

Deep Vector Autoregression for Macroeconomic Data

Marc Agustí (marc.agusti@barcelonagse.eu)

Patrick Altmeier (patrick.altmeyer@barcelonagse.eu)

Ignacio Vidal-Quadras Costa (ignacio.vidalquadrascosta@barcelonagse.eu)

June, 2021

Abstract

Vector autoregression (VAR) models are a popular choice for forecasting of macroeconomic time series data. Due to their simplicity and success at modelling the monetary economic indicators VARs have become a standard tool for central bankers to construct economic forecasts. Impulse response functions can be readily retrieved and are used extensively to investigate the monetary transmission mechanism. In light of the recent advancements in computational power and the development of advanced machine learning and deep learning algorithms we propose a simple way to integrate these tools into the VAR framework. This paper aims to contribute to the time series literature by introducing a ground-breaking methodology which we refer to as Deep Vector Autoregression (Deep VAR). By fitting each equation of the VAR system with a deep neural network, the Deep VAR outperforms the VAR in terms of in-sample fit, out-of-sample fit and point forecasting accuracy. In particular, we find that the Deep VAR is able to better capture the structural economic changes during periods of uncertainty and recession.

Keywords— Vector Autoregression, Deep Learning, Forecasting, Neural Networks, Macroeconomic Timeseries

1 Introduction

As stated by the European Central Bank, the monetary transmission mechanism is the process through which monetary policy decisions affect the economy in general and the price level in particular. Uncertainty with respect to this transmission is generally huge, given that it is characterized by long, variable and uncertain dependencies through time and variables. Hence, it is typically challenging to predict how changes in monetary policy actions affect real economic outcomes. It is therefore of foremost importance for policy-makers to use adequate tools to model the underlying mechanisms.

With this in mind, a lot of research on the forecasting of time series has been developed to assess the effect of current policy decisions on future economic variables. Thanks to this, over the last decades policy makers have had more information when taking decisions. This information usually comes in the form of point estimates and interval forecasts. To come up with these estimates, several methodologies have been developed and applied in the time series forecasting literature.

At the time of writing, one the most common methodologies to produce these estimates is the so-called Vector Autoregression (VAR). This framework, which belongs to the traditional toolkit of econometric forecasting techniques, has been shown to provide policy-makers with fairly good and consistent point and interval estimates. It has therefore been used extensively in the monetary policy divisions of central banks.

Simultaneously, with the recent advancements in computational power, and more importantly, the development of advanced machine learning algorithms and deep learning, interesting novel tools have become available that may be useful for forecasting time series. Whereas the good performance of techniques such as VAR is well established, it is still uncertain whether deep learning algorithms can be applied successfully to macroeconomic data.

To this end, this paper contributes a new and ground-breaking methodology that combines the VAR equation-by-equation structure with deep learning. We provide evidence that this improves the model's capacity to capture potentially highly non-linear relationships in the underlying data generating process. The primary objective of this paper is to develop a methodology that produces improved modelling outcomes while deviating as little as possible from the established VAR framework, thereby keeping things straight-forward and familiar to economists. We show that the existing VAR methodology can be easily extended to the broader class of Deep VAR models and provide solid empirical evidence that Deep VARs consistently outperform the conventional approach.

To the best of our knowledge, this is the first paper to propose a Deep VAR framework of this structure, namely, to fit a deep neural network for each equation of the VAR process. Although previous work has explored the use of deep learning to forecast macroeconomic time series, previous proposed methodologies deviate more from the conventional VAR framework. For example, Verstyuk (2020) chooses to model the whole system through one unified deep neural network. We find that the equation-by-equation approach not only helps to maintain interpretability and simplicity, but also appears to produce better modelling outcomes. To enable researchers and practitioners to easily implement our proposed methodology, we have developed a unified framework for estimating Deep VARs in R and plan to continue its development going forward.

We find that the Deep VAR methodology outperforms the traditional VAR framework in terms of in-sample and out-of-sample fit as well as with respect to forecasting accuracy. In particular, the Deep VAR appears to be better at capturing non-linear dynamics underlying the time series process. It therefore leads to consistently lower modelling errors than the VAR, especially during periods of

economic downturn and uncertainty.

Arguably policy makers are not only interested in the forecasting accuracy of the model, but are typically also concerned with inference. For example, central banks are often interested in knowing to what extent interest rates granger cause other variables within the monetary transmission mechanism. Another aspect policy makers and researchers care about is how the variables of the system evolve through time in response to innovations. This information is typically recovered using Impulse Response Functions (IRFs). The linear additive modelling assumption underlying the conventional VAR makes inference straight-forward. In the case of Deep VARs inference is arguably more complicated, though promising avenues have recently been explored (Verstyuk 2020). We believe that the methodology proposed in this paper can be relatively easily augmented to the inference realm in future work.

