDP) from the information of higher frequency features (for instance monthly data like industrial production and employment) to obtain insight on the yet unknown quarterly realization. Because high frequency variables are usually published asynchronously (for instance, not all monthly economic variables are updated the same day in the course of a given month), panels are rarely balanced. Nowcasting models are thus built to cope with these two aspects. Three models were retained in this field for the project: the dynamic factor model of Giannone et al. (2008); the MIDAS regression of Ghysels et al.

(2004); and the mixed frequency Bayesian VAR of Schorfheide and Song (2015).

Econometrics has a long tradition of prediction models. Since the seminal contribution of Sims (1980), Vector Autoregressions (VAR models, in short) have become the workhorse of macroeconomics. The methodology has been widely adopted by Central Banks and financial institutions for their prediction routines. VAR models are appealing because they are flexible and integrate multivariate settings. They also achieve good predictive performance. Three VAR models are considered for the project: a simple maximum likelihood VAR; a Bayesian VAR in the line of Karlsson (2012); and a time-varying VAR, following the methodology of Primiceri (2005).

Finally, Machine learning models have become central in data analysis with the rise of big data technologies in the 2010's. Powerful models can now be trained and used for prediction purposes thanks to the power of computers and distributed algorithms. Three machine learning methodologies are used in this project: the LSTM model of Hochreiter and Schmidhuber (1997), since it is conceptually closest to the VAR model from the econometrics field; the random forest model of Breiman (2001); and the gradient boosting approach of Friedman (2001).

2.3 The dataset

The dataset of the project is designed to be consistent with the large macro dataset used in the dynamic factor model of Giannone et al. (2008). It comprises a series of quarterly real GDP for the United States, which represents the main target to predict. Besides GDP, the nowcasts are realised from a set of 31 monthly variables for the United States, covering different sides of economic activity. The dataset is organised over seven blocks: general business indicators, production and sales, labor and wages, macroeconomic aggregates, prices, money and credits, interest rates and finance. All the series come from standard public sources such as the OECD, the Federal Reserve, or the Saint Louis FRED database. The details of the series are provided in Figure 4.

	feature	block	details
1	pmi	business_indicators	purchasing managers index, Institute for Supply Management (ISM)
2	business_outlook_survey	business_indicators	manufacturing business outlook survey, diffusion indexes, seasonally adjusted
3	business_confidence_index	business_indicators	Main economic indicators (MEI) : business confidence indicator, normalised, amplitude and seasonally adjusted
4	consumer_confidence_index	business_indicators	Main economic indicators (MEI) : business confidence indicator, normalised, amplitude and seasonally adjusted
5	industrial_production	production_and_sales	Index, 2012=100, seasonally adjusted
6	mgdp	production_and_sales	IHS Markit monthly GDP index
7	business_sales	production_and_sales	total business sales, millions of dollars, seasonally adjusted
8	new_residential_sales	production_and_sales	houses for sale at end of period, thousands of units
9	inventories	production_and_sales	total business inventories, millions of dollars, seasonally adjusted
10	unemployment_rate	labor_and_wages	monthly unemployment rate, all persons, seasonally adjusted
11	employment	labor_and_wages	all employees, total nonfarm, thousands of persons, seasonally adjusted
12	weekly_hours	labor_and_wages	average weekly hours of production and nonsupervisory employees, manufacturing, seasonally adjusted
13	hourly_earnings	labor_and_wages	average hourly earnings of production and nonsupervisory employees, total private, dollars per hour, seasonally adjusted
14	consumer_credit	labor_and_wages	total consumer credit owned and securitized, outstanding, billions of dollars, seasonally adjusted
15	personal_income	labor_and_wages	real disposable personal income, billions of chained 2012 dollars, annual rate, seasonally adjusted
16	federal_debt	macroeconomic_aggregates	market value of gross federal debt, billions of dollars, not seasonally adjusted
17	exports	macroeconomic_aggregates	exports with world, millions of dollars, seasonally adjusted
18	imports	macroeconomic_aggregates	imports with world, millions of dollars, seasonally adjusted
19	ppi	prices	producer price index, 1982=100, all commodities, not seasonally adjusted
20	cpi	prices	consumer price index, 1982=100, seasonally adjusted, all urban consumers, all itemsin U.S city average
21	monetary_base	money_and_creadits	monetary base, total , millions of dollars, not seasonally adjusted
22	bank_assets	money_and_creadits	total assets, all commercial banks, millions of dollars, seasonally adjusted
23	bank_liabilities	money_and_creadits	total liabilities, all commercial banks, millions of dollars, seasonally adjusted
24	mortgage_rate	money_and_creadits	30-Year fixed rate mortgage average, percent, not seasonally adjusted (monthly, end of period value)
25	federal_funds_rate	interest_rates_and_finance	effective federal funds rate
26	treasury_bill	interest_rates_and_finance	3-month Treasury bill, secondary market rate
27	treasury_bill_10	interest_rates_and_finance	10-year Treasury bill, constant maturity rate
28	effective_exchange_rate	interest_rates_and_finance	narrow effective exchange rate, index, 2010=100, not seasonally adjusted
29	spot_euro_us	interest_rates_and_finance	Euro/ECU exchange rates with US dollar
30	nyse_composite_index	interest_rates_and_finance	NYSE delayed price, currency in USD, open
31	vix	interest_rates_and_finance	VIX index, not seasonally adjusted

Figure 2.3: Details of the monthly features used in the project

This dataset calls for a few comments. First, the raw data is typically non-stationary. It must thus be differenced before being used within a time-series or machine learning framework. Following the recommandations of Giannone et al. (2008), three possible transformations are applied. These transformations are detailed in Table 2.1.

	Transformation	Description
1	$x_{i,t} = (1-L^3)(1+L+L^2)X_{i,t}$	quaterly difference
2	$x_{i,t} = (1-L^3)(1+L+L^2)log(X_{i,t}) \times 100$	quarterly growth rate
3	$x_{i,t} = (1-L^3)(1+L+L^2)(1-L^{12})log(X_{i,t}) \times 100$	quarterly difference of yearly growth rate

Table 2.1: Raw data transformations

These transformations serve two purposes: stationarize the data, and formulate monthly values as quarterly quantities. This way, monthly estimates become comparable with the quarterly values of the real GDP series. The transformation applied to each specific series (along with the source of the data series) are reported in Appendix A.2. An overview of the transformed series is provided in Figure 2.4.

Second, due to the availability of certain data series, the sample only covers the period from 1993 to 2019. This represents a monthly sample of size 320. While this is usually fine for now-casting and econometrics models, it is usually insufficient for machine learning methods which are extremely data greedy and requires several thousands of observations to properly learn the possible non-linearities in the data. The latter are thus clearly at a structural disadvantage here. The choice to end the sample in 2019 is explicit. Because of the recent COVID pandemy, the world economy collapsed brutally in 2020 so that the data for this year represents a huge outlier. Preliminary estimates with the COVID year proved quite abnormal and sometimes displayed aberrent behaviours. While predicting economic behaviours in a time of changing environment is a question with an interest on its own, it is not the purpose of the present project. As a consequence, it seemed safer to exclude 2020 from the sample to keep only regular behaviours, at least for now.

Figure 2.4: Processed features

Third and finally, the dataset was sometimes adapted to fit a given model. Most econometrics and machine learning models for instance can only handle single frequency data. In this case, the dataset had to be reduced to a quarterly dataset, omitting two out of three months in the original monthly set to keep only the months were quarterly releases occur. This reduces the sample size to just over 100, which represents a very short sample. Also, the full dataset of 31 variables does not seem suitable for all models. While certain methodologies like the dynamic factor model or the random forest usually perform best with a large dataset, other methods like the VAR model perform better with a dozen of feature or so (see for instance Giannone et al. (2015) for a discussion on the predictive performance of large VAR model). As a consequence, and to keep the exercise fair to all the models considered, a smaller dataset was created as a subset of the original dataset. This small dataset comprises only 9 monthy features (in addition to quarterly GDP), which are detailed in Figure 2.5.

	feature	block	details
1	business_confidence_index	business_indicators	Main economic indicators (MEI) : business confidence indicator, normalised, amplitude and seasonally adjusted
2	industrial_production	production_and_sales	Index, 2012=100, seasonally adjusted
3	business_sales	production_and_sales	total business sales, millions of dollars, seasonally adjusted
4	new_residential_sales	production_and_sales	houses for sale at end of period, thousands of units
5	unemployment_rate	labor_and_wages	monthly unemployment rate, all persons, seasonally adjusted
6	weekly_hours	labor_and_wages	average weekly hours of production and nonsupervisory employees, manufacturing, seasonally adjusted
7	cpi	prices	consumer price index, 1982=100, seasonally adjusted, all urban consumers, all itemsin U.S city average
8	bank_assets	money_and_creadits	total assets, all commercial banks, millions of dollars, seasonally adjusted
9	federal_funds_rate	interest_rates_and_finance	effective federal funds rate

Figure 2.5: Features included in the reduced dataset

3 The models

3.1 Nowcasting: the dynamic factor model

The dynamic factor model is the econometric methodology used by Now-casting.com to produce its predictions. The model has been originally proposed in Giannone et al. (2005), but the main reference is the contribution of Giannone et al. (2008), which essentially repeats the same methodology and provides a detailed technical appendix. Doz et al. (2011) provides deeper theoretical developments, justifying the statistical correctness of the two-steps methodology developed below. Banbura et al. (2010) proposes the same methodology with an alternative use of the EM algorithm in place of the Kalman filter.

The dynamic factor model is a mixed frequency model for nowcasting. More specifically, it aims at predicting quarterly GDP growth from many higher-frequency, monthly economic variables. The dataset used by Giannone et al. (2008) is consistent with the one proposed for this project, except that it is larger (around 250 features, most variables being decomposed into sub-components). Because a large number of data series in the model would lead to the curse of dimensionality issue, the authors propose to reduce the many series to a few dynamic factors that are in turn be used to predict real GDP.

3.1.1 Formulation

Assume that are n monthly variables involved in the model over T periods. Denote by $x_{i,t}$ the value of variable i at period t, with $i = 1, \cdots, n$ and $t = 1, \cdots, T$. It is assumed that each variables is linearly related to a small number r of factors with $r \ll n$. Denoting by $f_{j,t}$ the value of factor j at period t, the relation obtains as:

$$x_{i,t} = \mu_i + \lambda_{i1} f_{1,t} + \cdots + \lambda_{ir} f_{r,t} + \xi_{i,t} \qquad \xi_{i,t} \sim \mathcal{N}(0, \psi_i) \tag{3.1}$$

Stacking then all the n variables in a single vector x_t, the equation rewrites compactly as:

$$x_t = \mu + \Lambda f_t + \xi_t \qquad \xi_t \sim N(0, diag(\psi)) \tag{3.2}$$

where x_t, μ, f_t, ξ_t and ψ are n-dimentional vectors, and Λ is a $n \times r$ matrix of factor loadings. It is further assumed that the dynamic factors follow an auto-regressive process:

$$f_t = Af_{t-1} + Bu_t \qquad u_t \sim \mathcal{N}(0, I_q) \tag{3.3}$$

where A is a $r \times r$ matrix of autoregressive coefficients, and u_t is a q-dimensional vector of common shocks assume to drive the dynamics of the factors. Typically, we set $q \leq r$ in order to capture the lead and lag relations among variables.

Finally, denote by $\hat{y}_t$ the nowcast for real GDP growth at time t. It is related to the value of the dynamic factor by the following equation:

$$\hat{y}_t = \alpha + \beta \hat{f}_t \tag{3.4}$$

where $\hat{f}_t = \mathbb{E}(f_t|\Omega_t)$, that is, $\hat{f}_t$ is the optimal projection of f_t, given the available information set Ω_t.

3.1.2 Estimation

If the series of dynamic factors f_t was known, estimating the model would be straightforward. However, they cannot be estimated directly from (3.2) since the parameters μ and Λ are also unknown. For this reason, Giannone et al. (2008) propose a procedure in 2 steps: first, estimate a preliminary version of the r factors by applying PCA (principal component analysis) on x_t; second, estimate the model parameters from these raw factors, then use these parameters to obtain optimal projections of the factors from Kalman filtering. It is then trivial to obtain the predictions for GDP and the other features.

In details, the authors propose the following procedure:

Algorithm 1: dynamic factor model

1. Feature standardization:
Standardize the data so that it has zero mean and unit variance: $z_{i,t} = (x_{i,t} - \hat{\mu}_i)/\hat{\sigma}_i$. Then calculate the correlation matrix $S = \frac{1}{T}\sum_{t=1}^{T} z_t z_t'$.

2. Principal component analysis:

denote by D the $r \times r$ diagonal matrix with the r largest eigenvalues of S and by V the corresponding $n \times r$ matrix of eigenvectors. A raw estimate of the dynamic factors obtains as: $f_t = V' z_t$.

3. Parameters for the feature dynamics:

$\Lambda = \sum_{t=1}^{T} z_t f_t' \left(\sum_{t=1}^{T} f_t f_t' \right)^{-1} = V$ $\qquad$ $\psi = diag(S - VDV)$

4. Parameters for the factor dynamics:

$A = \sum_{t=2}^{T} f_t f_{t-1}' \left(\sum_{t=2}^{T} f_{t-1} f_{t-1}' \right)^{-1}$ $\qquad$ $B = MP^{1/2}$

where P is the $q \times q$ diagonal matrix with the q largest eigenvalues of Σ, M is the $r \times q$ matrix of corresponding eigenvectors, and Σ is defined as:

$\Sigma = \frac{1}{T-1} \sum_{t=2}^{T} f_t f_t' - A \left(\sum_{t=2}^{T} f_{t-1} f_{t-1}' \right) A'$

5. Optimal projections of the factors:

Given that (3.2) and (3.3) are formulated in state-space form, they can be used to obtain optimal projections $\hat{f}_t$ of the factors from the Kalman smoother. Please refer to Appendix A.3 for details on the Kalman filter procedure.

6. Parameters for GDP nowcasting:

With the projected factors $\hat{f}_t$ at hands, estimate (3.4) by OLS.

3.1.3 Prediction

Once the model is trained, it is straightforward to obtain predictions. From (3.2), predictions for the features at horizon $t+h$ can be obtained as:

$$\hat{x}_{t+h} = \mu + \Lambda \hat{f}_{t+h} \tag{3.5}$$

as for predictions for quarterly GDP, they obtain directly from (3.4) as:

$$\hat{y}_{t+h} = \alpha + \beta \hat{f}_{t+h} \tag{3.6}$$

Both predictions require $\hat{f}_{t+h}$. But $\hat{f}_{t+h}$ can be trivially obtained from the Kalman filter, by continuing the Kalman iterations up to period $t+h$.

3.1.4 Application

The dynamic factor model is applied to the project dataset. Figure 3.1 provides an overview of the structural factors, both raw and smoothed by the Kalman filter, while Figure 3.2 plots actual GDP growth and the in-sample predictions obtained from the model.

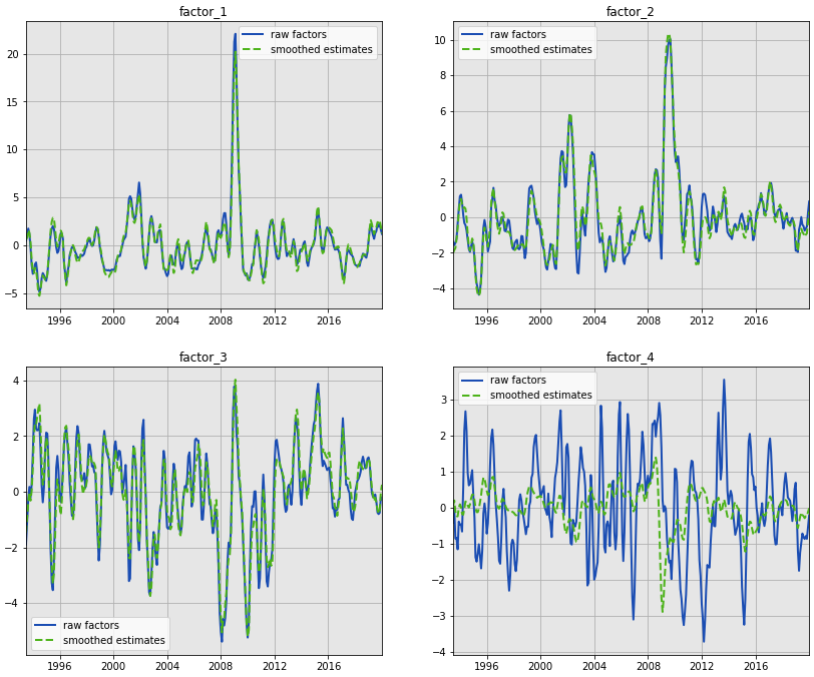


Figure 3.1: Structural factors for the project dataset

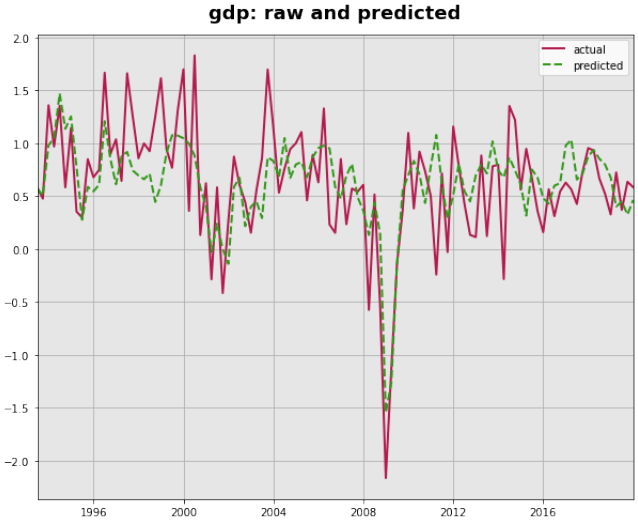


Figure 3.2: In-sample GDP: actual and predicted

The discrepancy between the raw factors and their smoothed versions seems small, except for the final one for which the prediction differs considerably from the raw estimate. There seems to exist a significant correlation between GDP and the structural factors, though this correlation is negative for some factors. It is then informative to check which features contribute to the design of each factor, and hence have most explanatory power on GDP in the dynamic factor model. This information is reported in Figure 3.3.

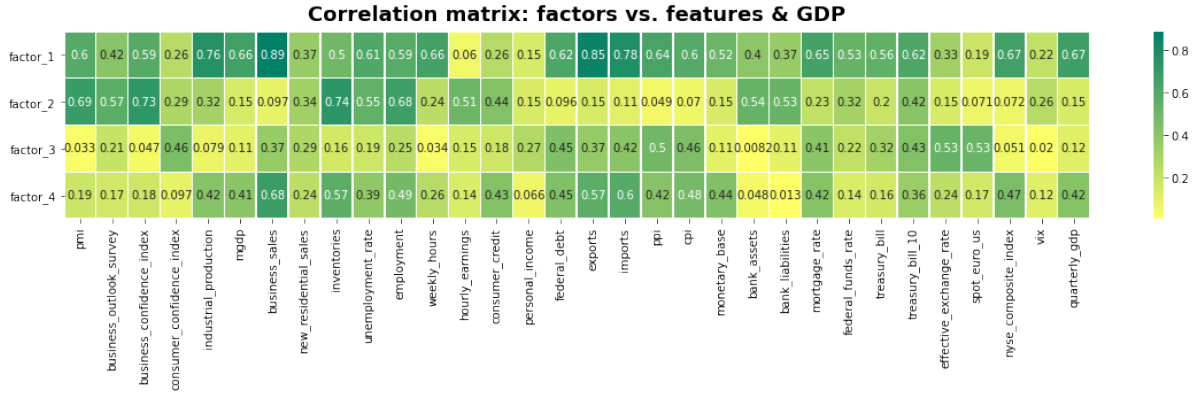


Figure 3.3: Correlations: factors, features and GDP

The figure reports the correlations (in absolute values) between each factor and the model features, along with GDP growth. The results suggest that the first factor is strongly correlated with the variables that represent the business cycles: industrial production, business sales, weekly hours, exports and imports. The second factor seems more related to business sentiment, exhibiting strong correlations with features such as the pmi, business outlook survey, the business confidence index, and inventories. The last two factors display weaker correlations with all the features, which suggests they represent economic activity in general rather than specific aspects of it.

Interestingly enough, the strongest correlation of GDP occurs with the first and fourth factors. A high correaltion with the first factor is expected, since it is the one that represents the major part of the data variance. However, the lower correlation with the second and third factors and the higher correlation with the fourth factor is more surprising. This questions the relevance of a pure principal component approach, and suggests that at least a sufficient number of components should be retained to carry enough information on GDP. This is confirmed by Figure 3.2, which shows a rather erratic accuracy of the in-sample GDP predictions, with good fit at some periods, and only loose relations at other periods.

3.2 Nowcasting: MIDAS regression

Another approach to mixed frequency modelling and nowcasting is the MIDAS regression (for MIxed DAta Sampling). This methodology was originally proposed by Ghysels et al. (2004), with an essentially theoretical approach. More applied aspects of the methodology have then been studied in contributions such as Ghysels et al. (2007) and Ghysels and Wright (2009).

3.2.1 Formulation

Similarly to the dynamic factor model, consider the case where one wants to predict one low-frequency variable (say a quarterly variable for instance) by the way of n higher frequency variables (say for instance monthly features). Denote by t the low frequency periods, and assume that the high frequency variables are observed m times during one period of low frequency (so for instance for a quaterly/monthly model, t corresponds to quarters, and there are $m = 3$ months between any two quarters). Denote by y the low frequency variable and by $x_1, \cdots, x_n$ the set of n high frequency features. Because there are m times more periods in the high frequency features, the observation y_t corresponds to the feature observations $x_{1,tm}, \cdots, x_{n,tm}$. The low frequency variable is assumed to be explained by p of its own (low frequency) lags, as well as q (high frequency) lags of the n features.

A simple MIDAS regression can then be formulated as:

$$y_t = \mu + \sum_{j=1}^{p} \gamma_i \, y_{t-j} + \sum_{i=1}^{n} \sum_{j=1}^{q} \beta_{ij} \, x_{i,tm+1-j} + \varepsilon_t \qquad \varepsilon_t \sim N(0, \sigma^2) \tag{3.7}$$

μ is an intercept term, γ_i are lag-specific coefficients for the low-frequency variable, while β_{ij} are feature and lag-specific coefficients. A direct approach would consist in estimating (3.7) as it is. However, this strategy might be problematic as one would be quickly facing the curse of dimensionality. Consider for instance the extreme case where y_t is yearly while each $x_{i,t}$ is daily. Assuming 22 open days a month or 264 open days a year, one would have to estimate 264 coefficients β_{ij} for each of the n feature to match just one period of the low frequency variable.

For this reason, Ghysels et al. (2007) suggest a more parsimonious approach that drastically reduces the dimensionality of the regression. To do so, reformulate (3.7) slightly to obtain:

$$y_t = \mu + \sum_{j=1}^{p} \gamma_i \, y_{t-j} + \sum_{i=1}^{n} \beta_i \Big(\sum_{j=1}^{q} w_{ij} \, x_{i,tm+1-j} \Big) + \varepsilon_t \qquad \varepsilon_t \sim N(0, \sigma^2) \tag{3.8}$$

The β_i are now purely feature-specific coefficients, while the $w_{i,j}$ are weights such that $\sum_{j=0}^{q} w_{i,j} = 1$ (a requirement for β_i to be uniquely defined). The strategy then consists in estimating the numerous weights $w_{i,j}$ for each feature through a small number k of parameters $\theta_i = (\theta_{i1}, \cdots, \theta_{ik})$. In particular, Ghysels et al. (2007) propose two strategies to estimate the series of $w_{i,j}$ coefficients. The first one relies on the so-called exponential Almon lag polynomial, which for $k = 2$ is defined as:

$$w_{ij}(\theta_i) = \frac{\exp(\theta_{i1} j + \theta_{i2} j^2)}{\sum_{j=1}^{q} \exp(\theta_{i,1} j + \theta_{i,2} j^2)} \tag{3.9}$$

The second approach is the Beta lag approach. It also uses only $k = 2$ parameters, and is defined as:

$$w_{ij}(\theta_i) = \frac{x_j^{\theta_{i1}-1} \, (1-x_j)^{\theta_{i2}-1}}{\sum_{j=1}^{q} x_j^{\theta_{i1}-1} \, (1-x_j)^{\theta_{i2}-1}} \qquad x_j = \frac{j}{q} \tag{3.10}$$

Both formulations are quite flexible and can handle a wide variety of structures for the weights $w_{i,j}$ (this is discussed in more details in section 3.2.2). Once the weights are estimated, there exists a direct correspondance between (3.7) and (3.8), the latter implying $\beta_{ij} = \beta_i w_{ij}$ in the former. Simply, the estimation of (3.8) is considerably more parsimonious, which makes it quite appealing.

3.2.2 Estimation

Estimation of a MIDAS model consists in calculating the values of μ, γ_i, β_i, and θ_i. Formally, it consists in finding the parameter values that minimize the sum of squared residuals in (3.8):

$$\{\hat{\mu}, \hat{\gamma}_i, \hat{\beta}_i, \hat{\theta}_i\} = \underset{\mu, \gamma_i, \beta_i, \theta_i}{argmin} \left(y_t - \mu - \sum_{j=1}^{p} \gamma_i \, y_{t-j} - \sum_{i=1}^{n} \beta_i \Big(\sum_{j=1}^{q} w_{ij} \, x_{i,tm+1-j} \Big) \right)^2 \tag{3.11}$$

Solving for (3.11) only implies $1+p+3n$ parameters, thanks to the parsimonious lag polynomials (3.9) and (3.10). This may sound like an easy problem at first, but a number of pitfalls complicate the estimation. First, the lag polynomials render the model nonlinear, preventing direct estimation by OLS. In theory, one could just try minimzing (3.11) with a numerical solver. In practice, this will typically fail if q or n is even moderately large, due to the highly nonlinear nature of the lag polynomials and the size of the parameter space.

For this reason, Ghysels and Qian (2016) propose to use a "profile likelihood" approach that simplifies the estimation process. As a first simplification, the authors suggest to reduce the lag polynomials to only one parameter, noticing that a wide range of realistic declining weight structures can be obtained by setting $\theta_1 = 0$ in (3.9) or $\theta_1 = 1$ in (3.10). This is illustrated in Figure 3.4.

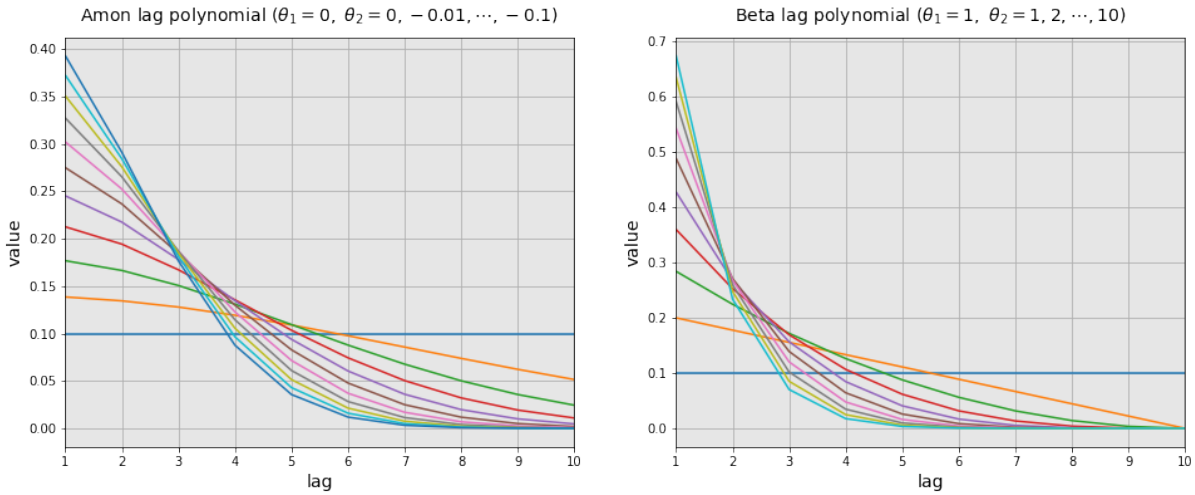


Figure 3.4: Amon and Beta lag polynomials for fixed θ_1 values

With this first simplification only θ_2 remains to be estimated, which considerably simplifies the work of a numerical solver.

The second simplification is best understood by rewriting (3.8) in compact form as:

$$y = c + Y\gamma + \beta_1 X_1 w_1 + \cdots + \beta_n X_n w_n + \varepsilon \tag{3.12}$$

with:

$$y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_T \end{pmatrix} \quad c = \begin{pmatrix} \mu \\ \mu \\ \vdots \\ \mu \end{pmatrix} \quad Y = \begin{pmatrix} y_0 & y_{-1} & \cdots & y_{-p} \\ y_1 & y_0 & \cdots & y_{-p-1} \\ \vdots & \vdots & \ddots & \vdots \\ y_{T-1} & y_{T-2} & \cdots & y_{T-p)} \end{pmatrix} \quad \gamma = \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_p \end{pmatrix}$$

$$X_i = \begin{pmatrix} x_{i,m} & x_{i,m-1} & \cdots & x_{i,m-(q-1)} \\ x_{i,2m} & x_{i,2m-1} & \cdots & x_{i,2m-(q-1)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{i,Tm} & x_{i,Tm-1} & \cdots & x_{i,Tm-(q-1)} \end{pmatrix} \quad w_i = \begin{pmatrix} w_{i1} \\ w_{i2} \\ \vdots \\ w_{iq} \end{pmatrix} \quad \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_T \end{pmatrix} \tag{3.13}$$

Assume now that the weights $w_1, \cdots, w_n$ are known (i.e. the parameters $\theta_1, \cdots, \theta_n$ are known). Then defining the tranformed regressors $\tilde{X}_i = w_i X_i$, equation (3.12) can be reformulated as a standard linear regression:

$$y = X\delta + \varepsilon \tag{3.14}$$

with:

$$X = \begin{pmatrix} 1 & Y & \tilde{X}_1 & \tilde{X}_2 & \cdots & \tilde{X}_n \end{pmatrix} \qquad \delta = \begin{pmatrix} c \\ \gamma \\ \beta_1 \\ \vdots \\ \beta_n \end{pmatrix} \qquad \tilde{X}_i = X_i \, w_i \tag{3.15}$$

where δ is the vector of unknown coefficients that remain to be estimated. Assuming normality of the residuals as in (3.7), the log likelihood of model (3.14) is proportional to:

$$\mathcal{L} \propto -0.5 \, (y - X\delta)' \, (y - X\delta) \tag{3.16}$$

This likelihood function is the so-called "profile likelihood" of Ghysels and Qian (2016), owing its name to the fact that it is done conditional on $w_1, \cdots, w_n$. Maximizing (3.16) yields the standard OLS estimator:

$$\hat{\delta} = (X'X)^{-1}(X'y) \tag{3.17}$$

Substituting $\hat{\delta}$ back in the likelihood function (3.16) yields:

$$\mathcal{L}_\theta \propto -0.5\ (y - X(X'X)^{-1}(X'y))'\ (y - X(X'X)^{-1}(X'y)) \tag{3.18}$$

where the notation $\mathcal{L}_\theta$ stresses the fact the likelihood (3.18) does not depend on δ anymore, but only on $\theta = \{\theta_1, \cdots, \theta_n\}$ through X. It is then straightforward that the value of θ that maximizes (3.18) also maximizes the likelihood of the model, so that a maximum likelihood estimator of θ can obtain as:

$$\hat{\theta} = \underset{\theta}{argmax} \left(-0.5\ (y - X(X'X)^{-1}(X'y))'\ (y - X(X'X)^{-1}(X'y))\right) \tag{3.19}$$

which can simplify further into:

$$\hat{\theta} = \underset{\theta}{argmax} \left(y'X(X'X)^{-1}X'y\right) \tag{3.20}$$

(3.17) and (3.20) thus provide an efficient algorithm to estimate the MIDAS model:

Algorithm 2: MIDAS regression

1. Use a numerical solver to find $\hat{\theta} = \underset{\theta}{argmax} \left(y'X(X'X)^{-1}X'y\right)$.

2. Once θ is known, calculate X, then compute $\hat{\delta} = (X'X)^{-1}(X'y)$.

The advantage of this approach compared to brute strength estimation of (3.11) is that the numerical part of the algorithm is here reduced to θ instead of all the parameters. The space on which the solver has to work is thus smaller, which increases the chance of success.

3.2.3 Prediction

The MIDAS model is primarily intended for prediction purposes. Assume one wants to produce a one-step ahead prediction. This can be obtained by updating (3.8) by one period to obtain:

$$y_{t+1} = \mu + \sum_{j=1}^{p} \gamma_i\, y_{t+1-j} + \sum_{i=1}^{n} \beta_i \Big(\sum_{j=1}^{q} w_{ij}\, x_{i,(t+1)m+1-j} \Big) + \varepsilon_{t+1} \tag{3.21}$$

(3.21) requires knowledge of the predictions $x_{i,t+1}, x_{i,t+2}, \cdots, x_{i,(t+1)m}$, which may be difficult or impossible to obtain. For this reason, Ghysels et al. (2016) proposes a more direct approach

to forecasting in the context of MIDAS models. Define the h-step ahead MIDAS regression as:

$$y_{t+h} = \mu + \sum_{j=1}^{p} \gamma_i \, y_{t+1-j} + \sum_{i=1}^{n} \beta_i \Big(\sum_{j=1}^{q} w_{ij} \, x_{i,tm+1-j} \Big) + \varepsilon_{t+h} \tag{3.22}$$

Taking expectations on both sides of (3.22), a prediction $\hat{y}_{t+h}$ can easily be obtained from known feature values up to period t:

$$\hat{y}_{t+h} = \mu + \sum_{j=1}^{p} \gamma_i \, y_{t+1-j} + \sum_{i=1}^{n} \beta_i \Big(\sum_{j=1}^{q} w_{ij} \, x_{i,tm+1-j} \Big) \tag{3.23}$$

With this method, the model becomes specific to the prediction h-steps ahead. In other words, one must now estimate a different model for each forecast horizon h.

Estimating the h-step ahead MIDAS regression is similar to the regular MIDAS model, except the matrices of regressors are now defined as:

$$Y = \begin{pmatrix} y_{1-h} & y_{1-h-1} & \cdots & y_{1-h-(p-1)} \\ y_{2-h} & y_{2-h-1} & \cdots & y_{2-h-(p-1)} \\ \vdots & \vdots & \ddots & \vdots \\ y_{T-h} & y_{T-h-1} & \cdots & y_{T-h-(p-1)} \end{pmatrix} \quad X_i = \begin{pmatrix} x_{i,m(1-h)} & x_{i,m(1-h)-1} & \cdots & x_{i,m(1-h)-(q-1)} \\ x_{i,m(2-h)} & x_{i,m(2-h)-1} & \cdots & x_{i,m(2-h)-(q-1)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{i,m(T-h)} & x_{i,m(T-h)-1} & \cdots & x_{i,m(T-h)-(q-1)} \end{pmatrix} \tag{3.24}$$

3.2.4 Application

Besides forecasts, the MIDAS regression can also be used to assess the relative importance of each high frequency feature in predicting the low frequency variable, through examination of the lag structure. To do so, a one-step ahead MIDAS regression is estimated on the reduced macroeconomic dataset of the project. The benchmark used is $q = 6$ high frequency lags, with a corresponding $p = 2$ low frequency lags.

Figure 3.5 reports the lag structure of the MIDAS model, for both the Almon and the Beta lag polynomial approaches. The features are normalised to avoid potential scale effects.

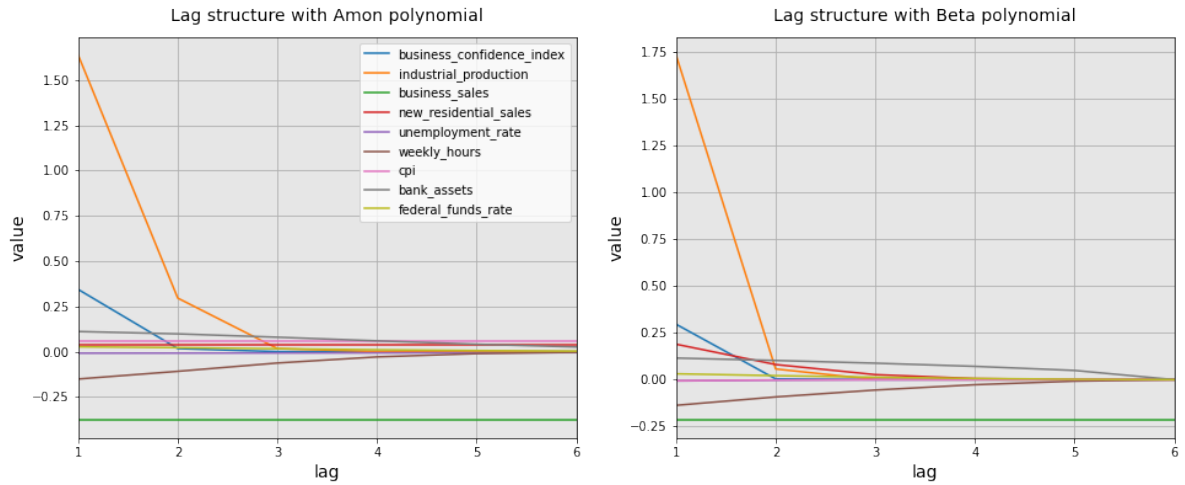


Figure 3.5: Amon and Beta lag polynomials for fixed θ_1 values

The two approaches yield fairly consistent estimates, with only small differences in magnitude observed. The main lesson from the plot is that over the 9 features included in the model, only 4 seem to have a strong predictive power on quarterly GDP: industrial production, the business confidence index, business sales, and weekly hours. The other variables contribute only marginally to GDP growth, and their lags are flat overall.

The strong positive correlations with industrial production and business confidence index makes sense, as the former is an important component of GDP, while the latter acts as a direct proxy. The negative correlation with business sales and weekly hours looks more surprising. However, the model considered is a one-step ahead MIDAS model. This may thus simply be the reflect that higher business sales and weekly hours today imply less production one quarter ahead, in an overshooting fashion.

3.3 Nowcasting: mixed frequency Bayesian VAR

Mixed frequency Bayesian VARs have been proposed recently as an extension to regular Bayesian VAR models. The aim is to account for datasets at mixed frequencies, in the hope of improving nowcasting performances. The reference paper is Schorfheide and Song (2015), and the developments below follow closely their methodologies. Because mixed frequency Bayesian VAR operate as an augmentation to regular Bayesian VARs, readers unfamiliar with the Bayesian VAR approach are advised to refer first to section 3.5 where they are covered in details.

3.3.1 Formulation

Assume one wants to estimate a VAR with both monthly and quarterly series. There are m monthly series and q quarterly series, implying a total of $n = m + q$ series in the VAR model. At each sample period $t = 1, \cdots, T$, the obervation y_t can thus be separated into its monthly component $y_{m,t}$ and its quarterly component $y_{q,t}$. At quaterly months (months at which quarterly release occur, e.g. March, June, September and December for GDP), $y_{q,t}$ is of dimension q and thus $y_t = \{y_{m,t}, y_{q,t}\}$ is of dimension n. The other months, $y_{q,t} = \varnothing$ (there is no release and thus no quarterly data for this month), so that $y_t = \{y_{m,t}\}$ is of dimension m only.

A VAR model requires a panel of series at a single frequency. It is thus assumed further that there exists a series of unobserved monthly components x_t for $t = 1, \cdots, T$ on which the VAR model will be estimated. x_t is an n-dimensional vector divided into $x_t = \{x_{m,t}, x_{q,t}\}$, with $x_{m,t}$ an m-dimensional component corresponding to the monthly series, and $x_{q,t}$ a q-dimensional component corresponding to the partially unobserved quarterly series. Clearly, $x_{m,t} = y_{m,t}$. However, $x_{q,t} \neq y_{q,t}$: the latter is a partial quarterly dataset, while the former represents its full unobserved monthly counterpart.

The VAR model is assumed to exist for the monthly series x_t. It is formulated as:

$$x_t = c + A_1 x_{t-1} + \cdots + A_p x_{t-p} + \varepsilon_t \qquad \varepsilon_t \sim N(0, \Sigma) \tag{3.25}$$

Denote by x the full series $\{x_1, x_2, \cdots, x_T\}$. Also, stack all the VAR coefficients in a single

vector β, as in section 3.5. Estimating the model then consists in estimating the unknown parameters x, β and Σ.

3.3.2 Estimation

The Bayesian approach consists in obtaining the posterior distributions of x, β and Σ from a basic application of Bayes rule:

$$\pi(x, \beta, \Sigma|y) \propto f(y|x)\pi(x|\beta, \Sigma)\pi(\beta)\pi(\Sigma) \tag{3.26}$$

$\pi(x, \beta, \Sigma|y)$ denotes the joint posterior distribution for x, β and Σ. $f(y|x)$ is the likelihood function. Because β and Σ become redundant once x is determined, they can be omitted from the latter which hence writes $f(y|x)$ and not $f(y|x, \beta, \Sigma)$. $\pi(x|\beta, \Sigma)$, $\pi(\beta)$ and $\pi(\Sigma)$ represent the joint prior $\pi(x, \beta, \Sigma)$. Indeed, one can note that:

$$\pi(x, \beta, \Sigma) = \frac{\pi(x, \beta, \Sigma)}{\pi(\beta, \Sigma)}\pi(\beta, \Sigma) = \pi(x|\beta, \Sigma)\pi(\beta, \Sigma) = \pi(x|\beta, \Sigma)\pi(\beta)\pi(\Sigma) \tag{3.27}$$

where the final term obtains from the independance assumption between β and Σ. It is worth noting on the other hand that x is not independant from β and Σ. This is clearly implied by equation (3.27) and justifies the conditional prior $\pi(x|\beta, \Sigma)$. Such a prior where one parameter of the model is conditional on other model parameters is called a hierarachical prior.

The prior distributions for β and Σ are similar to those used in section 3.5 and are thus respectively: $\pi(\beta) \sim N(\beta_0, \Omega_0)$ and $\pi(\Sigma) \sim IW(S_0, \nu_0)$, with respective densities:

$$\pi(\beta) = (2\pi)^{-nq/2}|\Omega_0|^{-1/2} \exp\left[-\frac{1}{2}(\beta - \beta_0)'\Omega_0^{-1}(\beta - \beta_0)\right] \tag{3.28}$$

and:

$$\pi(\Sigma) = (2^{\nu_0 n/2}\Gamma_n(\nu_0/2))^{-1}|S_0|^{\nu_0/2}|\Sigma|^{-(\nu_0+n+1)/2} \exp\left[-\frac{1}{2}tr\{\Sigma^{-1}S_0\}\right] \tag{3.29}$$

For x, one notes that conditional on β and Σ, equation (3.25) is similar to the VAR model (3.50). Following, the prior for x is also multivariate normal, with density function given by (3.54):

$$\pi(x|\beta, \Sigma) = (2\pi)^{-nT/2}|\bar{\Sigma}|^{-1/2} \exp\left[-\frac{1}{2}(x - \bar{W}\beta)'\bar{\Sigma}^{-1}(x - \bar{W}\beta)\right] \tag{3.30}$$

with:

$$x = vec(X) \qquad X = \begin{pmatrix} x_1' \\ x_2' \\ \vdots \\ x_T' \end{pmatrix} \qquad \bar{\Sigma} = \Sigma \otimes I_T \qquad \bar{W} = I_n \otimes W \qquad W = \begin{pmatrix} 1 & x_0 & \cdots & x_{1-p} \\ 1 & x_1 & \cdots & x_{2-p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{T-1} & \cdots & x_{T-p} \end{pmatrix} \tag{3.31}$$

The joint posterior (3.26) is clearly intractable (it cannot be marginalised for x, β and Σ analytically) so the use of the Gibbs sampling algorithm is required. The conditional posteriors for β and Σ are fairly straightforward. For β, start from the joint posterior (3.26) and relegate to the normalizing constant any term not involving β. This yields $\pi(\beta|y,x,\Sigma) \propto \pi(x|\beta,\Sigma)\pi(\beta)$. Given (3.28) and (3.30), the resulting expression after manipulations is similar to (3.61):

$$\pi(\beta|y,x,\Sigma) \propto \exp\left[-\frac{1}{2}(\beta-\bar{\beta})'\bar{\Omega}^{-1}(\beta-\bar{\beta})\right] \tag{3.32}$$

with:

$$\bar{\Omega} = \left(\Omega_0^{-1} + \Sigma^{-1} \otimes W'W\right)^{-1} \qquad \bar{\beta} = \bar{\Omega}\left(\Omega_0^{-1}\beta_0 + (\Sigma^{-1} \otimes W')x\right) \tag{3.33}$$

(3.32) is the kernel of a multivariate normal distribution: $\pi(\beta|y,x,\Sigma) \sim N(\bar{\beta},\bar{\Omega})$.

Similarly, the conditional posterior $\pi(\Sigma|y,x,\beta)$ obtains from (3.26) by relegating to the normalization constant any term not involving Σ: $\pi(\Sigma|y,x,\beta) \propto \pi(x|\beta,\Sigma)\pi(\Sigma)$. Given (3.29) and (3.30), and some manipulations, the result is similar to (3.64):

$$\pi(\Sigma|y,x,\beta) \propto |\Sigma|^{(\bar{\nu}+n+1)/2} \exp\left[-\frac{1}{2}tr\{\Sigma^{-1}\bar{S}\}\right] \tag{3.34}$$

with:

$$\bar{S} = (X - WB)'(X - WB) + S_0 \qquad \bar{\nu} = T + \nu_0 \tag{3.35}$$

(3.35) is the kernel of an inverse Wishart distribution: $\pi(\Sigma|y,x,\beta) \sim IW(\bar{S},\bar{\nu})$.

The conditional posterior $\pi(x|y,\beta,\Sigma)$ for x is the most difficult to estimate. Starting from (3.26) and relegating to the normalization constant any term not involving x yields $\pi(x|y,\beta,\Sigma) \propto f(y|x)\pi(x|\beta,\Sigma)$. However, the x_t are hidden state variables which makes the formula impossible to apply directly. The solution in this case has been proposed by Carter and Kohn (1994).

It consists in formulating the conditional posterior $\pi(x|y,\beta,\Sigma)$ in state-space form, then use a Kalman filter procedure to recover the distribution.

To start with, reformulate the dynamic equation (3.25) in companion form:

$$\begin{pmatrix} x_t \\ x_{t-1} \\ \vdots \\ x_{t-p+1} \end{pmatrix} = \begin{pmatrix} c \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \begin{pmatrix} A_1 & \cdots & A_{p-1} & A_p \\ I_n & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & I_n & 0 \end{pmatrix} \begin{pmatrix} x_{t-1} \\ x_{t-2} \\ \vdots \\ x_{t-p} \end{pmatrix} + \begin{pmatrix} \varepsilon_t \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{3.36}$$

More compactly:

$$z_t = \delta + \Phi\, z_{t-1} + v_t \qquad\qquad v_t \sim N(0,\Omega) \tag{3.37}$$

with:

$$z_t = \begin{pmatrix} x_t \\ x_{t-1} \\ \vdots \\ x_{t-p+1} \end{pmatrix} \quad \delta = \begin{pmatrix} c \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad \Phi = \begin{pmatrix} A_1 & \cdots & A_{p-1} & A_p \\ I_n & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & I_n & 0 \end{pmatrix} \quad v_t = \begin{pmatrix} \varepsilon_t \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad \Omega = \begin{pmatrix} \Sigma & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix} \tag{3.38}$$

The simplest way to represent the link between the observed variables y_t and the unobserved components x_t is by the way of selection matrices:

$$y_t = \Lambda_t\, z_t \tag{3.39}$$

For quarterly months, we have simply the identity $y_t = x_t$. Hence:

$$\Lambda_t = \begin{pmatrix} I_n & 0_{n,n(p-1)} \end{pmatrix} \tag{3.40}$$

For non-quarterly months, we have $y_t = x_{m,t}$, which results in the definition:

$$\Lambda_t = \begin{pmatrix} I_m & 0_{m,q} & 0_{m,n(p-1)} \end{pmatrix} \tag{3.41}$$

(3.37) and (3.39) represent, respectively, the state equation and observation equation of a state-space system. It is then possible to derive the conditional posterior distribution $\pi(x|y,\beta,\Sigma)$ from the Carter and Kohn (1994) algorithm. The details of the algorithm can be found in Appendix A.3.

The Gibbs sampling algorithm for the mixed frequency Bayesian VAR then goes as follows:

Algorithm 3: Gibbs sampling for the mixed frequency Bayesian VAR

1. Initiate $\beta^{(0)}$, $\Sigma^{(0)}$ and $x^{(0)}$ with any values. In practice, use $x_{m,t} = y_{m,t}$ for the high frequency data, and define $x_{q,t} = y_{q,t}$, with replications of the last observed value for the missing entries. Then set $\beta^{(0)} = \hat{\beta}$ and $\Sigma^{(0)} = \hat{\Sigma}$, where the values are obtained from OLS estimation of (3.25) with $x^{(0)}$. Also decide the total number of repetitions of the algorithm (say r=2000 for instance), and the number of initial iterations to discard (for instance d = 1000) to ensure convergence.

2. Draw $\beta^{(1)}$ from the conditional distribution $\pi(\beta^{(1)}|y,\Sigma^{(0)},x^{(0)}) \sim N(\bar{\beta},\bar{\Omega})$.

3. Draw $\Sigma^{(1)}$ from the conditional distribution $\pi(\Sigma^{(1)}|y,\beta^{(1)},x^{(0)}) \sim IW(\bar{S},\bar{\nu})$.

4. Draw $x^{(1)}$ from the conditional distribution $\pi(x^{(1)}|y,\beta^{(1)},\Sigma^{(1)})$, using the Carter-Kohn algorithm.

5. Repeat r times:
draw $\beta^{(n)}$ from $\pi(\beta^{(n)}|y,\Sigma^{(n-1)},x^{(n-1)}) \sim N(\bar{\beta},\bar{\Omega})$.
draw $\Sigma^{(n)}$ from the conditional distribution $\pi(\Sigma^{(n)}|y,\beta^{(n)},x^{(n-1)}) \sim IW(\bar{S},\bar{\nu})$.
draw $x^{(n)}$ from $\pi(x^{(n)}|y,\beta^{(n)},\Sigma^{(n)})$, using the Carter-Kohn algorithm.

6. Discard d initial observations as burn-in sample to make sure convergence is achieved. Then the remaining values are draws from the unconditional posteriors $\pi(\beta|y)$, $\pi(\Sigma|y)$ and $\pi(x|y)$, which can be used to recover an empirical distribution.

3.3.3 Prediction

The MCMC algorithm for predictions is similar to that described in section 3.5:

Algorithm 4: prediction for the mixed frequency Bayesian VAR

2. draw a value β from $\pi(\beta,\Sigma,x|y)$. Recycle a draw from the Gibbs sampling algorithm.

3. draw a value Σ from $\pi(\beta,\Sigma,x|y)$. Recycle a draw from the Gibbs sampling algorithm.

1. draw a value x from $\pi(\beta,\Sigma,x|y)$. Recycle a draw from the Gibbs sampling algorithm.

4. conditional on β, Σ, and x, compute recursively $x_{t+1},\cdots,x_{t+h}$ from (3.25).

5. convert $x_{t+1},\cdots,x_{t+h}$ into $z_{t+1},\cdots,z_{t+h}$.

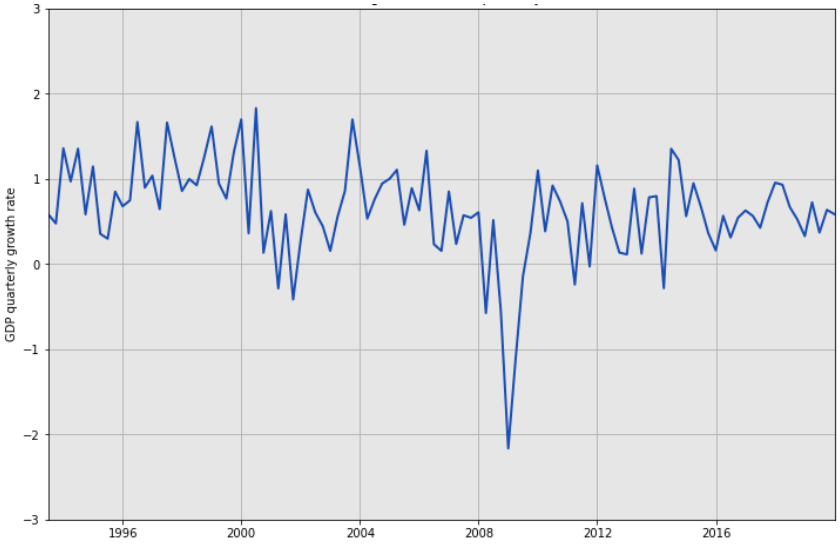
6. compute $y_{t+1},\cdots,y_{t+h}$ from $z_{t+1},\cdots,z_{t+h}$, using (3.39).

7. discard x, β and Σ.

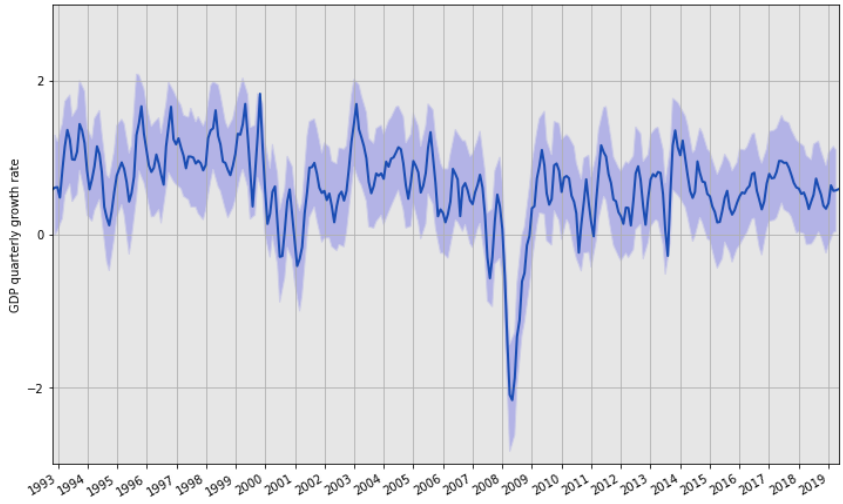
8. repeat the process $r-d$ times to obtain $r-d$ draws from the posterior predictive distribution.

3.3.4 Application

As a short application, it is interesting to visualize the actual, quarterly series of GDP growth along with the monthly estimates provided by the model. Figure 3.6 provides an overview of the results.



(a) Actual quarterly values



(b) Monthly estimates with 95% credibility bands

Figure 3.6: Actual values and monthly estimates for GDP growth

One can see that the monthly series look quite accurate. This is demonstrated by the tight credibility bands around the median of the posterior. Comparing the two series, they appear to be quite similar. Looking closely at the monthly series however, it looks a bit more jagged than the quarterly series, showing the short term monthly deviations from the quarterly trend. Overall, this graphic reflects well on the ability of the model to infer missing values from the information provided by the rest of the dataset.

3.4 Econometrics: Vector Auto-Regression

Vector autoregressions (VAR in short) have become popular since the contribution of Sims (1980). In the field of econometrics, they represent the state-of-the-art approach to model and predict time-series. VAR models can be very simple, like the one developed in this section, or adopt more sophisticated formulations and estimation methods, as demonstrated by the incoming econometric sections.

3.4.1 Formulation

Assume one wants to model n variables jointly. This can be done by the way of a VAR model, which is simply a system of n linear equations. Each equation explains the current value of one feature by its own lagged values, and the lagged values of the other features of the system. Concretely, a standard VAR model can be written as:

$$y_t = c + A_1 y_{t-1} + \cdots + A_p y_{t-p} + \varepsilon_t \qquad \varepsilon_t \sim N(0,\Sigma) \tag{3.42}$$

where y_t is a n-dimensional vector of endogenous variables, c is a constant term, the series $A_1 + \cdots + A_p$ represent a series of $n \times n$ matrices of autoregressive coefficients, and ε_t is a White noise error term with variance-covariance matrix Σ. The sample is considered over the time periods $t = 1, \cdots, T$.)

For estimation purposes, it is convenient to rewrite the model in compact form as:

$$Y = XB + E \tag{3.43}$$

with:

$$Y = \begin{pmatrix} y_1' \\ y_2' \\ \vdots \\ y_T' \end{pmatrix} \qquad X = \begin{pmatrix} 1 & y_0 & \cdots & y_{1-p} \\ 1 & y_1 & \cdots & y_{2-p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & y_{T-1} & \cdots & y_{T-p} \end{pmatrix} \qquad B = \begin{pmatrix} c' \\ A_1' \\ \vdots \\ A_p' \end{pmatrix} \qquad E = \begin{pmatrix} \varepsilon_1' \\ \varepsilon_2' \\ \vdots \\ \varepsilon_T' \end{pmatrix} \tag{3.44}$$

3.4.2 Estimation

Given (3.42) and (3.43), estimating a VAR amounts to obtain estimates for Σ and B. (3.43) can be recognised as a standard linear regression model that can be estimated by least squares. Therefore, a least square estimate $\hat{B}$ of B is given by:

$$\hat{B} = (X'X)^{-1}X'Y \tag{3.45}$$

Once $\hat{B}$ is obtained, one may obtain an estimate of the residuals from (3.43) as $\hat{E} = Y - X\hat{B}$. A (degree of freedom adjusted) OLS estimate for Σ is then given by:

$$\hat{\Sigma} = \frac{1}{T-k-1}(\hat{E}'\hat{E}) \tag{3.46}$$

where $k = np+1$ denotes the number of parameters to estimate per equation.

3.4.3 Prediction

Predicting with a VAR model is straightforward. Assume one wants to predict at the horizon $t+h$. First, update (3.42) by one period and take expectation conditional to the information set up to period t to obtain:

$$\hat{y}_{t+1} = c + A_1 y_t + \cdots + A_p y_{t-(p-1)} \tag{3.47}$$

To obtain a prediction at $t+2$, update instead (3.42) by two periods and take again conditional expectation to obtain:

$$\hat{y}_{t+2} = c + A_1 \hat{y}_{t+1} + \cdots + A_p y_{t-(p-2)} \tag{3.48}$$

where it can be seen that the prediction $\hat{y}_{t+2}$ makes use of the projection $\hat{y}_{t+1}$. Continuing sequentially up to period $t+h$ yields:

$$\hat{y}_{t+h} = c + A_1 \hat{y}_{t+h-1} + \cdots + A_p \hat{y}_{t+h-p} \tag{3.49}$$

3.5 Econometrics: Bayesian VAR

Bayesian VAR models have first been proposed by Doan et al. (1984). By the time, only very simple Bayesian models could be estimated due to the loack of computing power, so that the interest in Bayesian econometrics declined in the 1980's. From the 2000's on, the calculation capacities of computers started to increase exponentially, making it possible to use computationally intensive Monte Carlo Markov Chain algorithms. This marked the rebirth of Bayesian VAR models, which have since then attracted increasing interest. The presentation in this section follows the treatment of Karlsson (2012).

3.5.1 Formulation

The VAR model developed in this section is similar to the standard VAR:

$$y_t = c + A_1 y_{t-1} + \cdots + A_p y_{t-p} + \varepsilon_t \qquad \varepsilon_t \sim N(0,\Sigma) \tag{3.50}$$

It is again convenient to reformulate the model in compact form as:

$$y = \bar{X}\beta + \varepsilon \qquad \varepsilon \sim N(0,\bar{\Sigma}) \tag{3.51}$$

with:

$$y = vec(Y) \qquad Y = \begin{pmatrix} y_1' \\ y_2' \\ \vdots \\ y_T' \end{pmatrix} \qquad \bar{X} = I_n \otimes X \qquad X = \begin{pmatrix} 1 & y_0 & \cdots & y_{1-p} \\ 1 & y_1 & \cdots & y_{2-p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & y_{T-1} & \cdots & y_{T-p} \end{pmatrix}$$

$$\beta = vec(B) \qquad B = \begin{pmatrix} c' \\ A_1' \\ \vdots \\ A_p' \end{pmatrix} \qquad \varepsilon = vec(E) \qquad E = \begin{pmatrix} \varepsilon_1' \\ \varepsilon_2' \\ \vdots \\ \varepsilon_T' \end{pmatrix} \qquad \bar{\Sigma} = \Sigma \otimes I_T \tag{3.52}$$

3.5.2 Estimation

Given (3.51), estimating the VAR model comes down to estimating the two parameters β and Σ.

The Bayesian approach consists in obtaining the posterior distributions of these two terms, from a basic application of Bayes rule:

$$\pi(\beta,\Sigma|y) = \frac{f(y|\beta,\Sigma)\pi(\beta,\Sigma)}{f(y)} \propto f(y|\beta,\Sigma)\pi(\beta,\Sigma) \tag{3.53}$$

where $\pi(\beta,\Sigma|y)$ denotes the joint posterior distribution for β and Σ, $f(y|\beta,\Sigma)$ is the likelihood function, $\pi(\beta,\Sigma)$ is the joint prior distribution, and $f(y)$ is the marginal likelihood. In practice, the latter is relegated to the normalizing constant of the density as it does not involve β or Σ.

The normality of the residuals in (3.51) implies that the likelihood function considered jointly for the T time periods is multivariate normal:

$$f(y|\beta,\Sigma) = (2\pi)^{-nT/2}|\bar{\Sigma}|^{-1/2}\exp\left[-\frac{1}{2}(y-\bar{X}\beta)'\bar{\Sigma}^{-1}(y-\bar{X}\beta)\right] \tag{3.54}$$

For the prior, assuming independence between β and Σ implies that $\pi(\beta,\Sigma) = \pi(\beta)\pi(\Sigma)$. So the joint prior can be expressed as the product of marginal priors, which makes things easier.

For β, a natural choice consists in assuming a multivariate normal distribution: $\pi(\beta) \sim N(\beta_0,\Omega_0)$. Therefore, the prior density for β is given by:

$$\pi(\beta) = (2\pi)^{-nq/2}|\Omega_0|^{-1/2}\exp\left[-\frac{1}{2}(\beta-\beta_0)'\Omega_0^{-1}(\beta-\beta_0)\right] \tag{3.55}$$

To determine the values of β_0 and Ω_0, Litterman (1986) proposes the so-called Minnesota prior. For the prior mean β_0, the Minnesota prior states that most economic time-series are characterised by a random walk. Therefore, the VAR coefficients should take a value of 1 for each variable on its own first lag, and 0 otherwise. For instance, on a simple 2-variable VAR with 2 lags, this yields:

$$\begin{pmatrix} y_{1,t} \\ y_{2,t} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} y_{1,t-1} \\ y_{2,t-2} \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} y_{1,t-2} \\ y_{2,t-2} \end{pmatrix} + \begin{pmatrix} \varepsilon_{1,t} \\ \varepsilon_{2,t} \end{pmatrix} \quad \Rightarrow \quad \beta_0 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \tag{3.56}$$

In practice, when stationary data is used, a value around 0.9 is prefered to 1 to induce stationarity.

For the prior covariance Ω_0, the Minnesota prior postulates a diagonal matrix (no prior covariance between coefficients), with additional shrinkage for coefficients on further lags and related to other variables. This gives three cases:

1. for coefficients in β relating endogenous variables to their own lags, the variance is given by: $(\lambda_1/l^{\lambda_3})^2$, with λ_1 an overall tightness parameter, and λ_3 a lag shrinkage parameter (l designates the lag).

2. for coefficients relating variable i to variable j, the variance is given by: $(\sigma_i^2/\sigma_j^2)(\lambda_1\lambda_2/l^{\lambda_3})^2$, where σ_i^2 and σ_j^2 denote the OLS residual variance of auto-regressive models estimated for variables i and j, and λ_2 represents a cross-variable shrinkage parameter.

3. for the constants, the variance is given by: $\sigma_i^2(\lambda_1\lambda_4)^2$, with λ_4 a constant-specific shrinkage coefficient.

For the above simple VAR, Ω_0 is thus given by:

$$\Omega_0 = \begin{pmatrix}
\sigma_1^2(\lambda_1\lambda_4)^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & (\lambda_1)^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{\sigma_1^2}{\sigma_2^2}(\lambda_1\lambda_2)^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & (\frac{\lambda_1}{2^{\lambda_3}})^2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{\sigma_1^2}{\sigma_2^2}(\frac{\lambda_1\lambda_2}{2^{\lambda_3}})^2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \sigma_2^2(\lambda_1\lambda_4)^2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{\sigma_2^2}{\sigma_1^2}(\lambda_1\lambda_2)^2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & (\lambda_2)^2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\sigma_2^2}{\sigma_1^2}(\frac{\lambda_1\lambda_2}{2^{\lambda_3}})^2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & (\frac{\lambda_1}{2^{\lambda_3}})^2
\end{pmatrix} \tag{3.57}$$

As a guideline, Litterman (1986) proposes the following hyperparameter values: $\lambda_1 = 0.1$, $\lambda_2 = 0.5$, $\lambda_3 = 2$ and $\lambda_4 = 100$.

Finally, for Σ, a classical prior choice is an inverse Wishart distribution with scale S_0 and degrees of freedom ν_0. The density is therefore given by:

$$\pi(\Sigma) = (2^{\nu_0 n/2}\Gamma_n(\nu_0/2))^{-1}|S_0|^{\nu_0/2}|\Sigma|^{-(\nu_0+n+1)/2}\exp\left[-\frac{1}{2}tr\{\Sigma^{-1}S_0\}\right] \tag{3.58}$$

Classical choices (see e.g. Karlsson (2012)) for S_0 and ν_0 consist in setting $S_0 = diag(\sigma_1^2 \cdots \sigma_n^2)$ and $\nu_0 = n+2$.

Then from Bayes rule (3.53), the likelihood function (3.54), and the priors (3.55) and (3.58), the kernel of the joint posterior is given by:

$$\begin{aligned}
&\pi(\beta,\Sigma|y) \\
&\propto|\bar{\Sigma}|^{-1/2}\exp\left[-\frac{1}{2}(y-\bar{X}\beta)'\bar{\Sigma}^{-1}(y-\bar{X}\beta)\right] \\
&\times\exp\left[-\frac{1}{2}(\beta-\beta_0)'\Omega_0^{-1}(\beta-\beta_0)\right] \\
&\times|\Sigma|^{-(\nu_0+n+1)/2}\exp\left[-\frac{1}{2}tr\{\Sigma^{-1}S_0\}\right]
\end{aligned} \tag{3.59}$$

(3.59) is not exploitable as such since it is a joint distribution. In theory, one could obtain the marginal posteriors as $\pi(\beta|y) = \int\pi(\beta,\Sigma|y)d\Sigma$ and $\pi(\Sigma|y) = \int\pi(\beta,\Sigma|y)d\beta$. In practice however, β and Σ are so interwoven in (3.59) that it is not possible to evaluate the integrals analytically. To obtain the posteriors, it is then necessary to rely on simulation methods.

The Gibbs sampling algorithm represents the simplest solution. It belongs to the class of Monte Carlo Markov Chain (MCMC) algorithms, and relies on the convergence properties of Markov chains. In spirit, its principle is quite simple: draw the parameters in turn from their conditional posterior distributions, and repeat the process a large number of times. After a sufficient number of draws, the algorithm converges to the unconditional posteriors, so that an empirical distribution of the unconditional posteriors can be constructed.

Indeed, if it is not possible to compute the unconditional marginals $\pi(\beta|y)$ and $\pi(\Sigma|y)$, it is straightforward to obtain the conditional posteriors $\pi(\beta|y,\Sigma)$ and $\pi(\Sigma|y,\beta)$. For β for instance, note that $\pi(\beta|y,\Sigma) = \pi(\beta,\Sigma|y)/\pi(\Sigma|y) \propto \pi(\beta,\Sigma|y)$. In other words, to obtain the conditional posterior $\pi(\beta|y,\Sigma)$, one starts from the joint posterior (3.59) and relegates to the normalization constant any term not involving β. Applying this, the conditional posterior obtains as:

$$\pi(\beta|y,\Sigma) \propto \exp\left[-\frac{1}{2}(y-\bar{X}\beta)'\bar{\Sigma}^{-1}(y-\bar{X}\beta)\right] \times \exp\left[-\frac{1}{2}(\beta-\beta_0)'\Omega_0^{-1}(\beta-\beta_0)\right] \tag{3.60}$$

After some manipulations, this rewrites as:

$$\pi(\beta|y,\Sigma) \propto \exp\left[-\frac{1}{2}(\beta-\bar{\beta})'\bar{\Omega}^{-1}(\beta-\bar{\beta})\right] \tag{3.61}$$

with:

$$\bar{\Omega} = \left(\Omega_0^{-1} + \Sigma^{-1} \otimes X'X\right)^{-1} \qquad \bar{\beta} = \bar{\Omega}\left(\Omega_0^{-1}\beta_0 + (\Sigma^{-1} \otimes X')y\right) \tag{3.62}$$

(3.61) is the kernel of a multivariate normal distribution: $\pi(\beta|y,\Sigma) \sim N(\bar{\beta},\bar{\Omega})$.

Similarly, the conditional posterior $\pi(\Sigma|y,\beta)$ obtains from (3.59) by relegating to the normalization constant any term not involving Σ:

$$\begin{aligned}\pi(\Sigma|y,\beta) \propto |\bar{\Sigma}|^{-1/2}\exp&\left[-\frac{1}{2}(y-\bar{X}\beta)'\bar{\Sigma}^{-1}(y-\bar{X}\beta)\right] \\ &\times |\Sigma|^{-(v_0+n+1)/2}\exp\left[-\frac{1}{2}tr\{\Sigma^{-1}S_0\}\right]\end{aligned} \tag{3.63}$$

After some manipulations, this rewrites as:

$$\pi(\Sigma|y,\beta) \propto |\Sigma|^{(\bar{\nu}+n+1)/2} \exp\left[-\frac{1}{2} tr\{\Sigma^{-1}\bar{S}\}\right] \tag{3.64}$$

with:

$$\bar{S} = (Y - XB)'(Y - XB) + S_0 \qquad \bar{\nu} = T + \nu_0 \tag{3.65}$$

(3.64) is the kernel of an inverse Wishart distribution: $\pi(\Sigma|y,\beta) \sim IW(\bar{S},\bar{\nu})$.

The Gibbs sampling algorithm then goes as follows:

Algorithm 5: Gibbs sampling for the Bayesian VAR

1. Initiate $\beta^{(0)}$ and $\Sigma^{(0)}$ with any values. In practice, the OLS estimates $\hat{\beta}$ and $\hat{\Sigma}$ are often used. Also decide the total number of repetitions of the algorithm (say r=2000 for instance), and the number of initial iterations to discard (for instance d = 1000) to ensure convergence.

2. Draw $\beta^{(1)}$ from the conditional distribution $\pi(\beta^{(1)}|y,\Sigma^{(0)}) \sim N(\bar{\beta},\bar{\Omega})$.

3. Draw $\Sigma^{(1)}$ from the conditional distribution $\pi(\Sigma^{(1)}|y,\beta^{(1)}) \sim IW(\bar{S},\bar{\nu})$.

4. Repeat r times:
draw $\beta^{(n)}$ from $\pi(\beta^{(n)}|y,\Sigma^{(n-1)}) \sim N(\bar{\beta},\bar{\Omega})$.
draw $\Sigma^{(n)}$ from the conditional distribution $\pi(\Sigma^{(n)}|y,\beta^{(n)}) \sim IW(\bar{S},\bar{\nu})$.

5. Discard d initial observations as burn-in sample to make sure convergence is achieved. Then the remaining values are draws from the unconditional posteriors $\pi(\beta|y)$ and $\pi(\Sigma|y)$, which can be used to recover an empirical distribution.

3.5.3 Prediction

With the empirical posterior distribution recovered, it is easy to estimate predictions. Consider for instance a h-step ahead prediction for the model. Estimating the predictions amounts to obtaining the so-called the posterior predictive density $f(y_{t+1},\cdots,y_{t+h}|y_t)$.

Noting that this density can be expressed as:
$f(y_{T+1},\cdots,y_{T+h}|y_T) = \int f(y_{T+1},\cdots,y_{T+h}|\beta,\Sigma,y_T)\pi(\beta,\Sigma|y_T)d\beta,\Sigma,$
one can see that the posterior predictive distribution rewrites as an (integrated) product of two distributions: the posterior distribution, and the distribution of future observations, conditional on data and parameter values. This suggests a natural procedure to obtain draws from the predictive density. First, sample values for β and Σ from the posterior $\pi(\beta,\Sigma|y_T)$. This is done trivially by recycling the draws from the Gibbs sampling algorithm. Then, conditionally on these draws, generate values from $f(y_{T+1},\cdots,y_{T+h}|\beta,\Sigma,y_T)$, which is done easily by applying recursively (3.50). Finally, marginalise (compute the integral over β and θ) by simply discarding the values of β and Σ.

The MCMC algorithm for predictions is then given by:

Algorithm 6: prediction for the Bayesian VAR

1. draw a value β from $\pi(\beta,\Sigma|y)$. Recycle a draw from the Gibbs sampling algorithm.

2. draw a value Σ from $\pi(\beta,\Sigma|y)$. Recycle a draw from the Gibbs sampling algorithm.

3. conditional on β and Σ, compute recursively $y_{t+1},\cdots,y_{t+h}$ from (3.50).

4. discard β and Σ.

5. repeat the process $r-d$ times to obtain $r-d$ draws from the posterior predictive distribution.

3.5.4 Application

To stress the difference between a regular VAR and a Bayesian VAR, it can be informative to examine the coefficients of the model. To do so, the two models are estimated on the reduced dataset for the project. The benchmark is realised on 2 lags, and Minnesota parameter values of $\rho = 0.85, \lambda_1 = 0.4, \lambda_2 = 0.3, \lambda_3 = 1.5, \lambda_4 = 1000$.

Table 3.1 reports the VAR coefficients for the GDP equation of both models.

	Lag 1		Lag 2	
	OLS VAR	Bayesian VAR	OLS VAR	Bayesian VAR
business confidence index	0.103	0.007	0.050	0.004
industrial production	0.095	0.044	0.042	0.003
business sales	-0.005	0.001	-0.041	-0.002
new residential sales	-0.001	0.002	0.003	0.002
unemployment rate	0.025	0.005	-0.037	-0.006
weekly hours	-0.087	-0.008	-0.058	-0.014
cpi	-0.015	-0.018	0.052	0.009
bank assets	-0.005	-0.004	0.026	0.007
federal funds rate	-0.013	0.024	0.091	-0.011
gdp	-0.082	0.046	0.056	0.159

Table 3.1: VAR coefficients of GDP equation

The effects of the Minnesota prior on the Bayesian VAR are clearly visible. The coefficients on the lags of all variables but GDP are smaller on the Bayesian VAR than on the OLS VAR, reflecting the effect of the λ_2 hyperparameter in (3.57). The effect is even more pronounced on the second lag, which results from the additional shrinkage generated by the hyperparameter λ_3.

As a compensation, the Bayesian VAR attributes considerably more weight to the own lags of GDP. The first lag is smaller than its OLS counterpart in magnitude, but it now takes a positive sign. As for the second lag, its value is considerably larger than that of the OLS model.

Overall, the results illustrate the impact of the Minnesota prior: more weight is given to the variables's own lags, while less weight is granted to the other variables and to further lags. This parsimonious representation typically performs better for out-of-sample predictions than the standard OLS estimate.

3.6 Econometrics: Time-varying Bayesian VAR

In a context of rapidly changing economic dynamics, static VAR models can be incapable of capturing the fast evolutions of the unerlying process and hence produce suboptimal forecasts. For this reason, a natural extention to VAR models consists in integrating dynamics in the model parameter themselves. The major contributions in the field are the papers by Primiceri (2005) and Del Negro and Primiceri (2015). The presentation in this section follows the equation-by-equation approach proposed by Legrand (2019), which improves on both the calculation speed and prediction accuracy.

3.6.1 Formulation

A general time-varying VAR model can be written as:

$$y_t = c_t + A_{1,t} y_{t-1} + \cdots + A_{p,t} y_{t-p} + \varepsilon_t \qquad \varepsilon_t \sim N(0, \Sigma_t) \tag{3.66}$$

This model is similar to (3.50), except that the parameters $c_t, A_{1,t}, \cdots, A_{p,t}$ and Σ_t are not allowed to be period-specific. Stacking in a vector β_t the set of VAR coefficients, (3.66) rewrites:

$$y_t = X_t \beta_t + \varepsilon_t \tag{3.67}$$

with:

$$X_t = I_n \otimes x_t \ , \ x_t = \begin{pmatrix} z_t' & y_{t-1}' & \cdots & y_{t-p}' \end{pmatrix} \ , \ \beta_t = vec(B_t) \ , \ B_t = \begin{pmatrix} C_t & A_{1,t} & \cdots & A_{p,t} \end{pmatrix}' \tag{3.68}$$

Considering specifically row i of (3.67) , the equation for variable i of the model rewrites:

$$y_{i,t} = x_t \beta_{i,t} + \varepsilon_{i,t} \tag{3.69}$$

where $\beta_{i,t}$ is the $k \times 1$ vector obtained from column i of B_t. Stacking (3.69) over the T sample periods yields a full sample formulation for equation i:

$$y_i = X \beta_i + \varepsilon_i \tag{3.70}$$

with:

$$y_i = \begin{pmatrix} y_{i,1} \\ y_{i,2} \\ \vdots \\ y_{i,T} \end{pmatrix} \quad , \quad X = \begin{pmatrix} x_1 & 0 & \cdots & 0 \\ 0 & x_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & x_T \end{pmatrix} \quad , \quad \beta_i = \begin{pmatrix} \beta_{i,1} \\ \beta_{i,2} \\ \vdots \\ \beta_{i,T} \end{pmatrix} \quad , \quad \varepsilon_i = \begin{pmatrix} \varepsilon_{i,1} \\ \varepsilon_{i,2} \\ \vdots \\ \varepsilon_{i,T} \end{pmatrix} \tag{3.71}$$

The variance-covariance matrix Σ_t for the reduced form residuals is decomposed into:

$$\Delta_t \Sigma_t \Delta_t' = \Lambda_t \qquad \Leftrightarrow \qquad \Sigma_t = \Delta_t^{-1} \Lambda_t \Delta_t^{-1\prime} \tag{3.72}$$

Δ_t (and Δ_t^{-1}) are unit lower triangular matrix, while Λ_t is a diagonal matrix with positive diagonal entries, taking the form:

$$\Delta_t = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ \delta_{21,t} & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ \delta_{n1,t} & \cdots & \delta_{n(n-1),t} & 1 \end{pmatrix} , \Lambda_t = \begin{pmatrix} s_1\, exp(\lambda_{1,t}) & 0 & \cdots & 0 \\ 0 & s_2\, exp(\lambda_{2,t}) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & s_n\, exp(\lambda_{n,t}) \end{pmatrix} \tag{3.73}$$

The triangular decomposition of the variance-covariance matrix Σ_t implemented in (3.72) is common in time-series models. Λ_t represents the volatility components of Σ_t, each s_i being a positive scaling term which represents the equilibrium value of the residual variance of equation i of the model. On the other hand, Δ_t can be interpreted as the (inverse) covariance component of Σ_t. Denoting by $\delta_{i,t}$ the vector of non-zero and non-one terms in row i of Δ_t so that $\delta_{i,t} = (\delta_{i1,t} \quad \cdots \quad \delta_{i(i-1),t})'$, $\delta_{i,t}$ then represents the covariance between the residual of equation i of the model and the other shocks.

The dynamics of the model time-varying parameters is specified as follows:

$$
\begin{aligned}
&\beta_{i,t} = (1-\rho_i)b_i + \rho_i\beta_{i,t-1} + \xi_{i,t} && t = 2,3,\ldots,T && \xi_{i,t} \sim \mathcal{N}(0,\Omega_i)\\
&\beta_{i,1} = b_i + \xi_{i,1} && t = 1 && \xi_{i,1} \sim \mathcal{N}(0,\tau\Omega_i)\\
&\lambda_{i,t} = \gamma_i\lambda_{i,t-1} + \nu_{i,t} && t = 2,3,\ldots,T && \nu_{i,t} \sim \mathcal{N}(0,\phi_i)\\
&\lambda_{i,1} = \nu_{i,1} && t = 1 && \nu_{i,1} \sim \mathcal{N}(0,\mu\phi_i)\\
&\delta_{i,t} = (1-\alpha_i)d_i + \alpha_i\delta_{i,t-1} + \eta_{i,t} && t = 2,3,\ldots,T && \eta_{i,t} \sim \mathcal{N}(0,\Psi_i)\\
&\delta_{i,1} = d_i + \eta_{i,1} && t = 1 && \eta_{i,1} \sim \mathcal{N}(0,\epsilon\,\Psi_i)
\end{aligned}
\tag{3.74}
$$

ρ_i, γ_i and α_i represent equation-specific autoregressive coefficients while b_i, s_i and d_i represent the equation-specific mean values of the processes. These values are exogenous hyperparameters sepcified by the user. For each process, the initial period is formulated consistently with the overall dynamics of the parameters. The mean corresponds to the unconditional expectation of the process, while the variance is scaled by the hyperparameters $\tau,\mu,\epsilon > 1$ to account for the greater uncertainty associated with the initial period. All the innovations in the model are assumed to be jointly normally distributed with the following assumptions on the variance covariance matrix:

$$
Var\begin{pmatrix}\varepsilon_t\\ \xi_{i,t}\\ \nu_{i,t}\\ \eta_{i,t}\end{pmatrix} = \begin{pmatrix}\Sigma_t & 0 & 0 & 0\\ 0 & \Omega_i & 0 & 0\\ 0 & 0 & \phi_i & 0\\ 0 & 0 & 0 & \Psi_i\end{pmatrix} \tag{3.75}
$$

3.6.2 Estimation

For $i = 1,\ldots,n$, the parameters of interest to be estimated are: the dynamic VAR coefficients β_i ; the dynamic volatility terms λ_i; the dynamic covariance terms δ_i; and the associated variance-covariance parameters Ω_i, ϕ_i and Ψ_i. To these six base parameters must be added the parameter $r_{i,t}$ whose role will be clarified shortly. Given the model, Bayes rule is given by:

$$
\begin{aligned}
&\pi(\beta,\Omega,\lambda,\phi,\delta,\Psi,r|y) \propto f(y|\beta,\lambda,\delta,r)\\
&\times \left(\prod_{i=1}^{n}\pi(\beta_i|\Omega_i)\pi(\Omega_i)\right)\left(\prod_{i=1}^{n}\pi(\lambda_i|\phi_i)\pi(\phi_i)\right)\left(\prod_{i=2}^{n}\pi(\delta_i|\Psi_i)\pi(\Psi_i)\right)\left(\prod_{i=1}^{n}\prod_{t=1}^{T}\pi(r_{i,t})\right)
\end{aligned}
\tag{3.76}
$$

From (3.67), an immediate formulation of the likelihood function obtains as:

$$f(y|\beta,\lambda,\delta,r) = \prod_{t=1}^{T}(2\pi)^{-n/2}|\Sigma_t|^{-1/2}exp\left(-\frac{1}{2}(y_t - X_t\beta_t)'\Sigma_t^{-1}(y_t - X_t\beta_t)\right) \tag{3.77}$$

The priors for β_i, λ_i and δ_i are fully defined by their dynamic equations in (3.74).

The prior distributions for the variance parameters Ω_i, ϕ_i and Ψ_i are standard inverse Wishart and inverse Gamma distributions. For Ω_i, the prior is inverse Wishart with degrees of freedom ζ_0 and scale Υ_0:

$$\pi(\Omega_i) \sim IW(\zeta_0, \Upsilon_0) \tag{3.78}$$

Following:

$$\pi(\Omega_i) = \frac{2^{-\zeta_0 k/2}}{\Gamma_k\left(\frac{\zeta_0}{2}\right)}|\Upsilon_0|^{\zeta_0/2}|\Omega_i|^{-(\zeta_0+k+1)/2}exp\left(-\frac{1}{2}tr\left\{\Omega_i^{-1}\Upsilon_0\right\}\right) \tag{3.79}$$

The prior distribution for each ϕ_i is inverse gamma with shape $\frac{\kappa_0}{2}$ and scale $\frac{\omega_0}{2}$:

$$\pi(\phi_i) \sim IG\left(\frac{\kappa_0}{2}, \frac{\omega_0}{2}\right) \tag{3.80}$$

Hence:

$$\pi(\phi_i) = \frac{\frac{\omega_0}{2}^{\kappa_0/2}}{\Gamma(\frac{\kappa_0}{2})}\phi_i^{-\kappa_0/2-1}exp\left(-\frac{\omega_0}{2\phi_i}\right) \tag{3.81}$$

Finally, the prior distribution for Ψ_i is inverse Wishart with degrees of freedom φ_0 and scale Θ_0:

$$\pi(\Psi_i) \sim IW(\varphi_0, \Theta_0) \tag{3.82}$$

Following:

$$\pi(\Psi_i) = \frac{2^{-\varphi_0(i-1)/2}}{\Gamma_{(i-1)}\left(\frac{\varphi_0}{2}\right)}|\Theta_0|^{\varphi_0/2}|\Psi_i|^{-(\varphi_0+(i-1)+1)/2}exp\left(-\frac{1}{2}tr\left\{\Psi_i^{-1}\Theta_0\right\}\right) \tag{3.83}$$

Bayes rule (3.76) is intractable, and requires the use of MCMC methods. As usual, the Gibbs sampling algorithm is used, relying on the conditional posterior. From Bayes rule (3.76), one obtains that the conditional posterior $\pi(\beta_i|y,\backslash\beta_i)$ is given by $\pi(\beta_i|y,\backslash\beta_i) \propto f(y|\beta,\lambda,\delta,r)\pi(\beta_i|\Omega_i)$. Starting from (3.77), and after some manipulations, the likelihood function $f(y|\beta,\lambda,\delta,r)$ reformulates as:

$$y_{i,t} + \delta'_{i,t}\varepsilon_{-i,t} = x_t\beta_{i,t} + e_{i,t} \qquad e_{i,t} \sim \mathcal{N}(0, s_i\, exp(\lambda_{i,t})) \tag{3.84}$$

where $\varepsilon_{-i,t} = (\varepsilon_{1,t}\cdots\varepsilon_{i-1,t})$. (3.84) and (3.74) respectively provide the observation and state equations for the dynamic parameter β_i. The conditional posterior $\pi(\beta_i|y,\backslash\beta_i)$ can then be recovered from the Carter-Kohn algorithm (see Appendix A.3 for the details of the algorithm).

From Bayes rule (3.76), one obtains that the conditional posterior $\pi(\delta_i|y,\backslash\delta_i)$ is given by $\pi(\delta_i|y,\backslash\delta_i) \propto f(y|\beta,\lambda,\delta,r)\pi(\delta_i|\Psi_i)$. Starting from (3.77), and after some manipulations, the likelihood function $f(y|\beta,\lambda,\delta,r)$ reformulates as:

$$\varepsilon_{i,t} = -\varepsilon'_{-i,t}\delta_{i,t} + e_{i,t} \qquad e_{i,t} \sim \mathcal{N}(0, s_i\, exp(\lambda_{i,t})) \tag{3.85}$$

(3.85) and (3.74) respectively provide the observation and state equations for the dynamic parameter δ_i. The conditional posterior $\pi(\delta_i|y,\backslash\delta_i)$ can then be recovered from the Carter-Kohn algorithm.

From Bayes rule (3.76), one obtains that the conditional posterior $\pi(\lambda_i|y,\backslash\lambda_i)$ is given by $\pi(\lambda_i|y,\backslash\lambda_i) \propto f(y|\beta,\lambda,\delta,r)\pi(\lambda_i|\phi_i)$. As it is, the likelihood function (3.77) is not workable due to the exponential terms in (3.73). Kim et al. (1998) thus propose to approximate the likelihood with a mixture of 7 normal distributions, where the mixture component is determined by the categorical random variable $r_{i,t}$ taking value $j = 1,2,\cdots,7$. The likelihood function then rewrites as:

$$\hat{y}_{i,t} - m_j = \lambda_{i,t} + \hat{e}_{i,t} \qquad \hat{e}_{i,t} \sim \mathcal{N}(0, v_j) \tag{3.86}$$

where $\hat{y}_{i,t} = \log(s_i^{-1}[\varepsilon_{i,t} + \varepsilon'_{-i,t}\delta_{i,t}])$ and m_j and v_j denote the respective mean and variance of the normal distribution when the random variable $r_{i,t}$ selects the mixture component j. (3.86) and (3.74) respectively provide the observation and state equations for the dynamic parameter δ_i. The conditional posterior $\pi(\delta_i|y,\backslash\delta_i)$ can then be recovered from the Carter-Kohn algorithm.

From Bayes rule (3.76), one obtains that the conditional posterior $\pi(r_{i,t}|y,\backslash r_{i,t})$ is given by $\pi(r_{i,t}|y,\backslash r_{i,t}) \propto f(y|\beta,\lambda,\delta,r)\pi(r_{i,t})$. Given that $r_{i,t}$ is a categorical random variable, and some rearrangements of the approximated $f(y|\beta,\lambda,\delta,r)$, one eventually obtains:

$$\pi(r_{i,t}|y,\backslash r_{i,t}) \propto \bar{q}_j^{\mathbb{1}(r_{i,t}=j)} \tag{3.87}$$

with:

$$\bar{q}_j = (2\pi v_j)^{-1/2} \exp\left(-\frac{1}{2}\frac{(\hat{y}_{i,t}-\lambda_{i,t}-m_j)^2}{v_j}\right) q_j \tag{3.88}$$

This is the kernel of a categorical distribution with probabilities $\bar{q}_1,\bar{q}_2,\ldots,\bar{q}_7$: $\pi(r_{i,t}|y,\backslash r_{i,t}) \sim Cat(\bar{q}_1,\bar{q}_2,\ldots,\bar{q}_7)$.

The conditional posteriors for the variance terms are standard. For Ω_i, start from Bayes rule (3.76) and relegate to the normalising constant any multiplicative term not involving Ω_i to obtain:

$$\pi(\Omega_i|y,\backslash\Omega_i) \propto \pi(\beta_i|\Omega_i)\pi(\Omega_i) \tag{3.89}$$

Given the priors (3.74) and (3.79) and rearranging yields:

$$\pi(\Omega_i|y,\backslash\Omega_i) = |\Omega_i|^{-(\bar{\zeta}+k+1)/2} exp\left(-\frac{1}{2} tr\left\{\Omega_i^{-1}\bar{\Upsilon}_i\right\}\right) \tag{3.90}$$

with:

$$\begin{aligned}
&\bar{\zeta} = T + \zeta_0 \qquad \bar{\Upsilon}_i = \tilde{B}_i + \Upsilon_0 \\
&\tilde{B}_i = (B_i - 1_T' \otimes b_i)\,(F_i'\, I_{-\tau}\, F_i)(B_i - 1_T' \otimes b_i)' \qquad B_i = \begin{pmatrix}\beta_{i,1} & \beta_{i,2} & \cdots & \beta_{i,T}\end{pmatrix} \\
&F_i = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ -\rho_i & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & -\rho_i & 1 \end{pmatrix} \qquad I_{-\tau} = \begin{pmatrix} \tau^{-1} & 0 & \cdots & 0 \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}
\end{aligned} \tag{3.91}$$

This is the kernel of an inverse Wishart distribution with degrees of freedom $\bar{\zeta}$ and scale $\bar{\Upsilon}_i$: $\pi(\Omega_i|y,\backslash\Omega_i) \sim IW(\bar{\zeta},\bar{\Upsilon}_i)$

For ϕ_i, start from Bayes rule (3.76) and relegate to the normalising constant any multiplicative term not involving ϕ_i to obtain:

$$\pi(\phi_i|y,\backslash\phi_i) \propto \pi(\lambda_i|\phi_i)\pi(\phi_i) \tag{3.92}$$

Given the priors (3.74) and (3.81) and rearranging yields:

$$\pi(\phi_i|y,\backslash\phi_i) \propto \phi_i^{-\bar{\kappa}-1} exp\left(-\frac{\bar{\omega}_i}{\phi_i}\right) \tag{3.93}$$

with:

$$\begin{aligned}
&\bar{\kappa} = \frac{T+\kappa_0}{2} \qquad \bar{\omega}_i = \frac{\lambda_i'(G_i' I_{-\mu} G_i)\lambda_i + \omega_0}{2} \\
&G_i = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ -\gamma_i & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & -\gamma_i & 1 \end{pmatrix} \qquad I_{-\mu} = \begin{pmatrix} \mu^{-1} & 0 & \cdots & 0 \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}
\end{aligned} \tag{3.94}$$

This is the kernel of an inverse Gamma distribution with shape $\bar{\kappa}$ and scale $\bar{\omega}_i$: $\pi(\phi_i|y,\backslash\phi_i) \sim IG(\bar{\kappa},\bar{\omega}_i)$

Finally, for Ψ_i, start from Bayes rule (3.76) and relegate to the normalising constant any multiplicative term not involving Ψ_i to obtain:

$$\pi(\Psi_i|y,\backslash\Psi_i) \propto \pi(\delta_i|\Psi_i)\pi(\Psi_i) \tag{3.95}$$

Given the priors (3.74) and (3.83) and rearranging yields:

$$\pi(\Psi_i|y,\backslash\Psi_i) \propto |\Psi_i|^{-(\bar{\varphi}+(i-1)+1)/2} exp\left(-\frac{1}{2} tr\left\{\Psi_i^{-1}\bar{\Theta}_i\right\}\right) \tag{3.96}$$

with:

$$\begin{aligned}
&\bar{\varphi} = T + \varphi_0 \qquad \bar{\Theta}_i = \tilde{D}_i + \Theta_0 \\
&\tilde{D}_i = (D_i - 1_T' \otimes d_i)\,(H_i'\, I_{-\epsilon}\, H_i)(D_i - 1_T' \otimes d_i)' \qquad D_i = \begin{pmatrix} \delta_{i,1} & \delta_{i,2} & \cdots & \delta_{i,T} \end{pmatrix} \\
&H_i = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ -\alpha_i & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & -\alpha_i & 1 \end{pmatrix} \qquad I_{-\epsilon} = \begin{pmatrix} \epsilon^{-1} & 0 & \cdots & 0 \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}
\end{aligned} \tag{3.97}$$

This is the kernel of an inverse Wishart distribution with degrees of freedom $\bar{\varphi}$ and scale $\bar{\Theta}_i$: $\pi(\Psi_i|y,\backslash\Psi_i) \sim IW(\bar{\varphi},\bar{\Theta}_i)$

It is then possible to describe the full Gibbs sampling algorithm for the time-varying Bayesian VAR model:

Algorithm 7: Gibbs sampling for the time-varying Bayesian VAR

1. Initiate $\beta_i^{(0)},\lambda_i^{(0)},\delta_i^{(0)},\Omega_i^{(0)},\phi_i^{(0)},\delta_i^{(0)}$ and $r_{i,t}^{(0)}$ with any values, for $i = 1,\cdots,n$ and $t = 1,\cdots,T$. In practice, use static OLS estimates for each variable and each period. Also decide the total number of repetitions of the algorithm (say r=2000 for instance), and the number of initial iterations to discard (for instance d = 1000) to ensure convergence.

2. For $i = 1,\ldots,n$, sample $\lambda_i^{(n)}$ equation by equation, using the Carter-Kohn algorithm.

3. For $i = 1,\ldots,n$, sample $\beta_i^{(n)}$ equation by equation, using the Carter-Kohn algorithm.

4. For $i = 2,\ldots,n$, sample $\delta_i^{(n)}$ equation by equation, using the Carter-Kohn algorithm.

5. For $i = 1,\ldots,n$, sample $\Omega_i^{(n)}$ equation by equation from: $\pi(\Omega_i^{(n)}|y,\backslash\Omega_i) \sim IW(\bar{\zeta},\bar{\Upsilon}_i)$.

6. For $i = 1,\ldots,n$, sample $\phi_i^{(n)}$ equation by equation from: $\pi(\phi_i^{(n)}|y,\backslash\phi_i) \sim IG(\bar{\kappa},\bar{\omega}_i)$.

7. For $i = 2,\ldots,n$, sample $\Psi_i^{(n)}$ equation by equation from: $\pi(\Psi_i^{(n)}|y,\backslash\Psi_i) \sim IW(\bar{\varphi},\bar{\Theta}_i)$.

8. For $i = 1,\ldots,n$ and $t = 1,\ldots,T$, sample $r_{i,t}^{(n)}$ from: $\pi(r_{i,t}^{(n)}|y,\backslash r_{i,t}) \sim Cat(\bar{q}_1,\ldots,\bar{q}_7)$.

9. Repeat r times.

10. Discard d initial observations as burn-in sample to make sure convergence is achieved. Then the remaining values are draws from the unconditional posteriors $\pi(\beta|y)$ and $\pi(\Sigma|y)$, which can be used to recover an empirical distribution.

3.6.3 Prediction

The MCMC algorithm for predictions is given by:

Algorithm 8: prediction for the time-varying Bayesian VAR

1. for $i = 1, \cdots, n$, draw values for $\lambda_i, \delta_i, \phi_i$ and Ψ_i from their conditional posteriors. Recycle draws from the Gibbs sampling algorithm.

2. for $i = 1, \cdots, n$, obtain values for $\lambda_{i,t+1}, \delta_{i,t+1}, \cdots, \lambda_{i,t+h}, \delta_{i,t+h}$ from (3.74). Recover $\Sigma_{t+1}, \cdots, \Sigma_{t+h}$ from (3.72) .

3. for $i = 1, \cdots, n$, draw values for β_i and Ω_i from their conditional posteriors. Recycle draws from the Gibbs sampling algorithm.

4. for $i = 1, \cdots, n$, obtain values for $\beta_{i,t+1}, \cdots, \beta_{i,t+h}$ from (3.74).

5. for $i = 1, \cdots, n$, compute recursively $y_{i,t+1}, \cdots, y_{i,t+h}$ from (3.66).

6. repeat the process $r - d$ times to obtain $r - d$ draws from the posterior predictive distribution.

3.6.4 Application

As an illustration, the estimates of the time-varying variances for the feature shocks are provided in Figure 3.7.

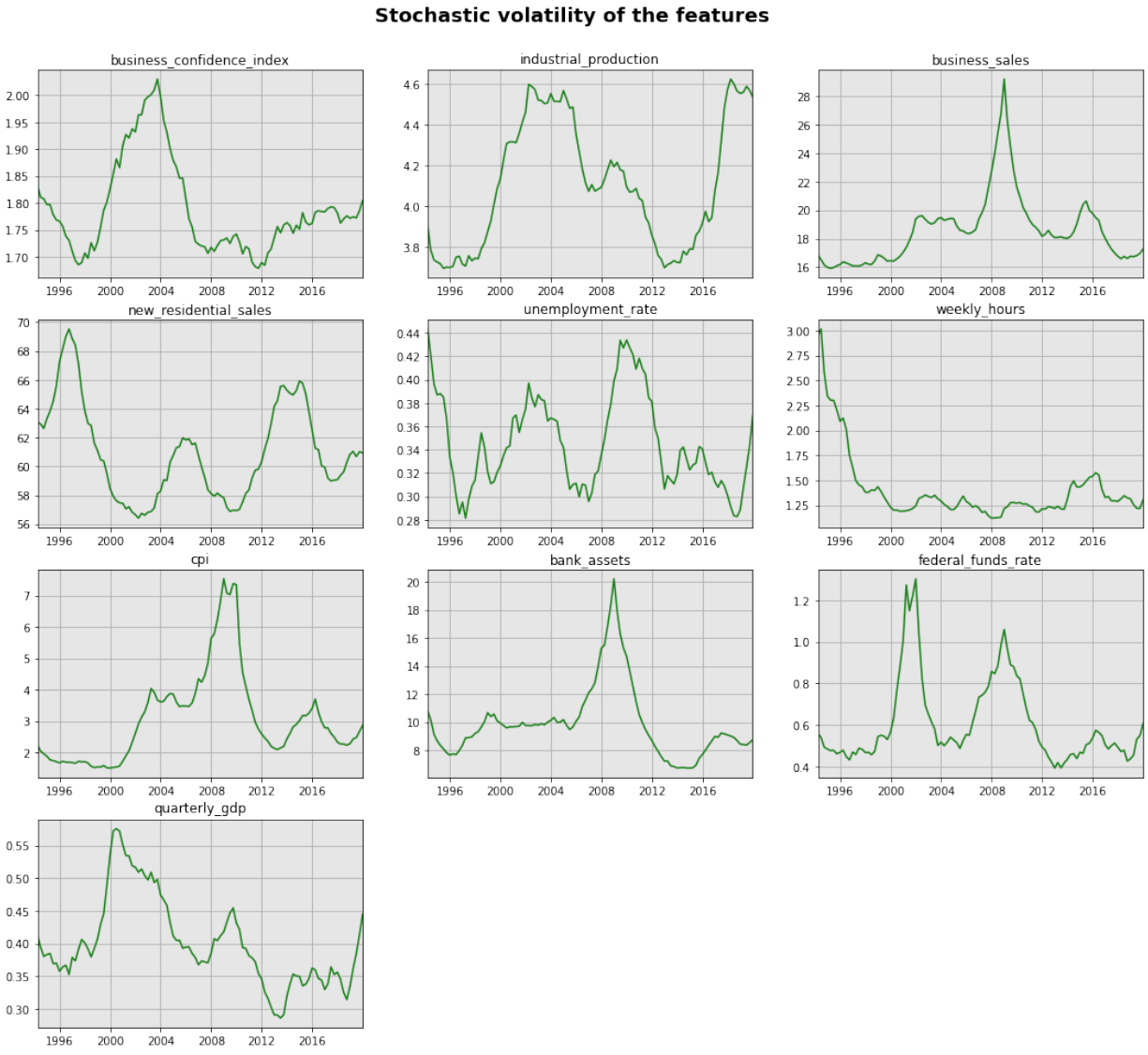


Figure 3.7: Estimates of the time-varying feature variances

The 2008 crisis is well pictured in many series, with a fueling in volatility (business sales, unemployment rate, bank assets, federal funds rate, quarterly GDP) over the period. Interestingly enough, the volatility in residential sales plumetted with the crisis, perhaps the sign that activity got stuck at a very low level over the period. Still worth noting, the model seems capable of anticipating the current crisis, with a rise in volatility of many key features over 2019, in particular industrial production, unemployment, the federal funds rate, and quarterly GDP.

3.7 Machine learning: LSTM

In the class of machine learning models, neural networks can be traced back to the 1950's with the contribution of Rosenblatt (1958). Since the 2010's, there has been renewed interest in neural networks thanks to the advent ot GPUs and distributed computing, leading to a booming in the field of deep learning.

Recurrent neural networks (RNN in short) are based on the work of Hopfield (1982). They constitute a special class of neural networks that explicitely account for temporal dynamic behavior. A variant of RNN models known as Long Short Term Memory models (LSTM in short) has been proposed by Hochreiter and Schmidhuber (1997). LSTM proved extremely effective for tasks like speech recognition and image captioning, but they also represent powerful alternatives for financial time-series data.

3.7.1 Formulation

A standard neural network is a model organised around layers: an input layer, several (possibly none) hidden layers, and an output layer. This is illustrated in Figure 3.8.

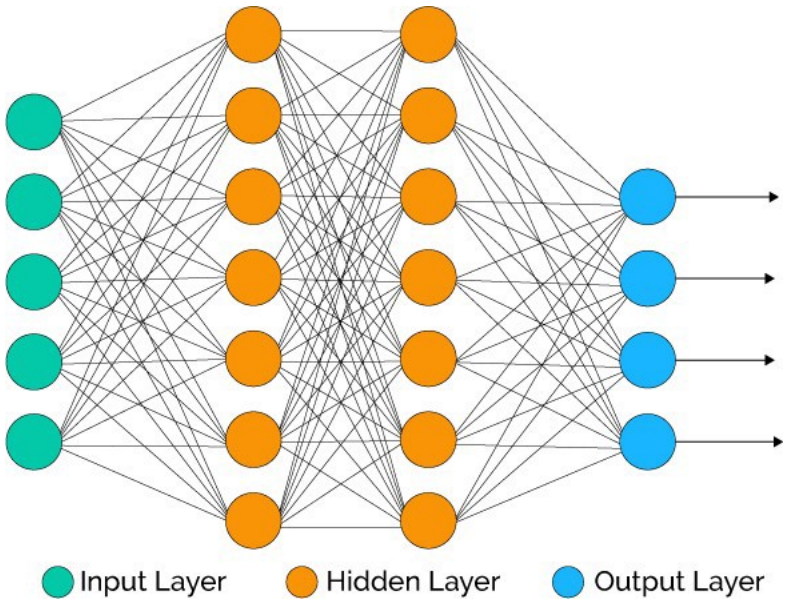


Figure 3.8: A stylised neural network (credit: Conor McDonald)

At each layer, the output inherited from the previous layer goes through some linear combination, then some nonlinearity is applied to transform the result, before it is channeled to the next layer. Formally, denoting by x the initial input, by a_i the output of layer i (for hidden layer $1, \cdots, m$), and by $\hat{y}$ the output layer, the model writes as the following sequence:

$$\begin{aligned} a_1 &= g(b_1 + xW_1) \\ a_2 &= g(b_2 + a_1 W_2) \\ &\vdots \\ a_m &= g(b_m + a_{m-1} W_m) \\ \hat{y} &= h(a_m) \end{aligned} \tag{3.98}$$

where b_i and w_i denote respectively vectors of bias and matrices of coefficients. g(.) is some function that generates the non-linearity at each layer. It used to be traditionally a sigmoid function, but today the most popular alternative is the so-called ReLU function (standing for Rectifed Linear Unit) defined as $g(x) = max(0,x)$. The output function h conditions the final layer output a_m for the output layer. Popular choices are the softmax function $g(x) = e^x / \sum e^x$ for classification, or the linear function $g(x) = x$ for regression.

Stacking for all n sample observations in matrix X, the model rewrites in compact form as the sequence:

$$\begin{aligned} A_1 &= g(1_n \otimes b_1 + XW_1) \\ A_2 &= g(1_n \otimes b_2 + A_1 W_2) \\ &\vdots \\ A_m &= g(1_n \otimes b_m + A_{m-1} W_m) \\ \hat{Y} &= h(A_m) \end{aligned} \tag{3.99}$$

where the A_i and $\hat{Y}$ now denote matrices of outputs.

Regular neural networks can be modified to account explicitely for the timely structure of the data. This results in the so-called recurrent neural network, illustrated in Figure 3.9.

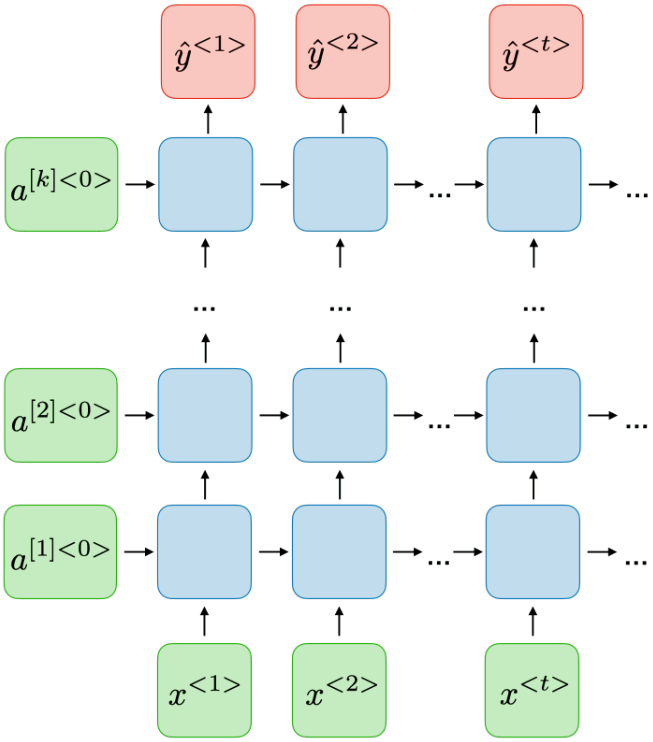


Figure 3.9: A stylised recurrent neural network (credit: Shervine Amidi)

As can be seen from the graph, at each layer the RNN works not only with the output of the previous layer, but also with the ouput of the same layer at previous period. Using a subscript t to denote the time period of the sample, with $t = 1, \cdots, T$, the model rewrites as:

$$
\begin{aligned}
a_{1,t} &= g(b_1 + x_t W_1 + x_{t-1} Z_1) \\
a_{2,t} &= g(b_2 + a_{1,t} W_2 + a_{2,t-1} Z_2) \\
&\vdots \\
a_{m,t} &= g(b_m + a_{m-1,t} W_m + a_{m,t-1} Z_m) \\
\hat{y}_t &= h(a_{m,t})
\end{aligned}
\tag{3.100}
$$

A more sophisticated version of the RNN is provided by the LSTM model. In standard RNNs, each cell only produces a single activation through the function $g(.)$. LSTM models by contrast provide a more alaborate structure at each cell level, as illustrated by Figure 3.10.

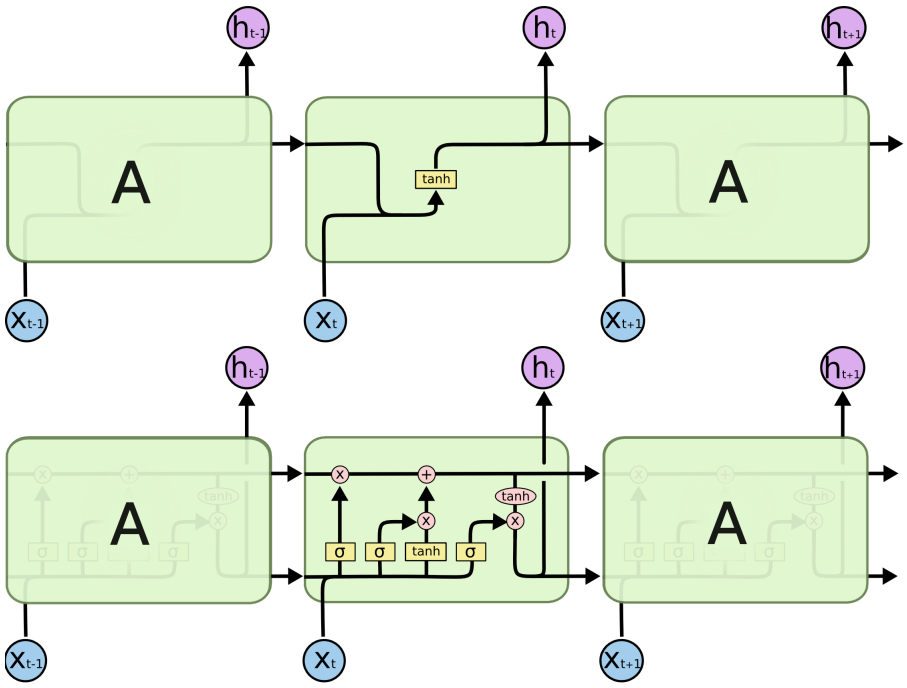


Figure 3.10: RNN cell (top) and LSTM cell (bottom) (credit: Christopher Olah)

Each cell in an LSTM has four kinds of activations or "gates": a forget gate that decides what information inherited from previous period should be thrown away or kept; an input gate that processess the output received from the previous layer; and an output gate that calculates the cell state from the first two gates and decides the output to be delivered to the next layer. The set of equations describing the LSTM cell for layer j at period t is then given by:

$$
\begin{aligned}
f_{j,t} &= \sigma(b^f_j + a_{j-1,t}W^f_j + a_{j,t-1}Z^f_j) && \text{(forget gate)} \\
i_{j,t} &= \sigma(b^i_j + a_{j-1,t}W^i_j + a_{j,t-1}Z^i_j) && \text{(input gate)} \\
o_{j,t} &= \sigma(b^o_j + a_{j-1,t}W^o_j + a_{j,t-1}Z^o_j) && \text{(output gate)} \\
\tilde{c}_{j,t} &= tanh(b^c_j + a_{j-1,t}W^c_j + a_{j,t-1}Z^c_j) && \text{(update gate)} \\
c_{j,t} &= f_{j,t-1} \odot c_{j,t-1} + i_{j,t} \odot \tilde{c}_{j,t} && \text{(state vector)} \\
a_{j,t} &= o_{j,t} \odot tanh(c_{j,t}) && \text{(cell output)}
\end{aligned}
\tag{3.101}
$$

where $\sigma(.)$ and $tanh(.)$ respectively denote the sigmoid and hyperbolic tangeant functions.

3.7.2 Estimation

LSTM models are estimated by minimizing a loss function which acts as a measure of fit of the model to observed data. In the case of LSTM models for regression with multivariate outputs, the loss function chosen is typically the mean squared error. Assuming an output of dimension n, denote by $y_{i,t}$ the observed value for feature i at period t, with $i = 1, \cdots, n$ and $t = 1, \cdots, T$. Denote by $\hat{y}_{i,t}$ the corresponding prediction produced by the output layer of the LSTM. Then the mean squared error loss function is defined as:

$$\mathcal{L}(\hat{y}, y) = \sum_{i=1}^{n} \sum_{t=1}^{T} (y_{i,t} - \hat{y}_{i,t})^2 \tag{3.102}$$

The model is then trained by finding the parameter values that minimize the loss function $\mathcal{L}(\hat{y}, y)$. This is typically done by gradient descent methods and their derivatives, relying on the back-propagation approach first introduced by Rumelhart et al. (1986). RNN are typically quite difficult to train due to the vanishing gradient issue in the back-propagation process. For this reason, LSTM are often prefered as their formulation prevents this issue, even though LSTM involve considerably more parameters to estimate.

3.7.3 Prediction

Predictions are trivially obtained in the case of an LSTM: given some input x_t, the prediction $\hat{y}_t$ directly obtains from the output produced by (3.98). Quite often, the output x_t represents lags of the target y_t, that is, $x_t = (y_{t-1}, y_{t-2}, \cdots, y_{t-p})$. In this case, predictions h steps ahead from period t can be produced sequentially: predict $\hat{y}_{t+1}$ from $x_{t+1} = (y_t, y_{t-1}, \cdots, y_{t+1-p})$; then using $\hat{y}_{t+1}$ as an input for the next period, predict $\hat{y}_{t+2}$ from $x_{t+2} = (\hat{y}_{t+1}, y_t, \cdots, y_{t+2-p})$. Continue sequentially until $\hat{y}_{t+h}$ obtains from $x_{t+h} = (\hat{y}_{t+h}, \hat{y}_{t+h-1}, \cdots, \hat{y}_{t+h-p})$.

3.8 Machine learning: random forest

Random forests constitute a very popular algorithm in machine learning. They belong to the class of ensemble learning models, and rely on a simple principle: combine several weak regressors to construct a strong one. Their intuitiveness and efficiency make them very appealing models. The main algorithm is due to Breiman (2001), but variants exist, like the extra tree algorithm of Geurts et al. (2006).

3.8.1 Formulation

At the base of random forests are regression trees. A regression tree is a decion rule that maps a set of p regressors $x = (x_1, x_2, \cdots, x_p)$ to a predicted value $\hat{y}$. To do so, The predictor space is segmented into a number M of simple rectangular regions $R_1, \cdots, R_M$. This generates a set of splitting rules that can be summarized as a tree, the nodes of the tree corresponding to the decisions over feature values, and the terminal leafs corresponding to the terminal regions that provide the predicted values. This is illustrated in Figure 3.11.

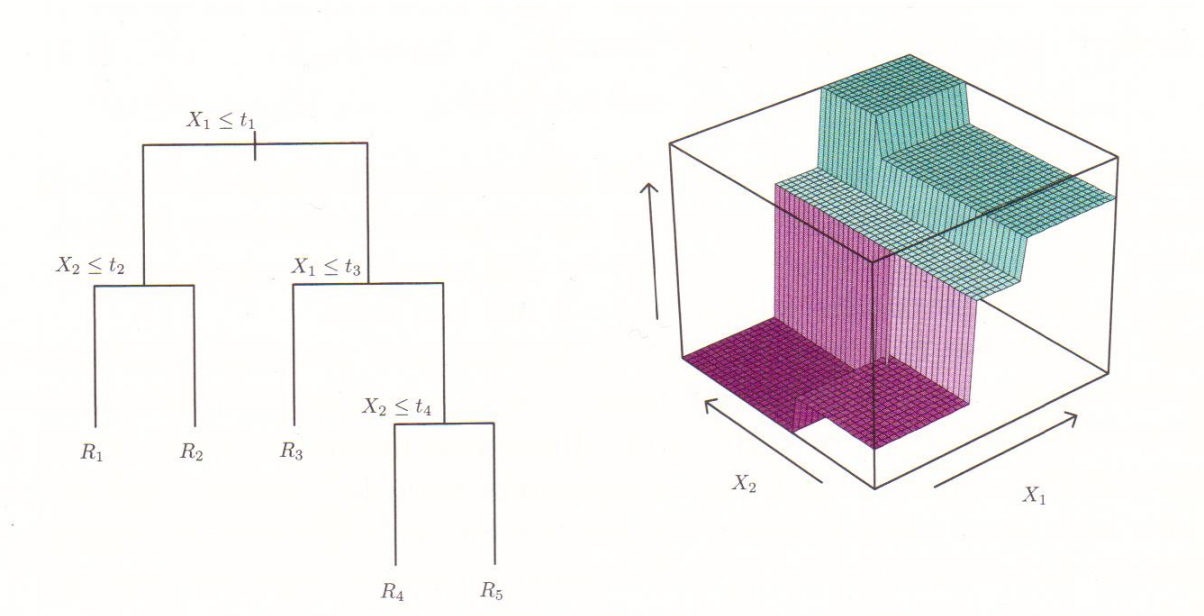


Figure 3.11: A regression tree with implied regression regions (credit: James et al. (2013))

In mathematical terms, a regression tree can be formulated as a function of the form:

$$\hat{y} = f(x) = \sum_{m=1}^{M} c_m \ . \ \mathbb{1}(x \in R_m) \tag{3.103}$$

Regression trees tend to overfit. That is, they have low bias but high variance, which results in poor predictive performance out-of-sample. The random forest algorithm overcomes this issue by reducing the variance thanks to model averaging. The principle is simple: rather than using a single tree, use an ensemble of bootstrapped trees, and average their predictions. In addition, each bootstrapped tree is fit only a small subset of features. This allows to explore a larger portion of the feature space and to decorrelate the trees.

Formally, the random forest algorithm goes as follows:

Algorithm 9: Random forest

1. Create a bootstrapped sample by randomly selecting J observations (with replacement) over the N available in the sample.

2. Fit a regression tree on this sample, but for each split considered, select randomly only a small number q of features over the p available. Set for instance $q = 0.2p$ or $q = \sqrt{p}$.

3. Repeat steps 1-2 a total number of B times, fitting at each iteration the regression tree $f^b(x)$, for $b = 1, \cdots, B$.

4. Obtain finally the random forest predictor $f(x) = \frac{1}{B}\sum_{b=1}^{B} f^b(x)$.

3.8.2 Estimation

Estimating a random forest amounts to fitting a set of regression trees. There exist several procedures to fit a regression tree,. A notorious one is the CART algorithm of Breiman et al. (1984), a simple version of which can be found in Hastie et al.:

Algorithm 10: regression tree

1. Starting with all the data, consider a splitting features $j \in 1, \cdots, p$ and a split point s, and define the pair of half-planes:
$R_1(j,s) = \{x : x_j \leq s\}$ and $R_2(j,s) = \{x : x_j > s\}$.

2. Find the splitting feature j and the split point s that solve:
$$\underset{j,s}{argmin} \left[\sum_{x_i \in R_1(j,s)} (y_i - \bar{y}_{R_1})^2 + \sum_{x_i \in R_2(j,s)} (y_i - \bar{y}_{R_2})^2 \right]$$
where $\bar{y}_{R_1}$ and $\bar{y}_{R_1}$ respectively denote the means for the observations in $R_1(j,s)$ and $R_2(j,s)$.

3. repeat the process over the resulting regions, and stop when some termination criterion is reached. Such criterion include a maximum tree depth, a mximum number of leaves, or a minimum number of observations per leaf.

3.8.3 Prediction

Prediction is perfectly trivial with random forests and follows from direct application of the model. Given a vector of regressors x, the prediction $\hat{y}$ is given by:

$$\hat{y} = f(x) = \frac{1}{B} \sum_{b=1}^{B} f^b(x) \tag{3.104}$$

3.8.4 Application

A useful aspect of random forests is that they permit to calculate what is known as variable importance. A feature x_j is important for the model if breaking the link between this feature and the target y significantly increases the prediction error. Formally, define by $\bar{x}^b$ the set of observations not selected by the bootstrap at iteration b of the random forest algorithm, and denote by $\bar{x}_p^b$ a copy of this set where a random permutation is applied to x_j. Denote respectively by $L(\bar{x}^b)$ and $L(\bar{x}_p^b)$ the mean squared errors obtained on the non-permuted and permuted datasets. If x_j is important, then randomly permuting it should deteriorate the predictions.

Therefore, one can measure the importance of x_j from:

$$I(x_j) = \frac{1}{B}\sum_{b=1}^{B} L(\tilde{x}_p^b) - L(\tilde{x}^b) \tag{3.105}$$

As a first application, Figure 3.12 plots the variable importance of the project dataset for the prediction of real GDP growth. The model predicts GDP on two lags of each feature.

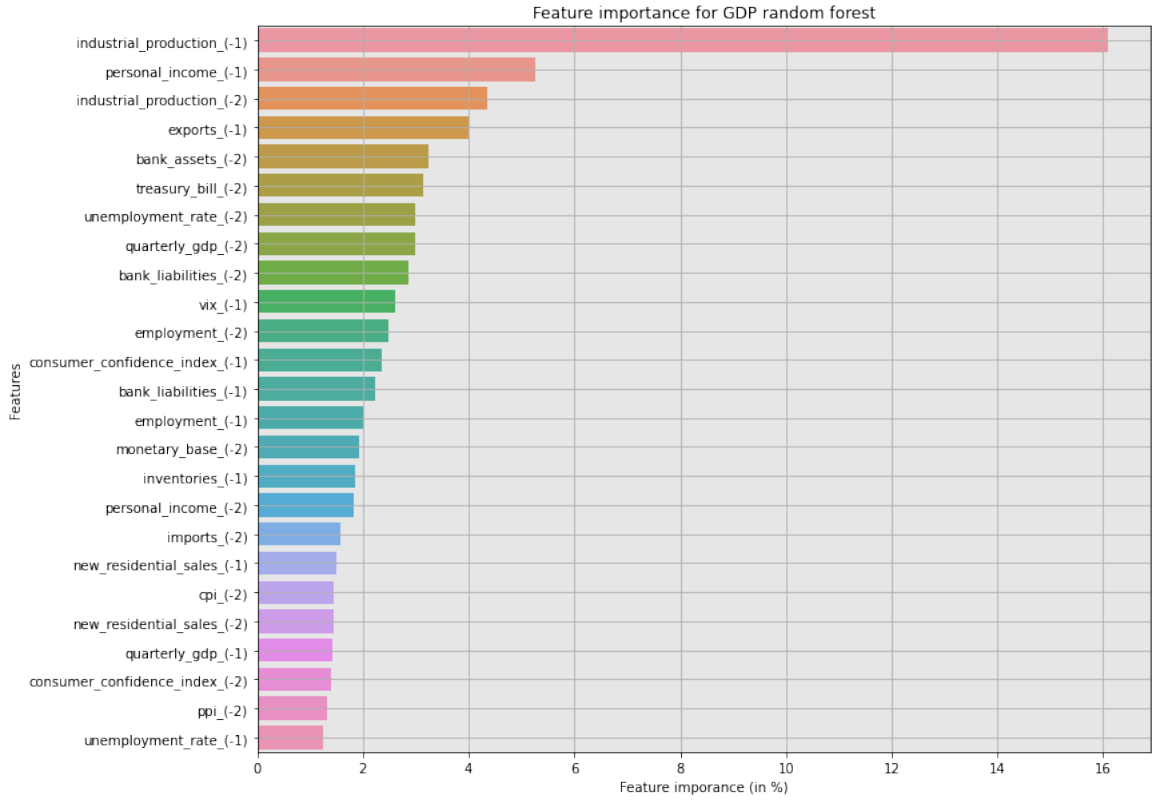


Figure 3.12: Variable importance in the prediction of GDP growth

More than 20% of the importance goes to one variable: industrial production. This is hardly surprising as industrial production represents a proxy for national production, so the correlation with GDP is high. The other important features are closely related to short-term activity and fluctuations, which once again makes sense for immediate predictions of GDP: exports and imports, employment and unemployment rate, consumer confidence index, inventories. Interestingly enough, a few financial variables also seem to help predict GDP: bank assets and liabilities, treasury bill and the VIX. The main point however is that past values of GDP contribute only moderately to nowcast GDP (about 5%). This stresses the domination of high frequency predictors over past values of low frequency variables, the former carrying more recent and valuable information.

It is also interesting to investigate which features determine the others within the dataset. This is shown in Figure 3.13.

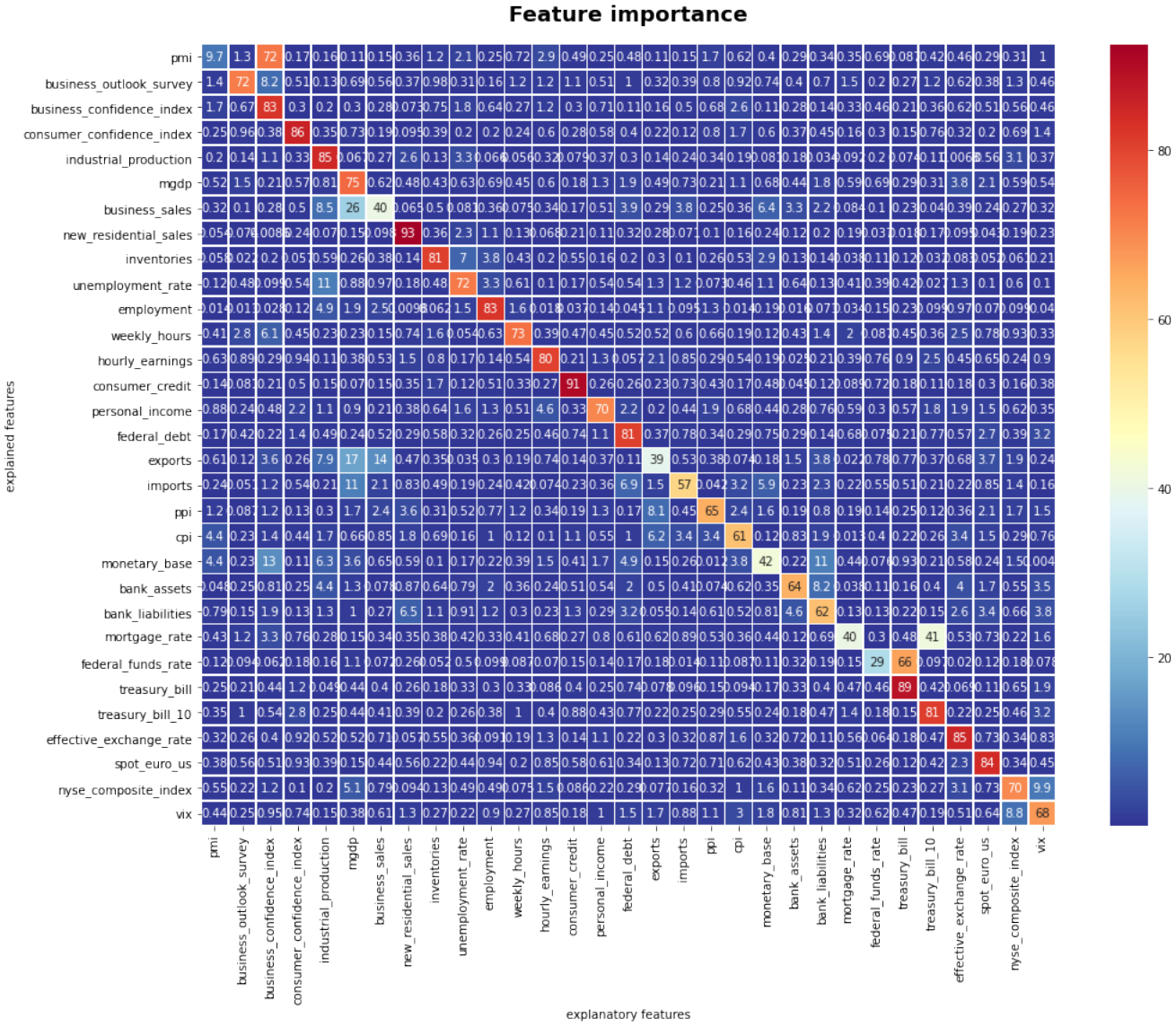


Figure 3.13: Variable importance for the dataset features

A vast majority of features is determined primarily by its own lags, as one would expect, the other features individualy representing only a marginal fraction of the predictive capacity. Yet, a few features are interestingly enough determined by other features. This is the case for instance for: the pmi which is primarily determined by the business confidence index; the federal funds rate which is primarily determined by the treasury bill; and the mortgage rate which is determined at par by the 10-year treasury bill. In all these cases, there exist however solid theoretical connections between the features which explain the observed results.

3.9 Machine learning: boosting

The field of ensemble methods is not restricted to bagging methods such as random forests. Boosting methods represent popular alternatives. They rely on model upgrading rather than model averaging. The boosting strategy was first introduced by Freund and Schapire (1997). The more general method of gradient boosting was then developed by Friedman (2001).

3.9.1 Formulation

Assume one has a sample of n observations $y = (y_1, y_2, \cdots, y_n)$ explained by n corresponding observations of the regressors $x = (x_1, x_2, \cdots, x_n)$. The objective is to fit some function $F(x)$ to predict the data y, and the corresponding loss function is expressed as:

$$L(F(x), y) = \sum_{i=1}^{n} L(y_i, F(x_i)) \tag{3.106}$$

Minimizing the loss can be viewed as a numerical optimization problem:

$$\hat{F} = \underset{f}{argmin} \; L(F(x), y) \tag{3.107}$$

Numerical optimization procedures typically amount to solving (3.107) as a sum of component vectors:

$$F(x) = \sum_{m=0}^{M} f_m(x) \qquad f_m(x) \in \mathbb{R}^n \tag{3.108}$$

where $F_0 = f_0$ is an initial guess, and each F_m is induced from F_{m-1}, the increment f_m aiming at improving the predictive capacity of the model.

3.9.2 Estimation

The gradient boosting methodology builds on the concept of steepest descent. Considering the loss function $L(F(x), y)$, one can obtain the direction of the steepest descent (fastest decrease of the loss function) as the negative of the gradient:

$$\nabla_m(x) = \frac{\partial L(F(x), y)}{\partial F(x)} \tag{3.109}$$

A natural procedure then consists in defining the increment f_m as a fraction of the negative of the gradient, in order to decrease the induced loss function:

$$f_m(x) = -\alpha \, \nabla_m(x) \tag{3.110}$$

with α some small constant that represents the learning rate of the model. One then obtains the update formula:

$$F_m(x) = F_{m-1}(x) - \alpha \, \nabla_m(x) \tag{3.111}$$

The problem with (3.111) is that the gradient $\nabla_m(x)$ is calculated only on the training points x, while the function F(x) should ideally generalize to new data not observed in the dataset. One possibility to solve this issue consists in replacing the gradient $\nabla_m(x)$ by a regression tree whose predictions will come as close as possible to it. To do so, one may simply fit a regression tree $T_m(x)$ to the negative gradient values. This will preserve most of the information provided by the gradient, but avoid the excessive specialisation of the model to the set of observations. In the case of regression models using the mean squared error $L(F(x), y) = \frac{1}{2}(y - F(x))^2$ as a loss function, it turns out that the gradient is trivial to compute:

$$\nabla_m(x) = \frac{\partial \frac{1}{2}(y - F(x))^2}{\partial F(x)} = -(y - F(x)) = -\varepsilon \tag{3.112}$$

where $\varepsilon = y - F(x)$ represents the residual from the model $F(x)$. In other words, the gradient of the loss is just the negative of the residuals, which once substituted in (3.111) yields:

$$F_m(x) = F_{m-1}(x) + \alpha \, T_m(x) \tag{3.113}$$

The model is typically initialized with a simple regressor such as $F_0(x) = \bar{y}$, with $\bar{y}$ the empirical mean of the observations. After M iteration, the final model obtains from (3.113):

$$F(x) = F_M(x) = F_0 + \alpha \sum_{m=1}^{M} T_m(x) \tag{3.114}$$

The full estimation algorithm for the gradient boosting model is then given by:

Algorithm 11: gradient boosting

1. Initialize $F_0(x) = \bar{y}$ (sample mean of the observations)

2. For $m = 1, 2, \cdots, M$, repeat:

3. Calculate the residual $\varepsilon_m = y - F_{m-1}(x)$.

4. Fit a regression tree T_m on $-\varepsilon_m$.

5. Update: $F_m = F_{m-1} + \alpha T_m$

6. final model: $F(x) = F_M(x) = F_0 + \alpha \sum_{m=1}^{M} T_m(x)$

3.9.3 Prediction

Prediction is straigtforward once the gradient boosting model is trained. It follows from direct application of (3.114). Given a vector of regressors x, the prediction $\hat{y}$ is given by:

$$\hat{y} = F(x) = F_0 + \alpha \sum_{m=1}^{M} T_m(x) \tag{3.115}$$

4 The nowcasting exercise

This final chapter develops the nowcasting exercise run on the macroeconomic dataset introduced in chapter 2, using the models detailed in chapter 3. Before discussing the results, it is useful to describe formally the predictive setting used in the exercise, along with respective spectifications of the different models.

4.1 Predictive setting

The main objective of the project is to evaluate the respective predictive performances of the models considered. The base dataset used to carry the exercise covers the period 1993m7 to 2019m12. This is for the models estimated in monthly frequency. Certain models however needs to be trained on a quarterly frequency (see next section for more information about the respective training frequencies of each model). For these models, the base sample becomes 1993q3-2019q4.

Predicting over a single sample would most likely yield results that are not robust. For this reason, the prediction exercise follows a sequential window approach. That is, the models are estimated on sample windows that grow sequentially larger: the first period of the sample is always the same, but the final period is variable. Predictions are then obtained for each sample window.

For monthly models, the first period of the sample is always 1993m7. For the final period, the sample of observations initially ends in 2016m1, then it is sequentially increased by one month. This is done over a period of 36 months or 3 years, so the largest sample covers the period 1993m7-2018m12. This gives 36 sequential predictions for monthly models.

For quarterly models, The initial period is always 1993q1, and the sample initially ends in 2016q1. It is then sequentially increased by one quarter over a period of 12 quarters or 3 years, similarly to monthly models. This yields 12 sequential predictions.

For each model and each sample, predictions are produced for 1, 2, 3 and 4 quarters ahead. What this implies in terms of timing depends on the model and the period considered. For monthly models, a one-quarter ahead predictions means a 3-month ahead prediction if the sample ends in March, June, September of December; a 2-month ahead prediction if the sample ends in January, April, July of October; and a one-month ahead prediction if the sample ends in February, May, August or November. For quarterly models, a one-quarter ahead prediction simply means predicting the next period. Even though the project is primarily focused on nowcasting, predicting at more than one quarter-ahead seems relevant as it may highlight the capacities of the different models at capturing the long-term nonlinearities.

Given the sequential windows, the exercise implies that the latest predictions will be realised over the 2019m12/2019q4 period. This excludes the COVID period from the project. This is purposeful, as the exercise intends to determine the best prediction model in normal times, the prediction of crisis times being a different question on its own.

For each prediction, the accuracy is measured using the root mean squared error as a criterion. The latter is defined as:

$$rmse(\hat{y}_t) = \sqrt{(\hat{y}_t - y_t)^2} \tag{4.1}$$

The overall performance of the model is then obtained by averaging the RMSE over all the samples. The RMSE are also normalised by the feature variances to avoid scale effects.

4.2 Model specifications

There are 9 base models considered in the projet: 3 nowcasting models (dynamic factor model, MIDAS regression and mixed frequency Bayesian VAR), 3 econometrics models (VAR, Bayesian VAR and time-varying Bayesian VAR), and 3 machine learning models (LSTM, random forest and gradient boosting).

The dynamic factor model is trained as the monthly frequency. It includes the full macroeconomic dataset (31 monthly series), and one quarterly series of GDP. It is fundamentally determined by two parameters: the number of dynamic factors p included in the model, and the

number of common shocks q assumed to drive the factor dynamics. The optimal specification is determined by a grid search, using the last 25% of the sample as a test set. This yields $p = 4$ and $q = 2$.

The MIDAS regression is trained on quarterly frequency, even though it includes also monthly regressors. The small dataset is used for monthly series, to keep the approach parsimonious. Following the MIDAS litterature on quarterly/monthly MIDAS models (see e.g. Ghysels et al. (2016)), the number of high frequency lags is set to match the number of lagged periods of the lower frequency variable, and keep the specification parsimonious. This leads to set $p = 2$ low frequency lags and $q = 6$ high frequency lags.

The mixed frequency Bayesian VAR is trained at the monthly frequency. Because it adopts the Bayesian approach, the priors for the VAR coefficients β and the residual covariance matrix Σ must be specified. The specification retained is the same as the regular Bayesian VAR and is described below.

The standard VAR model has to be trained on a balanced panel. Due to the quarterly frequency of the GDP growth series, the whole model must be trained on quarterly data. Also, because VAR models perform poorly on very large numbers of feature, the small dataset is used. The only remaining parameter to specify is the number of lags. Following standard practices, it is determined as the one that minimizes the Akaike Information Criterion (AIC). This yields $p = 2$.

The Bayesian VAR is similar to the regular VAR, but benefits from the Minnesota prior. The hyperparameter values have then to be specified. The litterature provides strandard values (Litterman (1986)), but it is better to optimize the values. This is done on a grid search seeking to optimize the marginal likelihood $f(y)$ of the model. This results in $\rho = 0.85$, $\lambda_1 = 0.4$, $\lambda_2 = 0.3$, $\lambda_3 = 1.5$ and $\lambda_4 = 1000$.

The time-varying Bayesian VAR also follows the same specification as the VAR and BVAR models., but its dynamic parameters involve specific hyperparameters. The values for the dynamic equations are $\rho_i = \gamma_i = \alpha_i = 0.9$. The equilibrium values b_i, s_i and d_i are derived from static OLS estimates. For the inverse Wishart priors on the variance-covariance hyperparame-

ters Ω_i and Ψ_i, the degrees of freedom are set to a small value of 5 additional to the parameter dimension. The scale parameters are set to $\Upsilon_0 = \Theta_0 = 0.001I$. Similarly, the shape and scale parameters of the inverse Gamma prior distribution on ϕ_i are set to $\kappa_0 = 5$ and $\omega_0 = 0.01$. These priors are mildly informative, being sufficiently loose to allow for a signicant degree of time variation in the dynamic parameters, but sufficiently restrictive to avoid implausible behaviours. Finally, the initial period variance scaling terms are set to $\tau = \mu = \epsilon = 5$ in order to obtain a variance over the initial periods which is roughly equivalent to that prevailing for the rest of the sample.

The LSTM model is trained on the small dataset at the quarterly frequency. To make it similar to the VAR models, the output vector y_t is chosen to be the set of the 9 features of the small dataset at period t, while the input features x_t is the first two lags of the output values. That is, the model is of the form $y_t = f(x_t) = f(y_{t-1}, y_{t-2})$. Predictions for $t+h$ can then be obtained easily in a recursive way by predicting $y_{t+1}, \cdots, y_{t+h}$. The optimal number of layers and cells is estimated on a test sample representing 20% of the sample. One hidden layer with only 3 cells is found to be optimal, resulting in a model in an encoder-like fashion. The model is trained on 500 epochs, with a 25% dropout rate on the hidden layer to prevent overfitting, and an ADAM optimizer.

The random forest algorithm is designed to work well on many features. For this reason, it uses the full dataset of 31 features. The setting is divided in two different models. The first model predicts GDP from its own lagged values and lags of the monthly regressors. In essence, it is thus similar to the MIDAS regression, except that the regression is estimated by random forest on the features rather than using the standard MIDAS approach. Consistently with the MIDAS model, it includes 2 lags of GDP and 6 lags of the monthly features. The second model aims at predicting all the non-GDP monthly features. It is designed as a large VAR model: one random forest model is estimated for each feature, the regressors being lags of the whole set of monthly features. This gives a system of random forest models which once put together act like a large VAR. Random forests work best with weak learners, so given the number of features, a maximum depth of 5 is granted, for a model with 100 trees.

The gradient boosting model, finally, is build to act a "boosted VAR". It uses the small dataset, on a quarterly frequency. Each feature in the dataset is regressed by gradient boosting on 2 of its own lags and other feature lags. The resulting system can be interpreted as a VAR with 2 lags, but with a boosting, nonlinear flavour. There is a trade-off between the learning rate α and the number of trees M. A reasonable choice to carry enough flexibility while avoiding overfitting consists in setting $\alpha = 0.1$ and $M = 100$. Each tree is made a weak learner by limitting the depth to 5.

This constitutes the range of base models. To these 9 models, a number of benchmarks or naive models can be added for the sake of comparison. The first is simply a static predictive model using the last known sample value before prediction. The second is a maximum likelihood VAR model using Ridge regularization. The third consists in the GDP predictions produced by Nowcasting.com. Finally, the last benchmark is the GDP predictions produced by Bloomberg[1] two months before release.

4.3 Nowcasting GDP

To obtain a preliminary view on the respective performances of the different model, a simple plot of the predictions can prove useful. On Figure 4.1, each row represents a prediction horizon (1 quarter ahead or nowcast at the top, up to 4 quarters ahead at the bottom). Each column represents one class of models: nowcasting on the left, econometrics in the middle, and machine learning on the right. The actual GDP data which represents the prediction target is depicted by the black line. The other models are represented for each class by the green, orange and purple lines.

Taking an overall view on the picture, it seems that none of the models manages to capture very closely the dynamics of the data, whatever the prediction horizons. Most of them exhibit periods of good fit, and other periods where the link with the target data becomes quite loose.

[1]The author is grateful to Adam Majewski for providing the Bloomberg series of predictions.

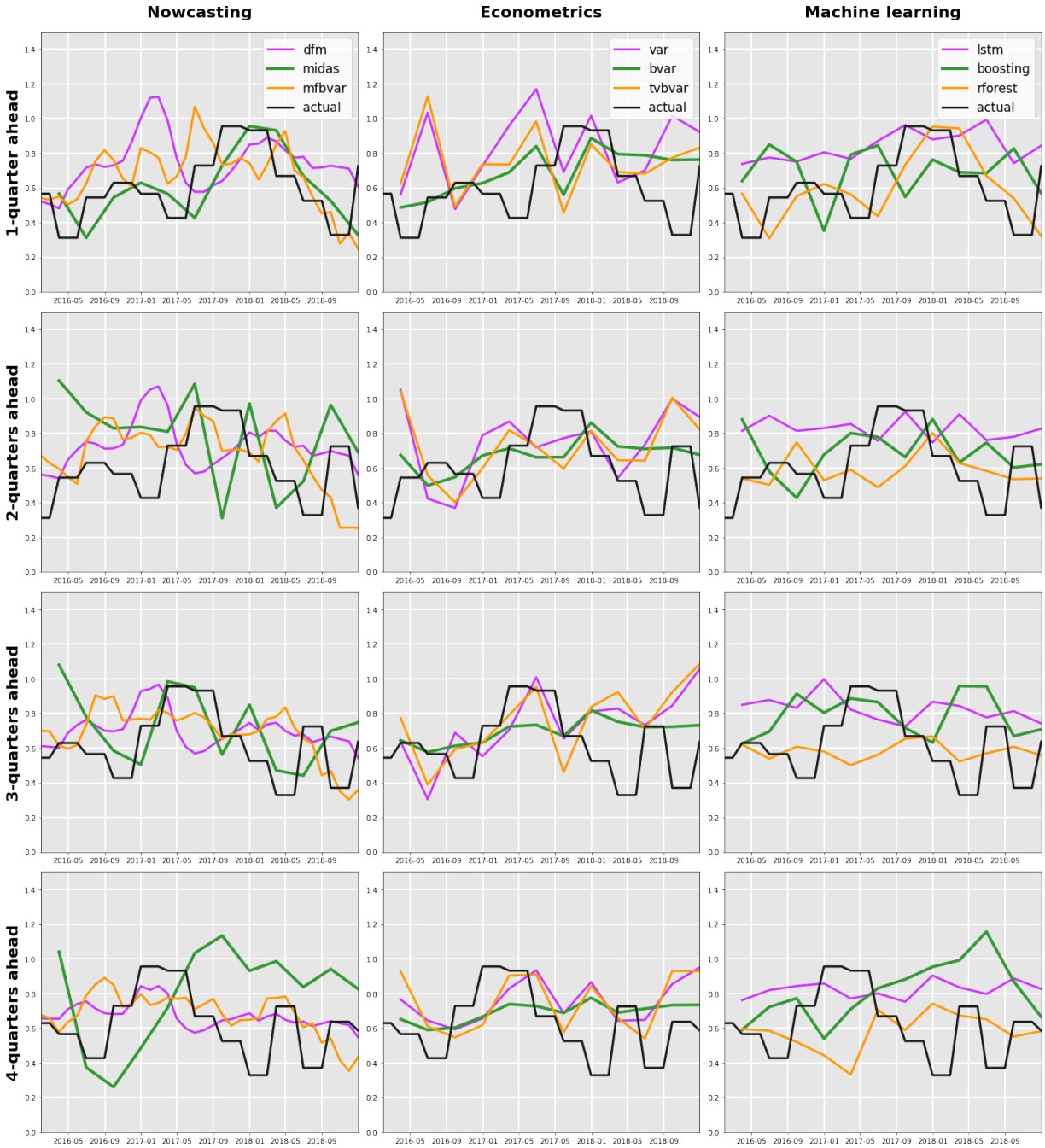


Figure 4.1: GDP predictions: all models, all prediction horizons

Looking at the nowcasting models, the dynamic factor model seems to perform average overall. Its fit is never really close, save perhaps for the 3 quarters ahead horizon. The MIDAS regression seems to perform reasonably well at the nowcasting horizon, but exhibit a lag between the dynamics and the prediction. Its performances significantly degrade at longer horizons. The mixed frequency Bayesian VAR produces fair performances. At nowcast horizons, it shows average performance on the first half of the sample and good performance on the second half.

In terms of econometrics models, the picture looks fairly clear. Both the regular VAR and the time-varying BVAR displays large swings which keep them most of the time far from the target. This is probably the effect of overfitting for the former, and the the capture of artifical dynamc moves for the latter. The Bayesian VAR by contrast shows predictions of more moderate amplitudes, as a consequence of the Minnesota prior which generates a parsimonious model. Its predictions look overall closer to the target.

The machine learning models, finally, don't seem to perform very well. The boosting approach seems to produce large and short-lived oscillations revolving around the target. The random forest predictions are virtually similar to that produced by the MIDAS regression. These two models share the same features, so this only indicates that the parsimonious feature selection exerted by the random forest is quite equivalent to the parsimonious lag structure strategy of the standard MIDAS regression. In the end, the random forest suffers from the same quality and defaults as the MIDAS model: the predictions may look close to the actual values, but they actually always come with a lag. The LSTM, finally, produces smoother predictions. It is however clear that it is biased upward, resulting in poor predictions.

To confirm these intuitions formally, the average RMSE on the predictions of the forecasting exercise are reported in Table 4.1. For each prediction horizon (i.e. each column), the green entry represents the best predictor while the yellow entry marks the second best prediction.

		1 quarter ahead	2 quarters ahead	3 quarters ahead	4 quarters ahead
nowcasting	dfm	0,707	0,694	0,598	0,443
	midas	0,593	0,909	0,609	1,154
	mfbvar	0,536	0,613	0,543	0,503
econometrics	var	0,885	0,808	0,697	0,729
	bvar	0,502	0,558	0,486	0,547
	tvbvar	0,784	0,732	0,740	0,706
machine learning	lstm	0,712	0,747	0,769	0,745
	random forest	0,592	0,566	0,635	0,552
	boosting	0,605	0,568	0,594	0,548
benchmarks	last value	0,593	0,697	0,775	0,749
	ridge var	0,835	0,761	0,704	0,678
	nowcasting.com	0,668	0,679	–	–
	bloomberg	1,019	–	–	–

Table 4.1: RMSE on GDP forecasts: all horizons

A few conclusions stand. First, two models seem to dominate the whole exercise: the Bayesian VAR (as the best model), and the mixed frequency Bayesian VAR (as the second best model). The conclusion is very robust as the two models always produce the best performance, except in two cases (the random forest at 2 quarters ahead and the dynamic factor model at 4 quarters ahead) where their performance remains significantly better than most of their competitors. Also, their RMSE proves considerably lower than that of the other models, usually of the order of 20% to 40%, so their superiority is quite significant. These two models share a number of properties: they are linear models; they are trained on the small dataset; and they both benefit from a parsimonious representation due to the implementation of the Minnesota prior. This suggests that the behaviour of the data is best represented by a linear regime using only the features with the most explanatory power and only a limited number of parameters to define the dynamics.

The second conclusion is that the class of nowcasting models don't produce the best short term performance (except of course the mfbvar), even though they are built for this purpose . The dynamic factor model and the MIDAS regression achieve fair, but not excellent predictions, even though they make use of high frequency, monthly data to extract more information about the incoming quarterly release. One possible explanation is the overly simplistic formulation of these two models. Another explanation, possibly more convincing, is the dichotomy that exists within these models. On the one hand, the dynamics of the high frequency features is estimated on T monthly observations. On the other hand, the predictive model of the low frequency GDP can still only be estimated on $T/3$ observations, due to the quarterly nature of the series. Apparently the reduced number of low frequency observations used to train the predictive model for GDP eventually anihilates the benefit of the larger high frequency sample.

The third conclusion is the incapacity of the machine learning models to produce very good predictions, save for the random forest at nowcast horizons. This in fact hardly surprising. These models are primarily designed to detect nonlinear behaviours, and are very data greedy. By contrast, the dataset of the exercise exhibit simple linear behaviours, and the dataset is quite short. The detected nonlinearities are probably the outcome of noisy processes rather than underlying dynamics, leading to overfitting and mediocre predictions. The same conclusion holds indeed for the tv-BVAR, typically intended to capture abrupt and nonlinear changes in

the dynamics, which results here in poor predictions.

The final conclusion is that none of the benchmarks manage to beat the BVAR and the mf-BVAR. In particular, the professional forecasts propose by Bloomberg and Nowcasting.com are significantly worse than the predictions produced by the simple Bayesian VAR. In fact, the performance of the nowcasting.com predictions are quite similar to that of the dynamic factor model developed for the project, confirming the robustness of the conclusion. The Ridge VAR performs poorly at any horizon. Most likely, the blind regularization applied by the methodology cannot compete with the more selective regularization induced by the Minnesota prior.

The simple predictor using the last known sample value performs surprisingly well at short horizon, beating most other models. Interstingly enough, the MIDAS regression and the random forest produce virtually similar performance. This is because these two models essentially replicate the last value for their nowcasts, as can be seen from Figure 4.1. This trivial predictor may thus represent a good option, even though it is possible to achieve better predictions than this simplistic strategy.

These results are not fully refined. Indeed, two models in the panel are estimated at the monthly frequency: the dynamic factor model and the mixed frequency Bayesian VAR, to which must be added the dynamic factor model of Nowcasting.com. Because these models are monthly, three forecasts are produced for a given quarter: three months before, two months before, and one month before the release. The timing of the prediction may matter: as more information normally yields more accurate forecasts, predictions closer to the release should be better than early forecasts. Table 4.2 reports the average RMSE for the four models, distinguishing the different timings before release.

	3 months before	2 months before	1 month before
dfm	0,755	0,703	0,663
mfbvar	0,659	0,504	0,444
nowcasting.com	0,726	0,620	0,659
bvar benchmark	0,502	0,502	0,502

Table 4.2: RMSE on nowcasts 1, 2 and 3 months before release

The results confirm the previous conclusions. The Bayesian VAR remains the best model three months and two months before release. The difference between the Bayesian VAR and the mixed frequency Bayesian VAR becomes however fairly insignificant at the two month horizon. At one month before release, the mixed frequency BVAR becomes best, with a significant discrepancy with the performance of the standard Bayesian VAR. This makes sense: because the Bayesian VAR is quarterly, the quarter represents the unit of measurement. Thus, predicting one quarter ahead implies a one period ahead prediction, and the Bayesian VAR performs well. The mixed frequency BVAR on the other hand is a monthly model. Hence a one quarter ahead prediction may imply a prediction up to three periods ahead. At three months before, the prediction proves much less accurate. Two months before the release, the two predictions play virtually at par: even though the mf-BVAR must predict at two periods ahead against one period only for the BVAR, the larger training sample and the additional information obtained in-between compensate and result in similar prediction performance. At one month before release, the mf-BVAR overtakes the Bayesian VAR and becomes the best predictor.

The other predictors provide consistent results and produce better forecasts as the timing gets closer to the release and more information is included in the information set. The dynamic factor model (both for the project and the one proposed by Nowcasting.com) looks overall weak compared to the Bayesian VAR models.

Beyond the pure measurement error illustrated by the Root Mean Squared Error, it migh also be of interest to analyse the capacity of a model to predict the right direction of change in GDP growth. Ideally, a model should not only produce predictions close to actual values, but it should also be capable of capturing adequately the direction of evolution of a variable, in particular in financial applications. The percentage of correct direction predictions are reported in Table 4.3.

		1 quarter ahead	2 quarters ahead	3 quarters ahead	4 quarters ahead
nowcasting	dfm	0.556	0.722	0.778	0.917
	midas	0.417	0.667	0.667	0.583
	mfbvar	0.805	0.694	0.833	0.861
econometrics	var	0.417	0.667	0.667	0.917
	bvar	0.667	0.667	0.833	0.833
	tvbvar	0.417	0.75	0.833	0.833
machine learning	lstm	0.583	0.75	0.833	0.583
	random forest	0.417	0.75	0.5	0.417
	boosting	0.583	0.583	0.75	0.167

Table 4.3: Percentage of correct direction predictions

At the nowcast horizon, the mixed frequency Bayesian VAR et the Bayesian VAR significantly dominate the other models. They represent, in fact, the only models that predict the correct direction of change more than two third of the times. The other models are either slitghly better than a random predictor, or below it like the MIDAS regression or the random forest. This is expected: because the MIDAS and random forest models are equivalent to the last value predictor at nowcast horizon, they miss the directional change most of the time, resulting in poor dorectional performance.

At longer horizon, the dynamic factor model displays good performances overall, predicting the correct direction in more than 70% of the cases. It is however (marginally) overperformed by other models like the tvbvar and the lstm, at the horizon of two and three quarters. It becomes the best model at the four quarter horizon. The VAR, random forest and boosting model produce overall poor performances. Eventually, which model is best to predict the direction beyond the nowcast horizon remains inconclusive.

As a conclusion to this section, it seems clear that two models dominate the nowcast exercise for GDP: the Bayesian VAR, and the mixed frequency Bayesian VAR. The two models are better than their competitors both in terms of forecast accuracy, and capacity to predict the correct direction of change. The standard Bayesian VAR performs best when the prediction is preduced one full quarter before the release. One month before release, higher frequency variables are updated and the mixed frequency BVAR becomes dominant. Which model is most suitable at the two month deadline remains ambiguous.

4.4 Nowcasting monthly features

The project focuses primarily on predicting the low frequency GDP using the higher frequency information of external features. However, the prediction of these features may also represent a subject of interest on its own. A graphical representation of all the forecasts for all the features would not be manageable. A simpler alternative consists in providing a correlation matrix of the predictions generated by the models with the actual data. Figure 4.2 displays such a correlation matrix for the small dataset, at the nowcast horizon (one quarter ahead). In the matrix, each entry represents the correlation between a feature (columns) and its prediction (by the model given on the row). A column of high correlations (dark red entries) thus indicates a feature that is overall well predicted, whatever the model. A row of high correlations reveals a model that predicts well in general.

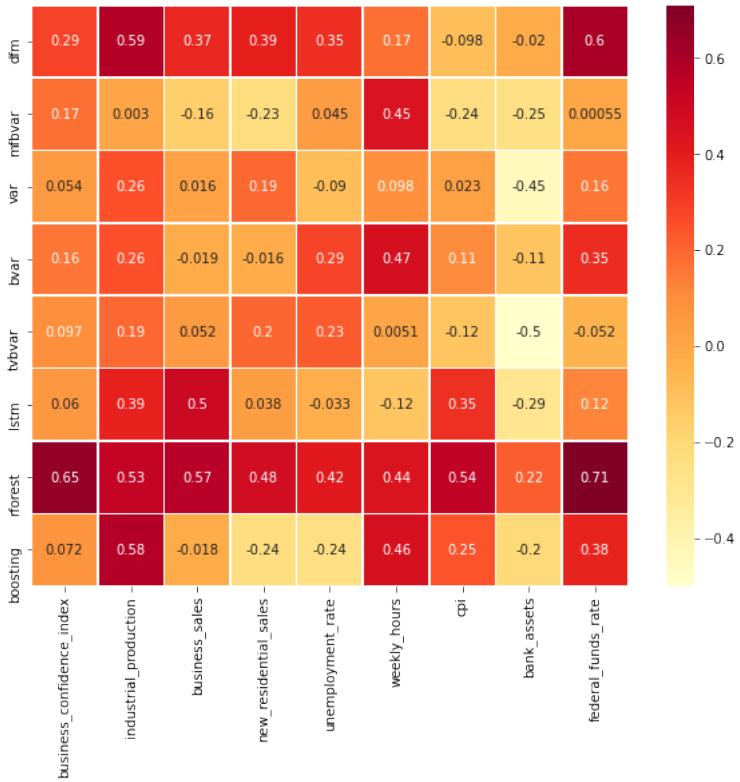


Figure 4.2: Correlations: actual VS. model predictions, one quarter ahead

It appears that the features that are being best predicted are those strongly correlated to business cycles: industrial production, business confidence index, business sales, and weekly hours. This confirms that most of the predictive power of the models goes to business cycles related features, and thus that the models can constitute good predictors for real GDP growth rate.

Two models seem to dominate the others in tems of nowcasting: the random forest, and to a lesser extent the dynamic factor model. To confirm this intuition, a formal analysis of the RMSE errors produced by the models and benchmarks are presented in Table 4.4.

		Business confidence index	Industrial production	Business sales	New residential sales	Unemployment rate	Weekly hours	CPI	Bank assets	Federal Funds rate
nowcasting	dfm	0,267	0,104	0,073	0,048	0,340	0,238	0,249	0,167	0,303
	mfbvar	0,341	0,165	0,099	0,037	0,398	0,216	0,322	0,140	0,252
econometrics	var	0,340	0,142	0,091	0,029	0,528	0,417	0,297	0,197	0,251
	bvar	0,311	0,125	0,078	0,031	0,309	0,245	0,249	0,190	0,216
	tvbvar	0,338	0,147	0,090	0,029	0,406	0,384	0,272	0,178	0,297
machine learning	lstm	0,320	0,127	0,056	0,042	0,462	0,338	0,211	0,189	0,265
	random forest	0,204	0,116	0,065	0,020	0,290	0,197	0,182	0,104	0,147
	boosting	0,316	0,126	0,081	0,035	0,471	0,210	0,258	0,190	0,268
benchmarks	last value	0,361	0,169	0,097	0,033	0,342	0,260	0,270	0,108	0,212
	ridge var	0,343	0,139	0,091	0,029	0,522	0,408	0,297	0,193	0,248

Table 4.4: Feature RMSE, 1 quarter ahead

The results confirm the intuition provided by the correlation matrix. The random forest represents the best model to nowcast the features in all cases, except for industrial production and business sales for which it becomes the second best. Its predictive performance is always significantly better than the competing models, with RMSE around 10-50% smaller.

It may look surprising that the random forest performs so well for the monthly features, while the random forest MIDAS performed only average to predict GDP. It seems that the use of the large macro dataset in the case of the random forest VAR manages to extract most of the information for feature nowcasts. By constrast, the other models, in particular in the VAR family work only on the small dataset. This may here prove insufficient as the variables in the small dataset were primarily chosen to predict GDP and may hence lack relevant information for the other features.

Also, the VAR models are clearly at a disadvantage since they are trained at the quarterly frequency, reducing the training sample to 1/3 of the monthly sample, and losing the underlying monthly dynamics. For the comparison to be really meaningful, the VAR models should be trained again, this time on fully monthly samples[2].

The other models prove hardly any better. The dynamic factor models produces decent forecasts, but constitutes the best or second best model in two cases only. The LSTM and boosting model constitute the best/second best model in only one case each. Their prediction performances are overall close to that provided by the VAR family, which supports the hypothesis that the small dataset and quarterly frequency may be at fault here.

The benchmarks, finally, look anecdotal. The last value predictor performs fair but its RMSE ar emostly above that of the other models, including the VARs. The Ridge VAR performs especially poorly.

It may also be interesting to consider the feature predictions at longer horizons. Tables 4.5, 4.6 and 4.7 display the average RMSE for the horizons of 2, 3 and 4 quarters ahead respectively.

The results are unexpected. Unlike the nowcast, the random forest almost completely disapears from the optimal models. It is replaced almost exclusively by the mixed frequency Bayesian VAR, which becomes the best model to predict at 2, 3 and 4 quarters ahead. The difference is quite significant, the mfbvar typically overperforming the random forest by about 30%. The same conclusion stands for the other competing models. They are usually unambiguously beaten by the mfbvar, with a 10%-30% margin. The most serious competitor seems to be the dynamic factor model which occasionally manages to be the best or second best model, with a performance close to the mfbvar. The other models on the other hand look quite anecdotal and achieve only sub-par predictive performance.

[2]The objective of the project consisted primarily in designing a nowcast model for GDP, which is why all the VARs were trained at the quarterly frequency. Repeating the exercise on monthly samples for the features could not be done due to lack of time

		Business confidence index	Industrial production	Business sales	New residential sales	Unemployment rate	Weekly hours	CPI	Bank assets	Federal Funds rate
nowcasting	dfm	0,299	0,106	0,077	0,043	0,359	0,232	0,277	0,186	0,290
	mfbvar	0,244	0,106	0,043	0,018	0,229	0,213	0,163	0,168	0,165
econometrics	var	0,412	0,153	0,093	0,047	0,605	0,287	0,273	0,222	0,348
	bvar	0,309	0,127	0,060	0,044	0,443	0,219	0,232	0,220	0,307
	tvbvar	0,357	0,155	0,094	0,046	0,596	0,296	0,296	0,184	0,439
machine learning	lstm	0,332	0,149	0,064	0,045	0,447	0,307	0,238	0,211	0,286
	random forest	0,375	0,146	0,083	0,039	0,379	0,267	0,277	0,126	0,305
	boosting	0,340	0,139	0,086	0,041	0,511	0,210	0,246	0,261	0,262
benchmarks	last value	0,320	0,175	0,097	0,050	0,508	0,299	0,395	0,140	0,274
	ridge var	0,393	0,153	0,090	0,047	0,587	0,272	0,271	0,221	0,347

Table 4.5: Feature RMSE, 2 quarters ahead

		Business confidence index	Industrial production	Business sales	New residential sales	Unemployment rate	Weekly hours	CPI	Bank assets	Federal Funds rate
nowcasting	dfm	0,291	0,124	0,058	0,047	0,457	0,263	0,256	0,208	0,331
	mfbvar	0,305	0,101	0,050	0,020	0,374	0,243	0,203	0,211	0,140
econometrics	var	0,397	0,190	0,097	0,048	0,521	0,337	0,221	0,202	0,434
	bvar	0,354	0,131	0,069	0,045	0,401	0,256	0,222	0,223	0,355
	tvbvar	0,379	0,189	0,097	0,048	0,440	0,322	0,204	0,158	0,534
machine learning	lstm	0,347	0,144	0,060	0,037	0,404	0,292	0,234	0,246	0,234
	random forest	0,319	0,171	0,066	0,045	0,410	0,296	0,287	0,143	0,351
	boosting	0,420	0,164	0,071	0,042	0,274	0,301	0,239	0,225	0,395
benchmarks	last value	0,291	0,210	0,083	0,051	0,390	0,321	0,338	0,165	0,365
	ridge var	0,390	0,181	0,094	0,048	0,487	0,325	0,218	0,209	0,432

Table 4.6: Feature RMSE, 3 quarters ahead

		Business confidence index	Industrial production	Business sales	New residential sales	Unemployment rate	Weekly hours	CPI	Bank assets	Federal Funds rate
nowcasting	dfm	0,309	0,117	0,056	0,051	0,464	0,245	0,234	0,210	0,421
	mfbvar	0,346	0,122	0,064	0,035	0,452	0,220	0,237	0,215	0,311
econometrics	var	0,304	0,166	0,065	0,049	0,478	0,303	0,179	0,190	0,427
	bvar	0,312	0,141	0,070	0,046	0,433	0,255	0,216	0,201	0,456
	tvbvar	0,361	0,176	0,082	0,049	0,445	0,335	0,183	0,136	0,485
machine learning	lstm	0,386	0,147	0,069	0,043	0,475	0,310	0,213	0,214	0,345
	random forest	0,359	0,181	0,090	0,043	0,419	0,310	0,266	0,141	0,411
	boosting	0,386	0,181	0,079	0,051	0,318	0,310	0,193	0,190	0,494
benchmarks	last value	0,441	0,219	0,107	0,051	0,496	0,321	0,299	0,156	0,424
	ridge var	0,303	0,158	0,065	0,050	0,465	0,288	0,184	0,198	0,427

Table 4.7: Feature RMSE, 4 quarters ahead

It is difficult to explain this change in performance between the random forest and the mfbvar. To clarify the case, Figure 4.3 plots the predictions for the radom forest and mixed frequency Bayesian VAR models at the one, two and three quarter horizon, for a selection of features exhibiting strong performance reversal between the random forest and the mfbvar.

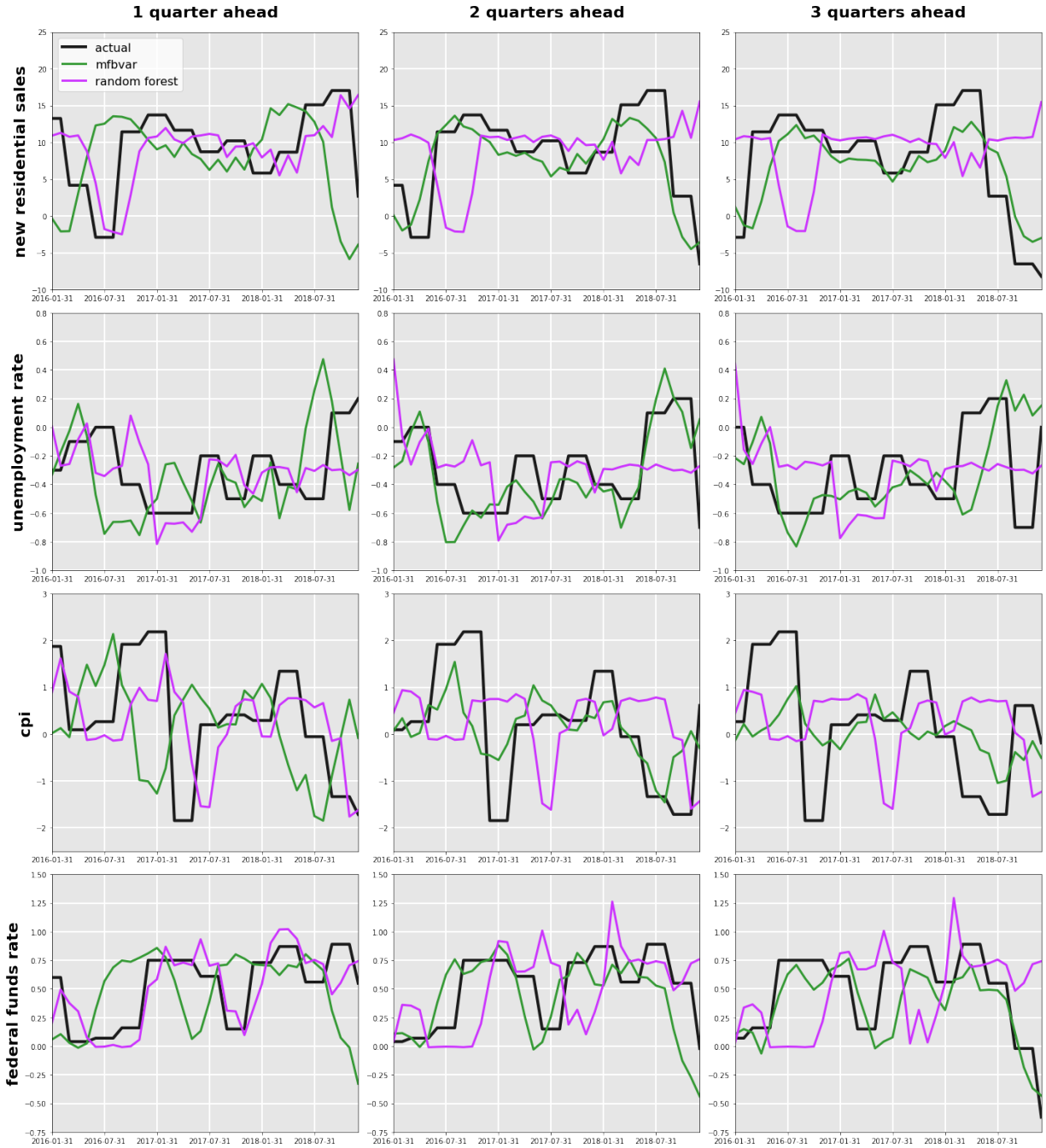


Figure 4.3: Predictions of random forest and mixed frequency Bayesian VAR

A pattern seems to emerge. The random forest captures the dynamics quite adequately at the nowcast horizon. Beyond the nowcast, however, it gradually starts lagging behind, which results in poor accuracy. By contrast, the mixed frequency Bayesian VAR seems to work the opposite way. As surprising at it may look, the model appears to anticipate the dynamic pattern at nowcast horizons. That is, it seems to identify correctly the overall direction of the dynamics, but somewhat too early. Following, the random forest proces more accurate for nowcasts. At

the two, three and four quarters ahead horizon however, the mfbvar predictions do not precede anymore the actual movemements of the data, and become better than the laggy random forests predictions. They become, in fact, quite close to the actual dynamics.

It is quite difficult to explain this behaviour. The only difference in terms of dataset between the two models is the presence of low frequency, GDP data in the mfbvar. Possibly, this key economic variable contains forward information on the features, which produces this anticipated behaviour. Possiby also, the Minnesota prior allows for a more easily mean-reverting model behaviour, rendering anticipations possible. Overall, these explanations are only mildly convincing, but the anticipation behaviour looks clear and unambiguous.

At the end of the exercise, the conclusions are clear. At the nowcast horizon, the random forest dominates. The class of VAR models however should not be discared as their restricted dataset and quarterly frequency place them at a strong disadvantage. More testing is probably necessary. At longer horizons, the mixed frequency Bayesian VAR becomes optimal, as it replicates closely the actual behaviour of the data. This is in contrast with the nowcasts where it seems to anticipate the dynamics and exhibits inadequate moves.

4.5 Future developments

This project was centered on one main objective: determine the best model to predict real GDP growth in the short run. The prediction exercise shed light on two models: a standard, quarterly Bayesian VAR, and a more sophisticated monthly mixed frequency Bayesian VAR. The former seems better when predicting one quarter before the release, while the latter proves more accurate closer to the release deadline.

It is still uncertain what is the best model in-between, that is, around two months before the release of GDP. To establish this formally and establish a model that could be internally used by CFM, more exploration is needed.

In the first place, one should determine what can be an optimal dataset for the US. Both the large and small datasets used in this project are standard, but they may not be optimal for the specific case of the United States. Also, one should keep in mind the trade-off that exists bewteen the flexibility in the choice of features, and the available history of the data. Standard datasets may result in less sophisticated dynamics, but the additional history may more than compensate in terms of forecast accuracy. As a simple example, the dataset used for the project starts in 1993, due only to the limitted history of certain features, while most standard series provided by the FED/FRED start in the 1940's. One must also pay attention to the availability of the different series in both monthly and quarterly frequencies. With an optimized dataset, one might then conduct a prediction exercise similar to the one carried in this project, but possibly on a longer window for improved confidence in the conclusion. Based on this further exploration, a solid nowcasting model could be obtained for nowcasting GDP.

Alternatively, a distinct exercise could be conducted to tackle a different issue: the building of an optimal predictor for GDP in a context of crisis. This question is important in general, and crucial for a hedge fund which will benefit from key information if a crisis is properly anticipated. The exercise would consist in detecting the crisis in the first place. This could be done for instance with a Markov-switching model in the line of Warne et al. (2015). Then the rapidly changing dynamics could be predicted from time-varying models similar to the one used in the project, as developed by Primiceri (2005).

I believe either of these tracks could be profitably pursued by CFM and significantly contribute to their investment decisions.

References

Banbura, M., Giannone, D., and Reichlin, L. (2010). Nowcasting. Working Papers 1275, European Central Bank.

Breiman, L. (2001). Random forests. *Machine Learning*, (45):5–32.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and regression trees*. The Wadsworth statistics/probability series. Wadsworth & Brooks/Cole Advanced Books & Software.

Carter, C. and Kohn, R. (1994). On gibbs sampling for state space models. *Biometrika*, 81(3).

Del Negro, M. and Primiceri, G. E. (2015). Time-varying structural vector autoregressions and monetary policy: a corrigendum. *Review of Economic Studies*, 82(4):1342–1345.

Doan, T., Litterman, R., and Sims, C. (1984). Forecasting and conditional projection using realistic prior distributions. *Econometric Reviews*, 3(1):1–100.

Doz, C., Giannone, D., and Reichlin, L. (2011). A two-step estimator for large approximate dynamic factor models based on kalman filtering. *Journal of Econometrics*, 164(1):188–205.

Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *journal of computer and system sciences*, 55:119–139.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232.

Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, 63:3–42.

Ghysels, E., Kvedaras, V., and Zemlys, V. (2016). Mixed frequency data sampling regression models: The R package midasr. *Journal of Statistical Software, Articles*, 72(4).

Ghysels, E. and Qian, H. (2016). Estimating MIDAS regressions via OLS with polynomial parameter profiling. *Econometrics and Statistics*, 9:1–16.

Ghysels, E., Santa-Clara, P., and Valkanov, R. (2004). The MIDAS touch: Mixed data sampling regression models. CIRANO Working Papers 2004s-20, CIRANO.

Ghysels, E., Sinko, A., and Valkanov, R. (2007). MIDAS regressions: Further results and new directions. *Econometric Reviews*, 26(1):53–90.

Ghysels, E. and Wright, J. H. (2009). Forecasting professional forecasters. *Journal of Business & Economic Statistics*, 27(4).

Giannone, D., Lenza, M., and Primiceri, G. (2015). Prior selection for vector autoregressions. *Review of Economics and Statistics*, 97(2):436–451.

Giannone, D., Reichlin, L., and Sala, L. (2005). Monetary policy in real time. pages 161–224.

Giannone, D., Reichlin, L., and Small, D. (2008). Nowcasting: The real-time informational content of macroeconomic data. *Journal of Monetary Economics*, (55):665–676.

Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning*. 2 edition.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Hopfield, J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, 79:2554–2558.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R*. Springer.

Karlsson, S. (2012). Forecasting with bayesian vector autoregressions. Working Papers 2012:12, Oerebro University, School of Business.

Kim, S., Shephard, N., and Chib, S. (1998). Stochastic volatility: likelihood inference and comparison with arch models. *Review of Economic Studies*, 65:361–393.

Legrand, R. (2019). Time-varying vector autoregressions: Efficient estimation, random inertia and random mean. Working Papers 95707, MPRA.

Litterman, R. (1986). Forecasting with bayesian vector autoregressions: Five years of experience. *Journal of Business And Economic Statistics*, 4(1):25–38.

Primiceri, G. (2005). Time-varying structural vector autoregressions and monetary policy. *Review of Economic Studies*, 72:821–852.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408.

Rumelhart, D., Hinton, G., and Williams, R. (1986). Learning representations by back-propagating errors. *Nature*, 323:533–536.

Schorfheide, F. and Song, D. (2015). Real-time forecasting with a mixed-frequency var. *Journal of Business and Economic Statistics*, 33(3):366–380.

Sims, C. (1980). Macroeconomics and reality. *Econometrica*, 48(1):1–48.

Warne, A., Droumaguet, M., and Woźniak, T. (2015). Granger causality and regime inference in bayesian markov-switching VARs. Working Papers 1794, European Central Bank.

Appendix

A.1 Now-casting.com report

Please see next page for the beginning of the report.

Now-casting.com

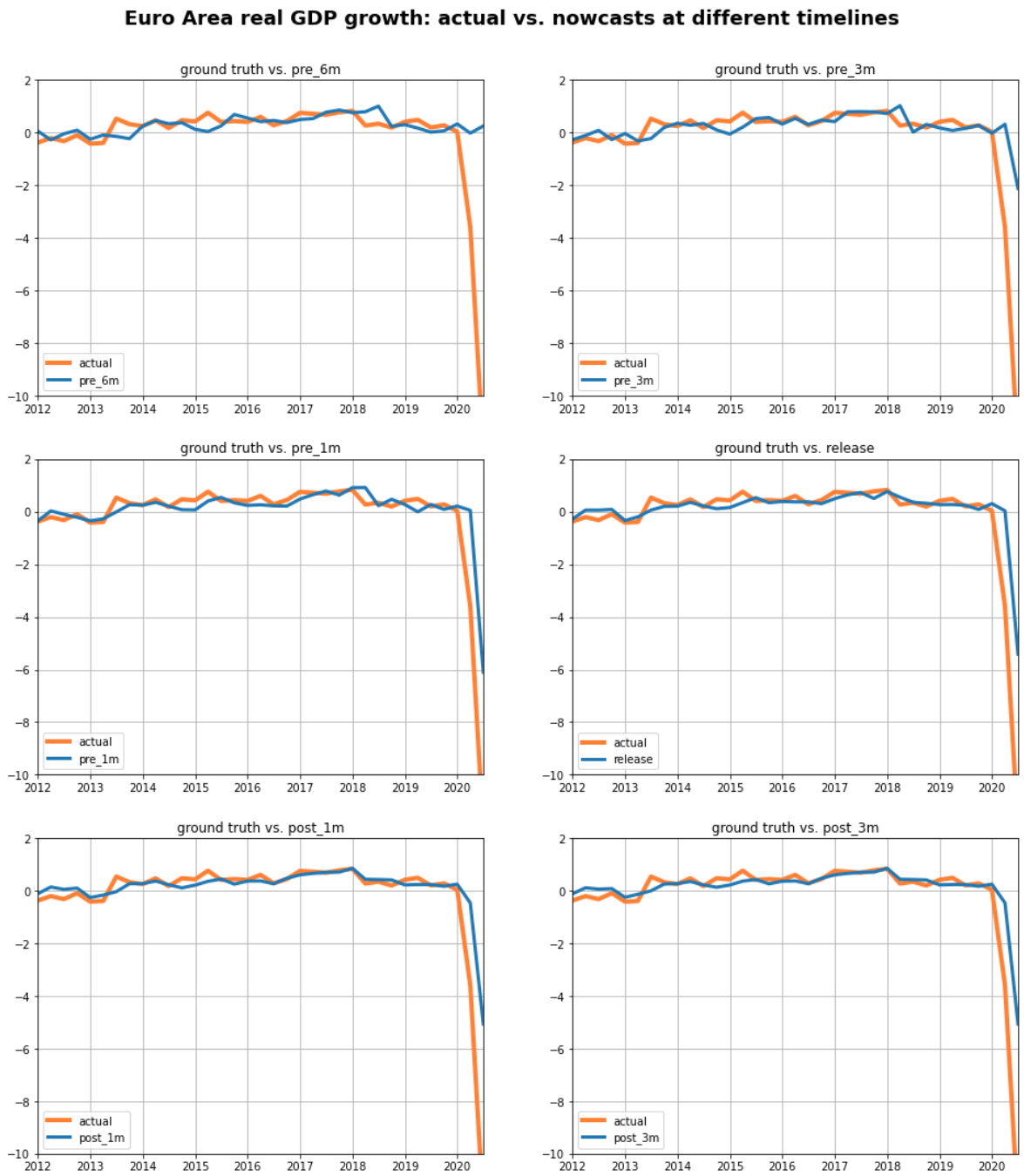
Objectives

- provide a brief summary of our exploratory analysis of the nowcasts provided by now-casting.com.
- point out the series that seem to work well, and the one for which the nowcasts may look less efficient.

Euro area, GDP growth

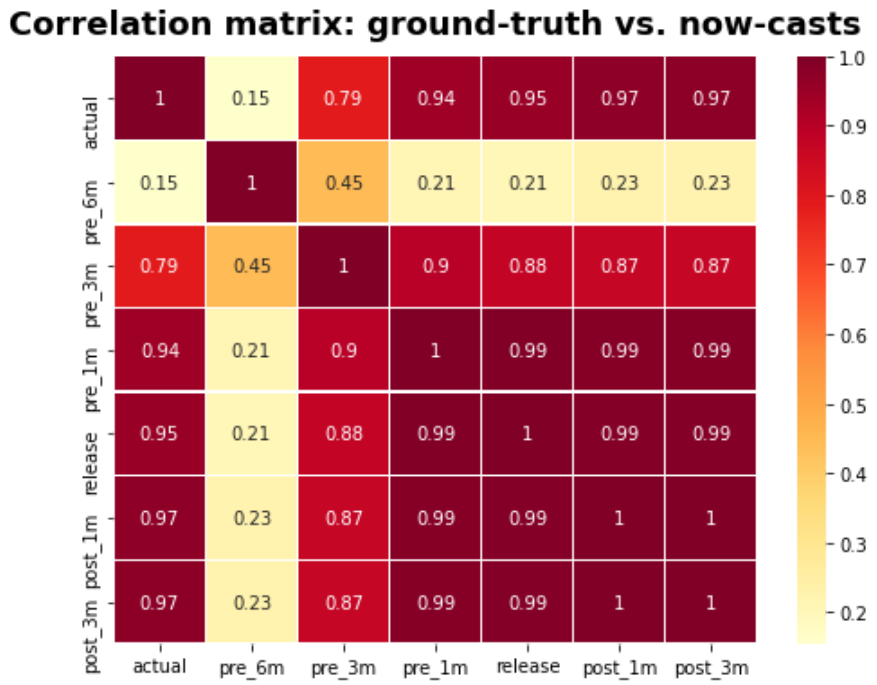
- we use simple visualizations and statistical measures to obtain preliminary assessments of the quality of the forecasts.
- a ground-truth or "actual" series is used for comparison with the nowcasts. It is real GDP growth (Q-to-Q) from OECD.
- first the actual value is plotted against the nowcasts at different timelines (6 months before release, 3 months before release, 1 month before release, at release, 1 month after release, 3 month after release).
- basic correlations are also calculated between the actual series and the different nowcasts.
- finally, the average root mean square forecast error (RMSFE) is estimated for the different nowcasts timelines (averaging is realised over the sample periods).

At first, compare the nowcasts at different timelines with the actual values.

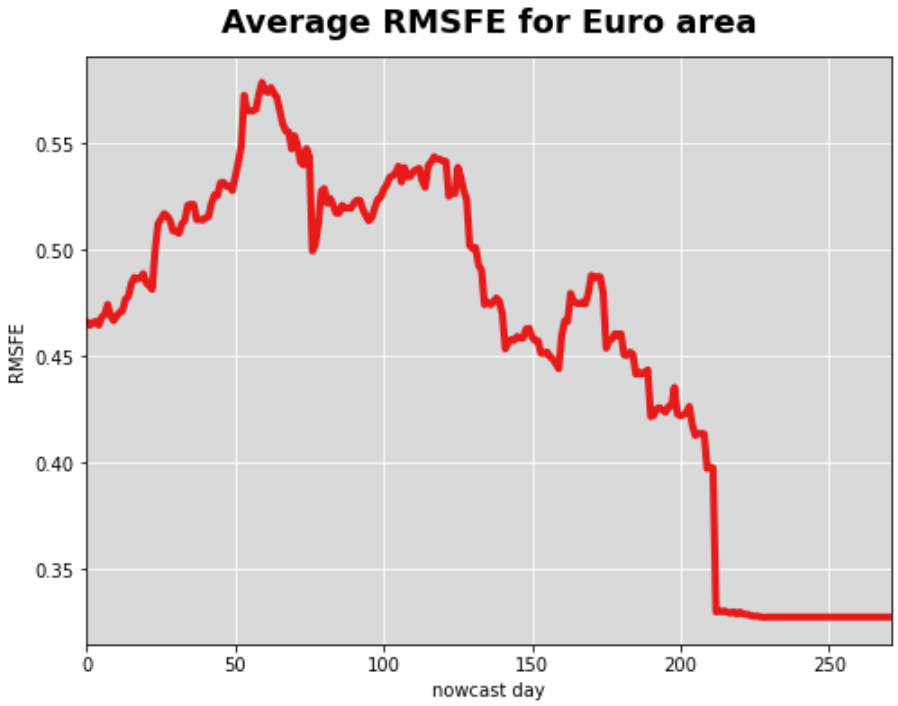


Overall, the nowcasts look quite good. The latest ones look really close to the actual values.

Then, basic correlations are computed between the nowcasts and the ground truth series.



The correlations between the actual series and the nowcasts also look good. The correlation for the earliest nowcast is low (0.15 for 6 month before release) but then increases quickly to more than 0.95.

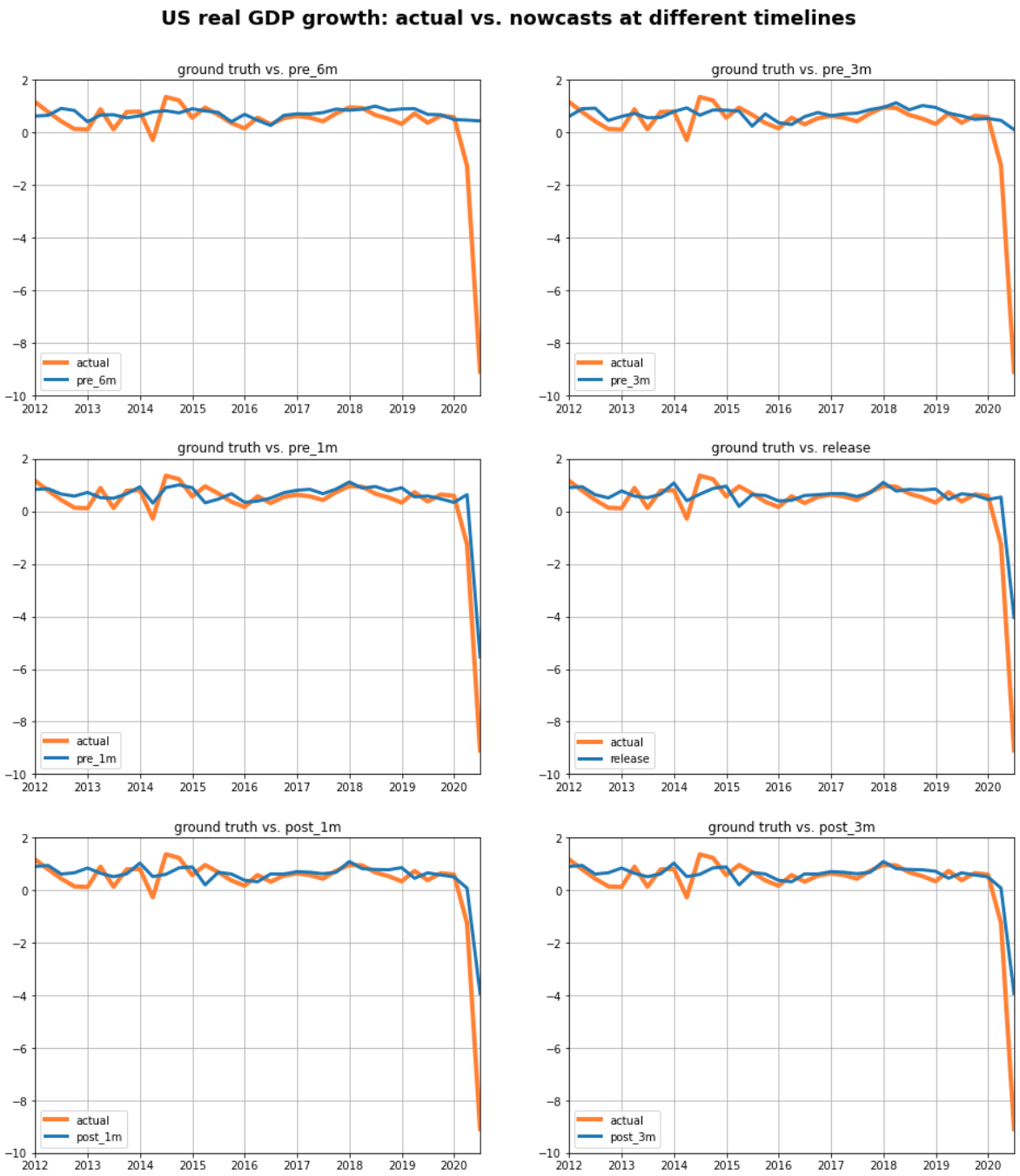


Finally, the RMSFE is plot (using the OECD GDP series as reference for computinfg the error). This plot replicates closely the one provided in the information materials presentation of Now-casting.com, slide 14. The initial rise in RMSFE observed here is due to the fact nowcasts are considered earlier (6 months before release against 3 month only in the presentation). Aside from this initial increase in RMSFE, the RMSFE steadily decreases, as expected when more information arrives.

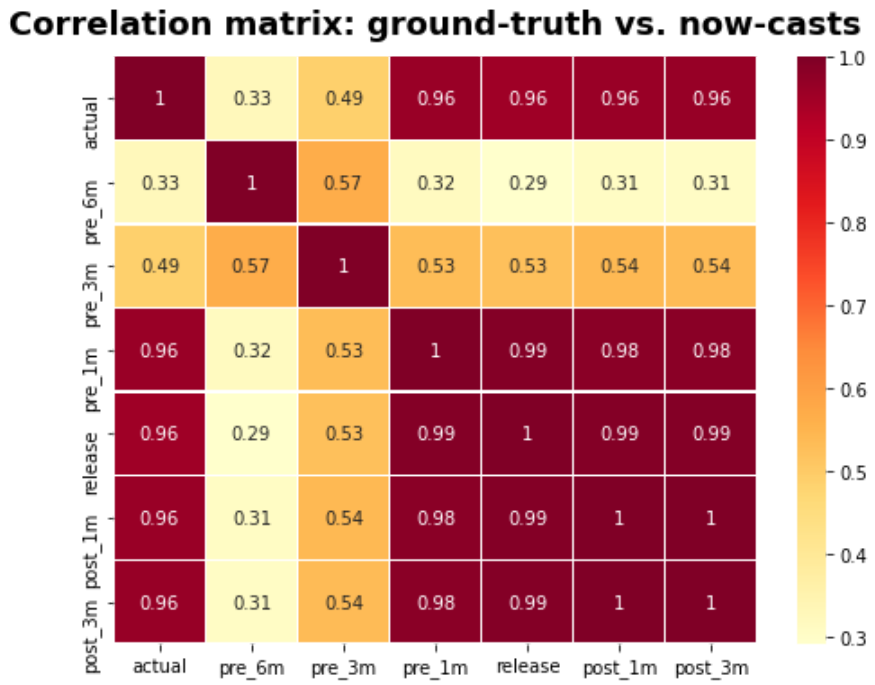
Overall, the results for the Euro Area are quite convincing!

United States, GDP growth

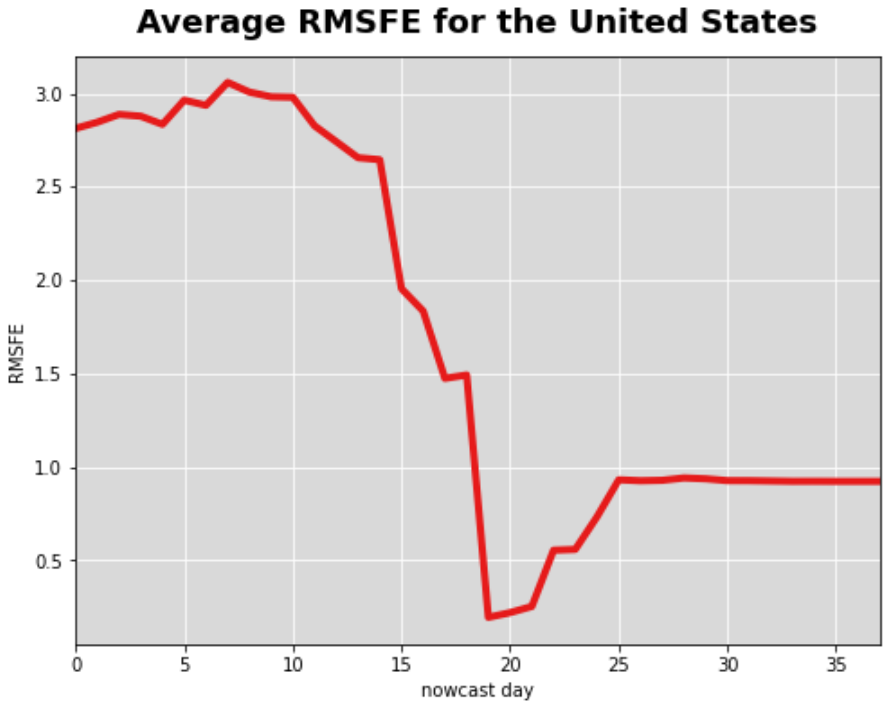
The same exercise is conducted for the United States.



The nowcasts seem again to be fairly close to the actual values, though they look somewhat smoother. The earliest forecasts are the least convincing and apparently fail to adequately capture the COVID. It also noteworthy that the nowcasts one month before release capture the COVID better than the ones 3 months after release.

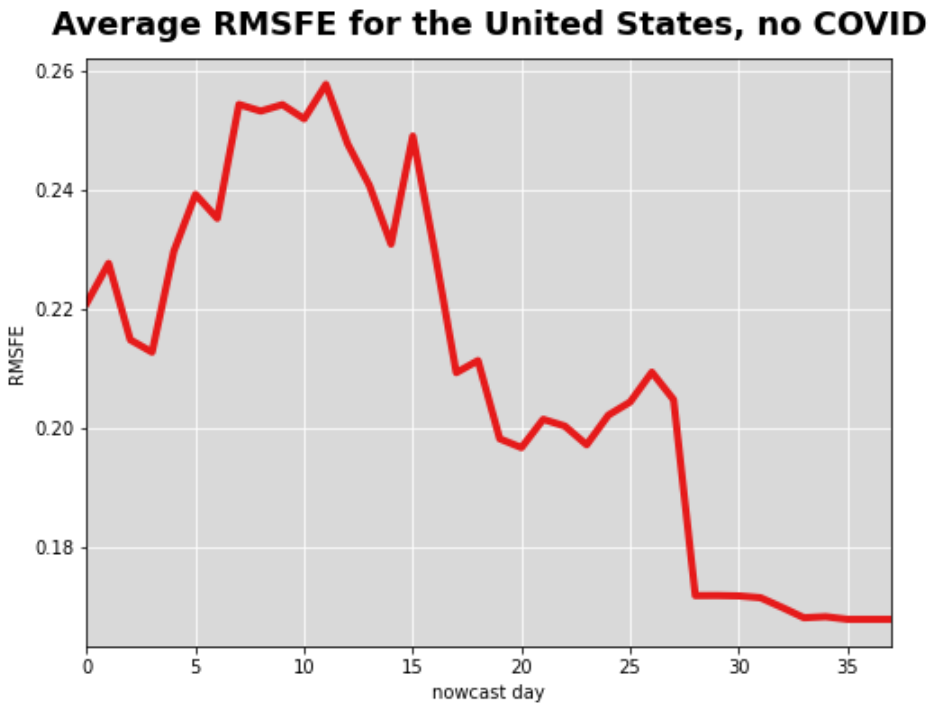


The pattern is similar to that of the Euro Area: earlier forecasts display low correlation, while later ones display really high correlations.



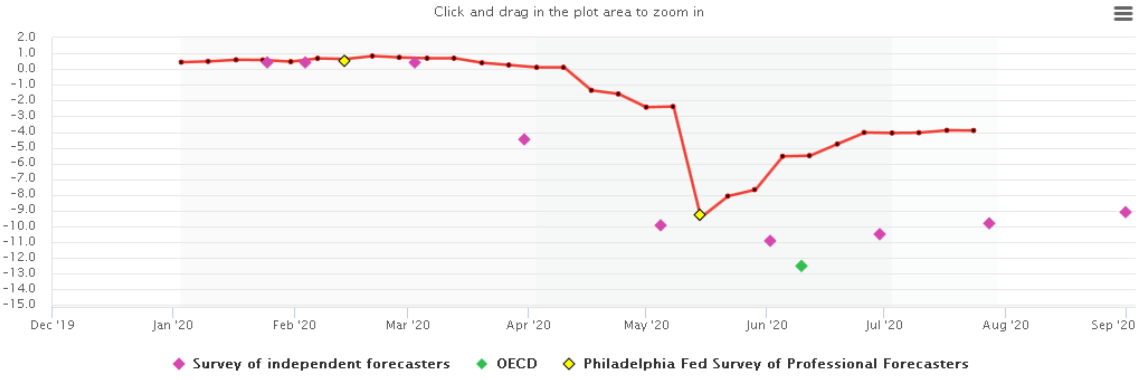
When looking at RMSFE, the pattern becomes unexpected. The nowcast RMSFE is first steadily declining, then getting back up roughly one month before release. This goes against the intuition that more information should produce more accurate forecasts.

After some more exploration, it seems that this surprising result might be - at least partially - due to the COVID. Indeed, by just excluding 2020q2 from the sample, the plot changes significantly:



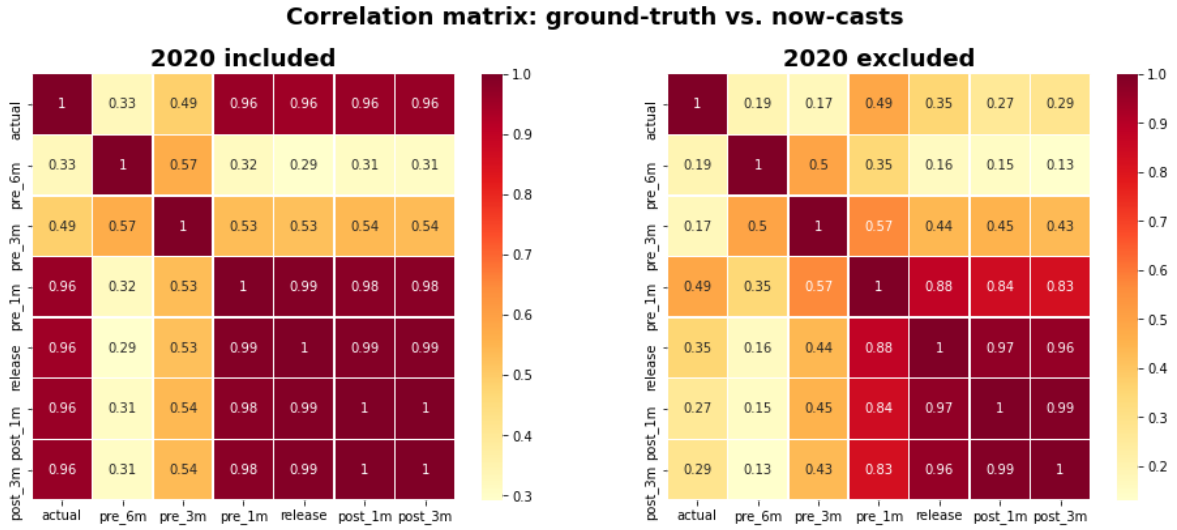
Now the RMSE steadily declines without rising again anymore at the end of the sample. It thus seems that the COVID represents a large outlier that it is sufficient to affect the average RMFSE.

This conclusion is in line with the visualisation of the COVID period on the now-casting.com website for the United States:

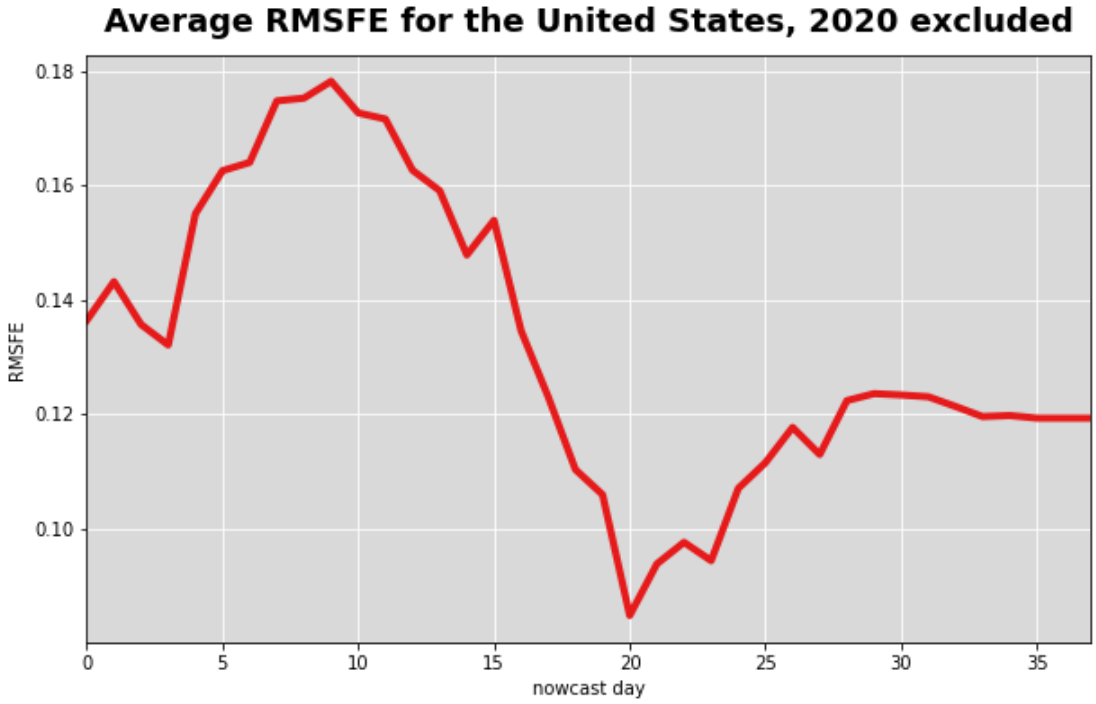


Overall, the value of -9% produced by the nowcasts in May seems closer to the ground truth than the value of -4% predicted by the updated nowcasts in July/August. Other surveys also seem to disagree with the nowcasts here, predicting rather a decline around -10%. This suggests that the model may struggle to account for the COVID effect.

Still, aside from the COVID, the US does exhibit some undesirable results. If instead of excluding just the COVID (2020q2) one excludes the whole year 2020 (hence 2020q1 and 2020q2), the results become poor again. The correlations become fairly low:



When excluding 2020 from the sample, the correlations between the actual data and the nowcasts become quite low. Also, the correlation actually declines for the most recent nowcasts. The best forecasts are the one obtained 1 month before the release of the data.

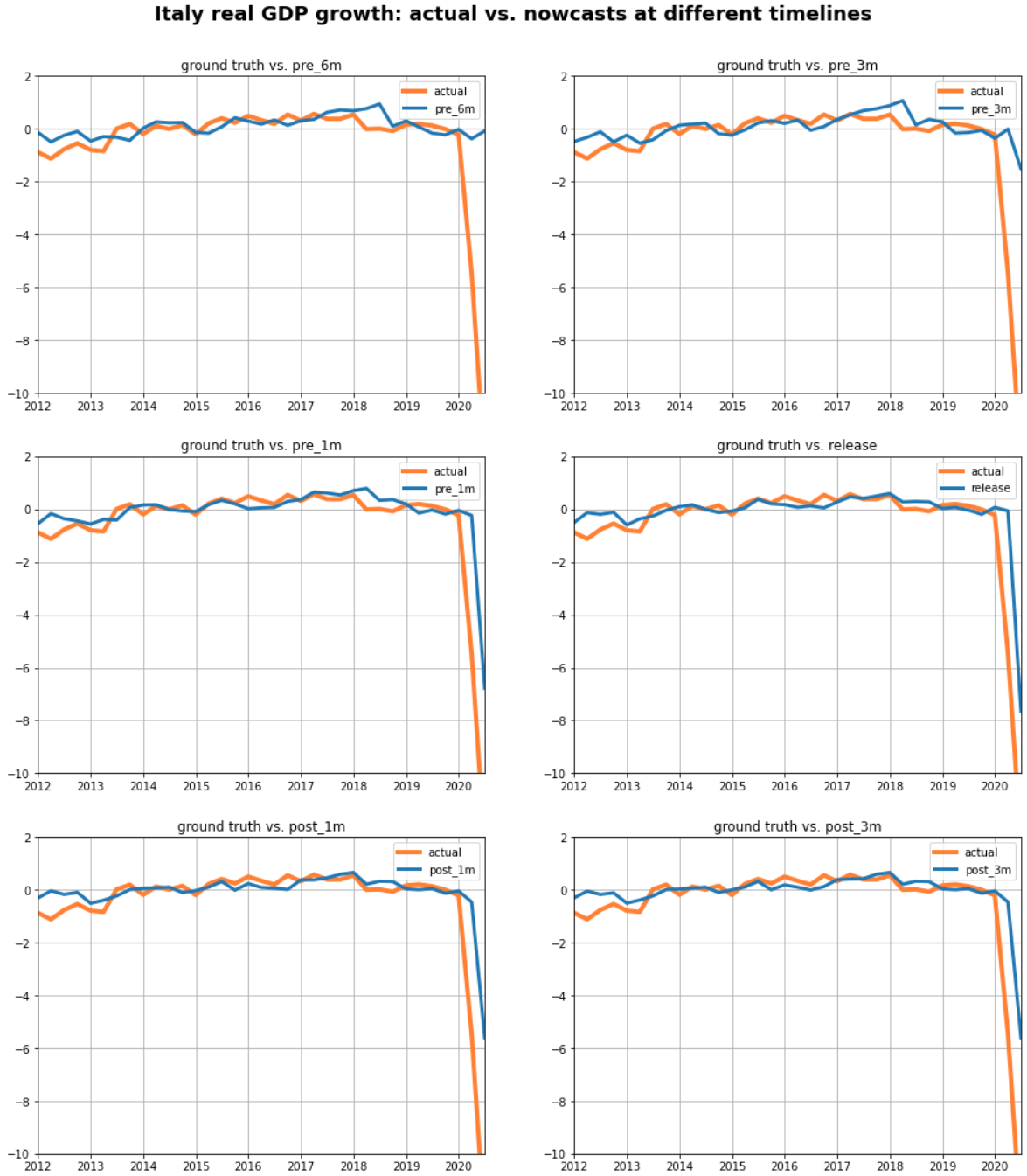


Eventually, one notices that the RMSE starts rising again at the end of the nowcast timeline. This should not happen if more information results in more accurate predictions.

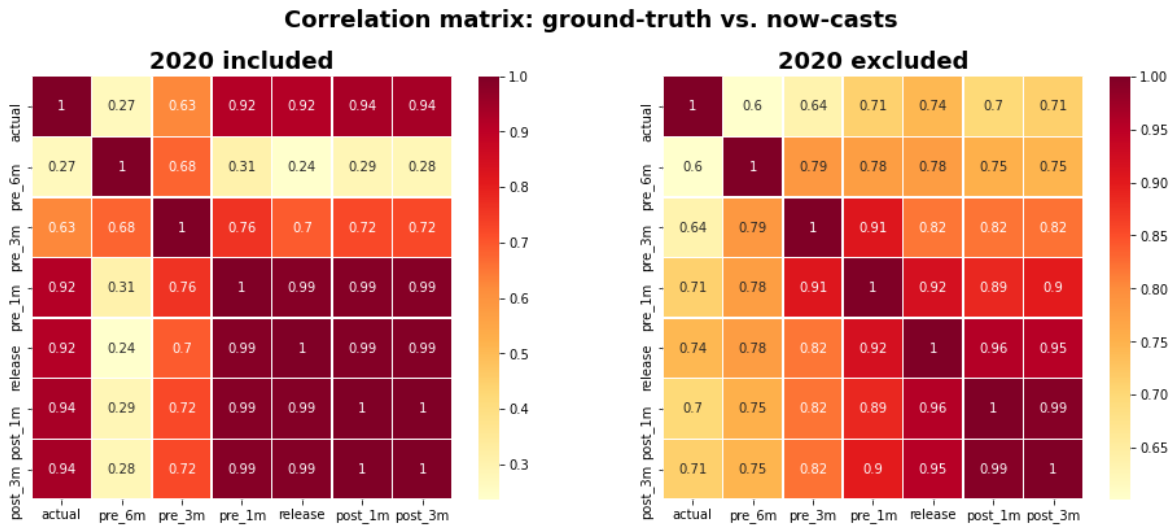
Other European Union countries, GDP growth

Overall, we found that most other developed economies (Japan, France, Germany, Spain and the United Kingdom) behave in a way that is fairly similar to the Euro Area: the nowcast series look close to the actual series, they display a fairly high correlation (at least for the later nowcasts), and the calculated RMSFE steadily decline as more information obtains. Italy, however, looks less convincing.

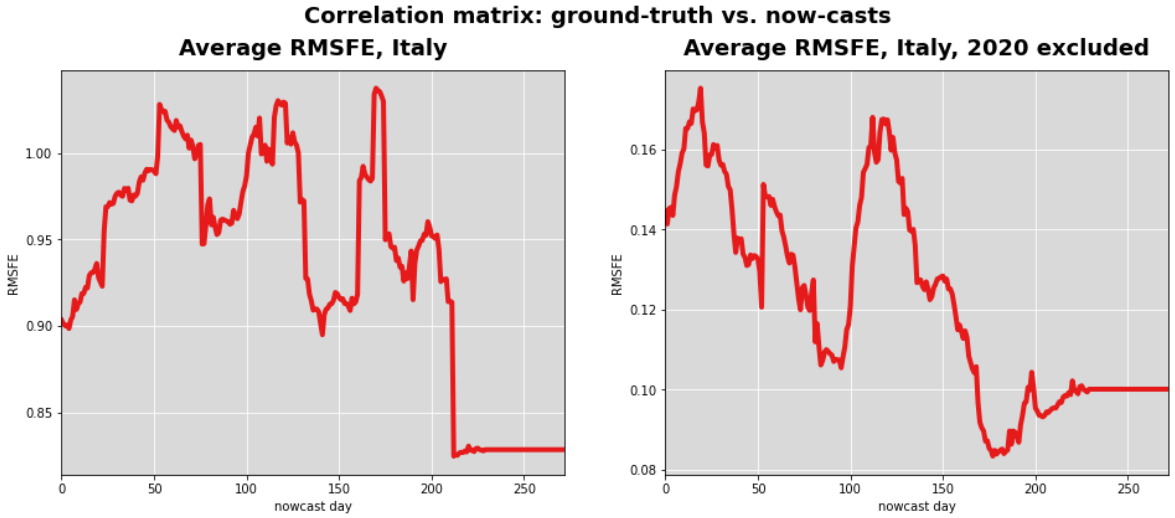
A focus on Italy, GDP growth



While the plots still reasonably approach the actual values, the model seems to do less well close to te COVID period: not only do the nowcasts seem to be lagging behind the actual values, but also the latest forecasts seem less accurate than those produced one month before release and at release.



When 2020 is excluded, the correlation does not seem to improve much as later nowcasts are produced. It also remains capped at 0.7, way beow the 0.95 level of the full sample.



For the full sample (left plot), the RMSFE behave strangely. It is overall declining, but reverts to higher values quite a few times, unlike what would be expected as more information becomes available. It starts dropping for good only after the first release.

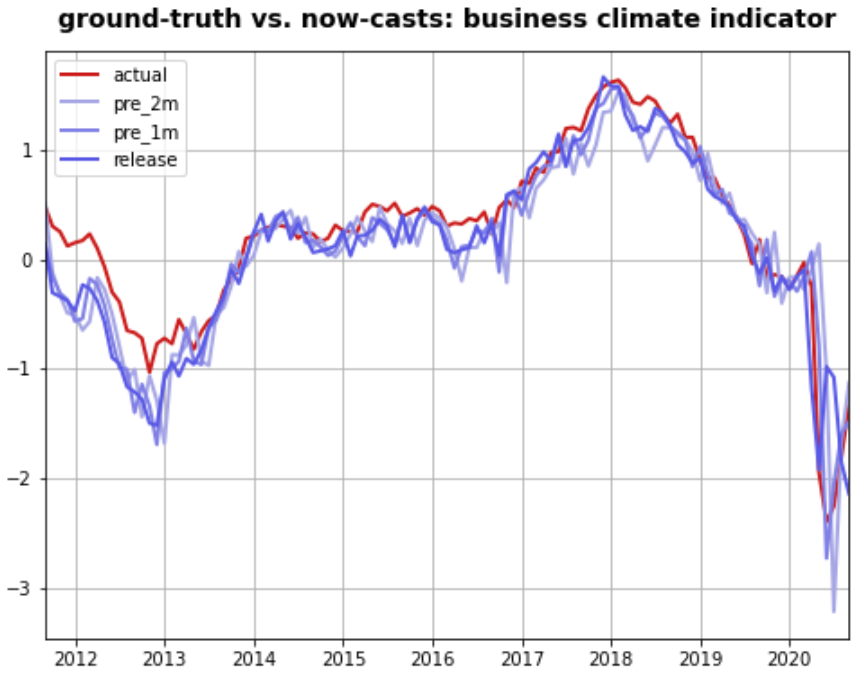
When 2020 is excluded from the sample, the RMSFE looks even worse. It starts increasing again at the date of release, before it reaches a plateau roughly two months after release. The increase following the release is clearly at odd with the expected improvement as more information gets available.

Analysis on other series

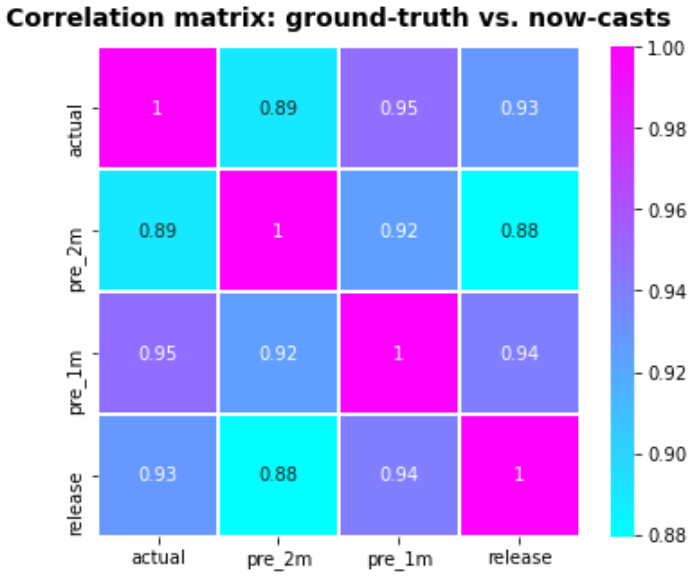
Now-casting.com also provided a number of non GDP series for the Euro Area. To obtain a broader overview of the dataset, some of these series were explored, with a similar methodology. Some yielded good results overall, while some produced unexpected results. A selection of these results are developed below.

business climate indicator

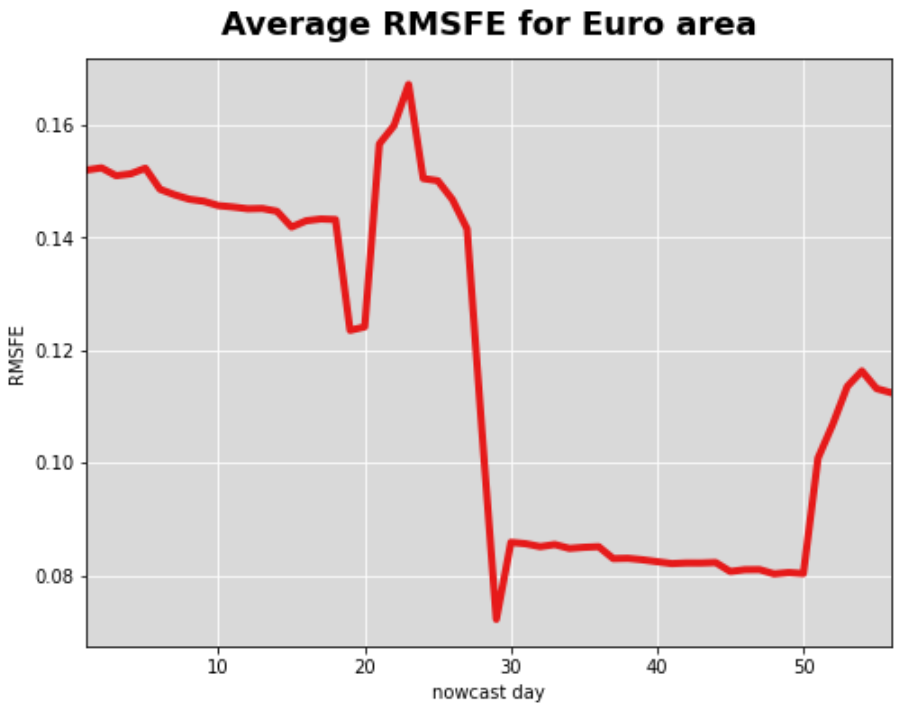
This is an example of a series for which the model works well.



The nowcasts follow the actual series quite closely.



The correlations are extremely high, showing a good fit of the nowcasts to the data.



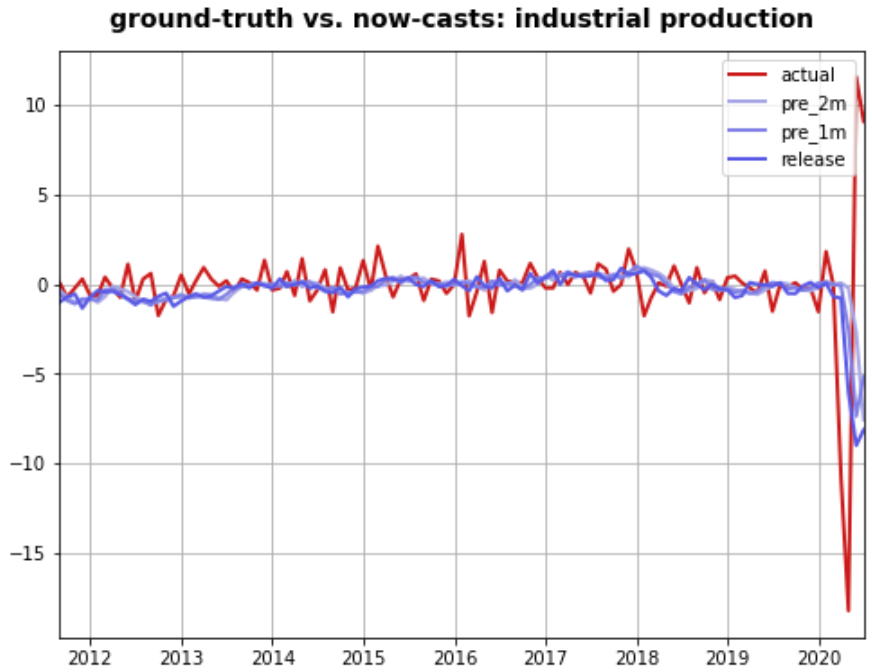
The RMSFE is overall declining except over the final periods.

Overall, the model seems good at predicting this series. Other series like construction output or manufacturing turnover also display the same kind of nice behaviour.

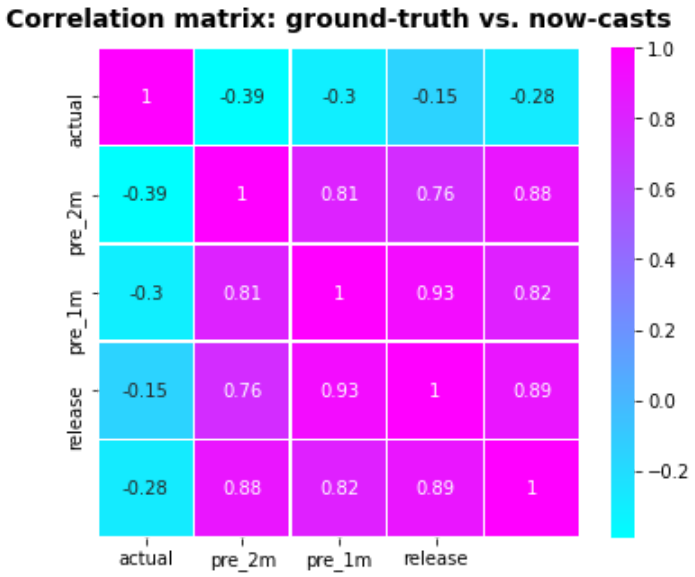
Certain series however produce less satisfactory results. Hereafter, two examples are developed: industrial production and car registrations.

Industrial production

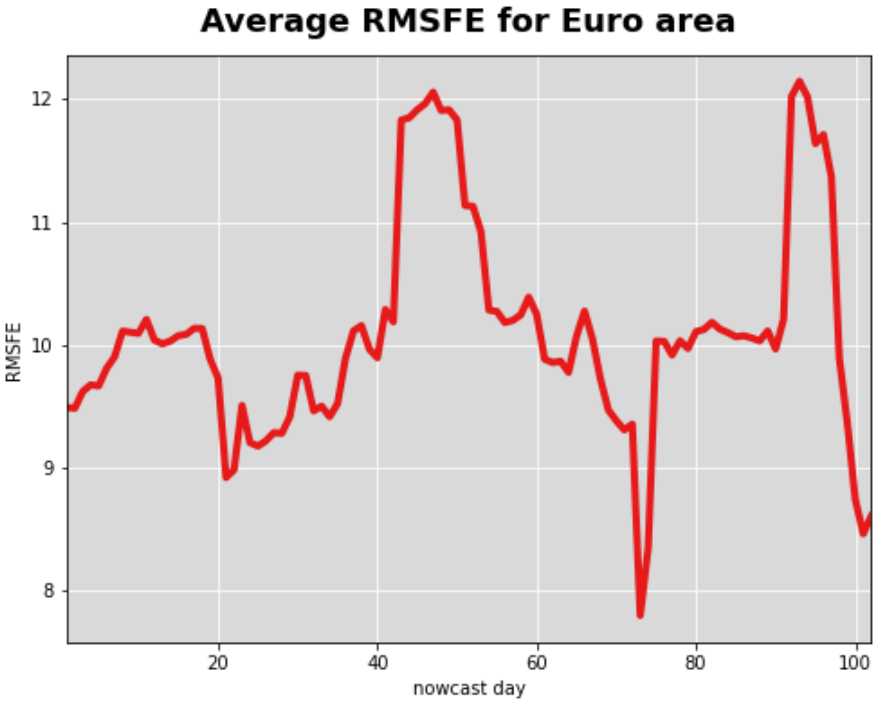
This is an example of a series where the model seems not to do too well.



The fit seems overall correct, but much smoother than the actual variations. Also, the nowcasts for the COVID period seem to be really lagging behind.



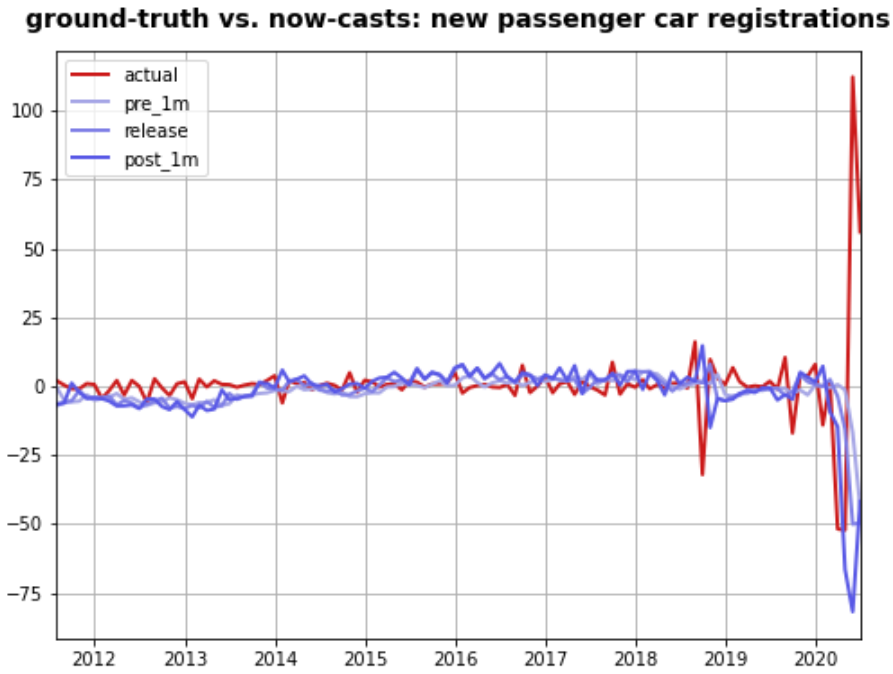
The correlations look really poor. At any timeline, the correlations between the actual series and the nowcasts are negative. This does not seem to improve as more information is obtained.



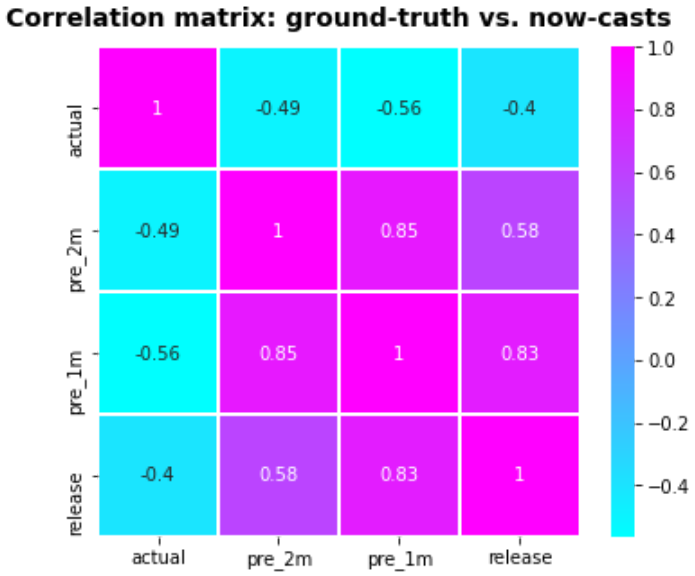
The RMSFE looks strange. It does overall decline slightly between the beginning and the end of the samples, but the reversion are so many and so large that it does not seem normal.

New passenger car registrations

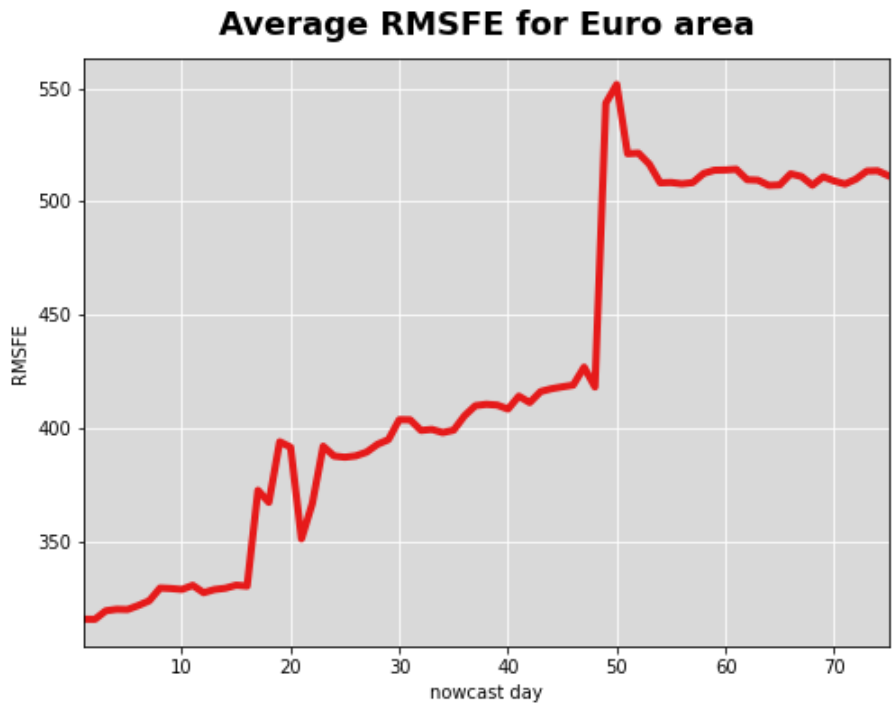
This is a final example of a series where things seem to go wrong.



The fit does not look extremely bad, but there seems to be some lag in the response of the nowcasts, which is especially obvious for the COVID period.



The correlations look poor. They remain significantly negative, even one month after the release.



The RMSFE is probably the most surprising result of the analysis. It is monotincally increasing, meaning that the prediction error increases with more information.

Conclusions

The nowcast models seems to work well for certain series. Except for the United States, the GDP forecasts look good for a wide range of countries.

Adding or excluding 2020 seems to affect the results for certain countries, especially those for which the model performance seems lower in general like the United States or Italy.

For a number of other variables, the performance seems mixed. Some variables achieve really good results, others display average performance, and some series present very unexpected behaviours.

We would be very interested in getting your feedback on these results. In particular, it would be nice if some intuitions could be provided about why certain series behave in an unexpected way.

Possibly, the COVID may represent a major factor of inefficiency. In this case, it would be informative as well to understand why other forecasts methods still manage to achieve fair prediction performances, while the nowcast models struggle a bit more with this event.

A.2 Transformations and sources of the monthly features

	feature	transformation	source
1	pmi	1	https://www.quandl.com/data/ISM/MAN_PMI-PMI-Composite-Index
2	business_outlook_survey	1	https://www.philadelphiafed.org/research-and-data/regional-economy/business-outlook-survey/historical-data
3	business_confidence_index	1	https://stats.oecd.org/#
4	consumer_confidence_index	1	https://stats.oecd.org/#
5	industrial_production	2	https://www.federalreserve.gov/datadownload/Choose.aspx?rel=g17
6	mgdp	2	https://ihsmarkit.com/products/us-monthly-gdp-index.html
7	business_sales	2	https://fred.stlouisfed.org/series/TOTBUSSMSA
8	new_residential_sales	2	https://www.census.gov/construction/nrs/historical_data/index.html
9	inventories	2	https://fred.stlouisfed.org/series/BUSINV
10	unemployment_rate	1	https://stats.oecd.org/#
11	employment	2	https://fred.stlouisfed.org/series/PAYEMS
12	weekly_hours	2	https://fred.stlouisfed.org/series/AWHMAN
13	hourly_earnings	3	https://fred.stlouisfed.org/series/AHETPI
14	consumer_credit	2	https://fred.stlouisfed.org/series/TOTALSL
15	personal_income	2	https://fred.stlouisfed.org/series/DSPIC96
16	federal_debt	2	https://fred.stlouisfed.org/series/MVGFD027MNFRBDAL
17	exports	2	https://www.census.gov/foreign-trade/balance/c0004.html
18	imports	2	https://www.census.gov/foreign-trade/balance/c0004.html
19	ppi	3	https://fred.stlouisfed.org/series/PPIACO
20	cpi	3	https://fred.stlouisfed.org/series/CPIAUCSL
21	monetary_base	2	https://fred.stlouisfed.org/series/BOGMBASE
22	bank_assets	2	https://www.federalreserve.gov/datadownload/Choose.aspx?rel=H8
23	bank_liabilities	2	https://www.federalreserve.gov/datadownload/Choose.aspx?rel=H8
24	mortgage_rate	1	https://fred.stlouisfed.org/series/MORTGAGE30US#0
25	federal_funds_rate	1	https://www.federalreserve.gov/datadownload/Choose.aspx?rel=H15
26	treasury_bill	1	https://www.federalreserve.gov/datadownload/Choose.aspx?rel=H15
27	treasury_bill_10	1	https://fred.stlouisfed.org/series/DGS10
28	effective_exchange_rate	2	https://fred.stlouisfed.org/series/NNUSBIS
29	spot_euro_us	2	https://www.federalreserve.gov/datadownload/Choose.aspx?rel=H10
30	nyse_composite_index	2	https://finance.yahoo.com/quote/%5ENYA/history?period1=-126316800&period2=1601424000&interval=1mo&filter=history&frequency=1mo
31	vix	2	https://fred.stlouisfed.org/series/VIXCLS

Figure 4: Transformations and sources of the monthly features

A.3 The Kalman filter and the Carter-Kohn algorithm

A.3.1 The Kalman filter

A general linear Gaussian dynamic model can be written in state-space form as:

Observation equation: $y_t = A_t z_t + \upsilon_t \qquad \upsilon_t \sim \mathcal{N}(0, \Upsilon_t)$

Transition equation: $z_t = B_t w_t + C_t z_{t-1} + \kappa_t \qquad \kappa_t \sim \mathcal{N}(0, K_t)$

where y_t denotes the observed variable, z_t the state or unobserved variable, and w_t the exogenous observed variable. A_t, B_t and C_t denote matrices of coefficients. υ_t and κ_t are shocks with respective variance-covariance matrices Υ_t and K_t.

Table 8 reports the state-space formulations for the project models: row 1 for the dynamic factor model of section 3.1, row 2 for the mixed frequency Bayesian VAR of section 3.3, rows 3-5 for the TV-BVAR of section 3.6.

	y_t	z_t	w_t	A_t	B_t	C_t	Υ_t	K_t
f	$x_t - \mu$	f_t	–	Λ	–	A	$diag(\psi)$	BB'
x	y_t	z_t	δ	Λ_t	1	Φ	–	Ω
β_i	$y_{i,t} + \delta'_{i,t}\varepsilon_{-i,t}$	$\beta_{i,t}$	$(1-\rho_i)b_i$	x_t	1	ρ_i	$s_i\ exp(\lambda_{i,t})$	Ω_i
λ_i	$\hat{y}_{i,t} - m_J$	$\lambda_{i,t}$	–	1	–	γ_i	v_J	ϕ_i
δ_i	$\varepsilon_{i,t}$	$\delta_{i,t}$	$(1-\alpha_i)d_i$	$-\varepsilon'_{i,t}$	1	α_i	$s_i\ exp(\lambda_{i,t})$	Ψ_i

Table 8: State-space representations for the dynamic parameters

Given a state-space representation, the Kalman filter is a simple procedure that derives conditional expectations of the unobserved dynamic parameters in a mostly mechanical way. To see this, introduce first the following notations:

$y_{t|s} = \mathbb{E}(y_t|y_1, \cdots, y_s) \qquad z_{t|s} = \mathbb{E}(z_t|y_1, \cdots, y_s) \qquad \Upsilon_{t|s} = var(y_t|y_1, \cdots, y_s)$

$K_{t|s} = var(z_t|y_1, \cdots, y_s) \qquad \tilde{z}_{t|s} = \mathbb{E}(z_t|z_s, y_1, \cdots, y_t) \qquad \tilde{K}_{t|s} = var(z_t|z_s, y_1, \cdots, y_t)$

By definition, this implies that:

$z_t|y_1,\cdots,y_s \sim \mathcal{N}(z_{t|s},K_{t|s}) \qquad z_t|z_s,y_1,\cdots,y_t \sim \mathcal{N}(\tilde{z}_{t|s},\tilde{K}_{t|s})$

Using basic properties of the normal distribution, one can then derive the following 6 steps wich constitutes the Kalman filter algorithm:

Algorithm A.1: the Kalman filter:

For $t = 1,\cdots,T$, do[3]:

step 1. state, prediction:	$z_{t\|t-1} = B_t w_t + C_t z_{t-1\|t-1}$
step 2. state, prediction error:	$K_{t\|t-1} = C_t K_{t-1\|t-1} C_t' + K_t$
step 3. observed, prediction:	$y_{t\|t-1} = A_t z_{t\|t-1}$
step 4. observed, prediction error:	$\Upsilon_{t\|t-1} = A_t K_{t\|t-1} A_t' + \Upsilon_t$
step 5. state, correction:	$z_{t\|t} = z_{t\|t-1} + \Phi_t (y_t - y_{t\|t-1})$
step 6. state, prediction error correction:	$K_{t\|t} = K_{t\|t-1} - \Phi_t \Upsilon_{t\|t-1} \Phi_t'$
with: $\Phi_t = K_{t\|t-1} A_t' \Upsilon_{t\|t-1}^{-1}$	

This simple algorithm sequentially derives the mean and variance ($z_{t|t}$ and $K_{t|t}$) of the unoberved state variable for each sample period.

A.3.2 The Carter-Kohn algorithm

The simple Kalman filter is suitable in a frequenctist approach. In a Bayesian context however, one must derive the full posterior distribution of the state variable (jointly for all sample periods), conditional on the observed variables. Formally, one seeks to derive:

$\pi(z_1,z_2,\cdots,z_T|y_1,y_2,\cdots,y_T) = \pi(z|y)$

To do so, one notes first that this joint posterior can be rewritten as:

$$\pi(z|y) = \pi(z_T|y_T) \prod_{t=1}^{T-1} \pi(z_t|z_{t+1},y_t)$$

[3]For the initial period $t = 1$, the first two steps are slightly different; they become $z_{1|0} = B_1 w_1$ and $K_{1|0} = K_1$, respectively.

Therefore, to obtain a draw from $\pi(z|y)$, it is sufficient to sample a value from $\pi(z_T|y_T)$, then sample recursively values from $\pi(z_t|z_{t+1},y_t)$, for $t = T-1, T-2, \cdots, 1$. This supposes yet that one can first recover the distributions $\pi(z_T|y_T)$ and $\pi(z_t|z_{t+1},y_t)$. Carter and Kohn (1994) notice that this is easily done thanks to the Kalman filter. Concretely, the authors propose a two-step procedure. The first step constitutes the forward pass of the algorithm, which is just a regular Kalman filter. The second step represents the backward pass, which obtains the distributions $\pi(z_t|z_{t+1},y_t)$ from the forward pass elements.

The full algorithm is as follows:

Algorithm A.2: the Carter-Kohn algorithm:

1. Apply the Kalman filter: for $t = 1, \cdots, T$, do (forward pass):

   step 1. state, prediction: $z_{t|t-1} = B_t w_t + C_t z_{t-1|t-1}$

   step 2. state, prediction error: $K_{t|t-1} = C_t K_{t-1|t-1} C_t' + K_t$

   step 3. observed, prediction: $y_{t|t-1} = A_t z_{t|t-1}$

   step 4. observed, prediction error: $\Upsilon_{t|t-1} = A_t K_{t|t-1} A_t' + \Upsilon_t$

   step 5. state, correction: $z_{t|t} = z_{t|t-1} + \Phi_t (y_t - y_{t|t-1})$

   step 6. state, prediction error correction: $K_{t|t} = K_{t|t-1} - \Phi_t \Upsilon_{t|t-1} \Phi_t'$

   with: $\Phi_t = K_{t|t-1} A_t' \Upsilon_{t|t-1}^{-1}$

2. Sample z_T from $\pi(z_T|y_T) \sim \mathcal{N}(z_{T|T}, K_{T|T})$.

3. For $t = T-1, \cdots, 1$, apply the following steps recursively (backward pass):

   step 1. state, correction: $\tilde{z}_{t|t+1} = z_{t|t} + \Xi_t (z_{t+1} - z_{t+1|t})$

   step 2. state, prediction error correction: $\tilde{K}_{t|t+1} = K_{t|t} - \Xi_t C_t K_{t|t}$

   with: $\Xi_t = K_{t|t} C_t' K_{t+1|t}^{-1}$

   step 3. sampling: $\pi(z_t|z_{t+1},y_t) \sim \mathcal{N}(\tilde{z}_{t|t+1}, \tilde{K}_{t|t+1})$.

This provides a series of draws from $\pi(z_1|z_2,y_1), \pi(z_2|z_3,y_2), \cdots, \pi(z_{T-1}|z_T,y_{T-1}), \pi(z_T|y_T)$ which jointly constitute a draw from $\pi(z|y)$.