The remainder of the paper is structured as follows: in section 2 we present a literature review of prior research on the methodologies used to provide forecasts and on the monetary transmission mechanism in general. Section 3 provides a detailed description of the data we use for our empirical exercises. In section 4 we present the traditional VAR methodology and develop our proposed Deep VAR model. Sections 5 and 6 present our empirical findings and possible extensions and caveats, respectively. Finally, section 7 concludes.

2 Literature review

There is broad agreement among economists on the fact that monetary policy affects economic activity in the short and medium term. Friedman and Schwartz (2008) found that monetary policy actions are followed by movements in real output that may last for two years or more (Romer and Romer (1989); Bernanke (1990)). The underlying forces that trigger these outcomes is of great interest to most economists. Central bankers in particular aim to understand the monetary transmission mechanism. If monetary policy affects the real economy, then what exactly is the transmission mechanism through which these effects occur? This is one of the questions which is among the most important and controversial topics in modern-day macroeconomics.

In the aftermath of the oil price shock in the 1970's, interest emerged in understanding business cycles. To this end economists initially made use of large-scale macroeconomic models, which was criticized by Lucas Jr (1976), stating that the assumption of invariant behavioural equations was inconsistent with dynamic maximizing behaviour. Hence, **New Classical** economists started to make use of so-called market clearing models of economic fluctuations. With the goal of really taking into account productivity shocks, **Real Business Cycle** models were developed (Kydland and Prescott (1982)).

Following the failure of large-scale macroeconomic models when trying to predict business cycles, the economic profession resorted to structural vector autoregressive (VAR) models to analyse business cycles, which proved to be useful for capturing the impact of policy actions. Sims and others (1986) suggested that VARs were an efficient tool to evaluate macroeconomic models. One of the advantages of VARs is their simplicity, which makes it easy to estimate and interpret them.

Yet, this simplicity comes at a cost: conventional VARs are typically not able to capture non-linear relationships in the data, which might be a significant limitation. In the very short run many time series can be expected to behave more or less according to their past and a linear model may

be efficient to capture dynamics, but for longer term dependencies this is typically not the case. With respect to the economic time series that form part of the monetary transmission mechanism, specifically output, inflation, interest rates and labour market variables, non-linear dependencies are likely to form part of the data generating process as shown by Brock et al. (1991). This is true in particular during times of abrupt and significant economic fluctuations.

During past years, economists have therefore started to add non-linear techniques to their forecasting tool kit. Machine Learning has contributed a lot to this field of research. Some of the most popular machine learning techniques which do not assume a linear relationship between inputs and outputs include K-Nearest Neighbours (first introduced by Fix and Hodges (1951)), Support Vector Machines (mostly developed by Cortes and Vapnik (1995)), Random Forests (first introduced in 1995 by Ho (1995)) and Deep Artificial Neural Networks (first proposed in 1943 by McCulloch and Pitts (1990)). The latter have been explored previously in the realm of time series forecasting (Hamzaçebi (2008), G. P. Zhang (2003), Kihoro, Otieno, and Wafula (2004)). Neural networks are non-parametric models that have been shown to be particularly successful at capturing non-linearities (G. Zhang, Patuwo, and Hu (1998), G. P. Zhang (2003)).

A particular subclass of neural networks used primarily for sequential data are recurrent neural networks (RNN). RNNs propagate previous outputs recursively allowing the model to learn persistent dependencies and thereby making them very efficient for time series data (Dorffner 1996). A recent staff working paper published by the Bank of England provides some empirical support for the argument deep learning can be successfully applied to macroeconomic data (Joseph et al. 2021). The authors run a horse race for forecasting inflation across different time horizons comparing the performance of linear and non-linear models. They find that neural networks in particular and also other common machine learning algorithms are useful for forecasting at particularly at a longer horizon given that.

3 Data

To study the monetary transmission mechanism we used a balanced panel of monthly US data which spans the period of January 1959 through March 2021. In order to explain the monetary transmission mechanism, the VAR literature relies on different indicators such us output and income, industrial new orders and turnover, retail sales and turnover, building permits, employment, consumption, price indices, exchange rates, short and long term interest rates, stock price indices, money and credit quantity aggregates, balance of payments and external trade, confidence indicators, and some foreign variables such as output, prices, interest rates, and stock markets from other countries used as proxies for external real, nominal and monetary influences.

On this paper we relied on four of the macroeconomic indicators mentioned above; output, price indices, employment and rates. For each indicator we chose a time series as its proxy. We can see the proxies used for each variable below.

- *output*: Industrial Production (IP)
- *price indices*: Consumer Price Index (CPI)
- *unemployment*: Unemployment rate (UR)
- *rates*: US Federal Fund Rate (FFR)

Note that we use IP rather than GDP as a proxy for output because for the case of the GDP only quarterly data was available.

As consequence of working with monthly data, we worked with 745 observations, each of them with the four variables previously mentioned.

The data is taken from the Economic Research department of the Federal Reserve Bank of Saint Louis (FRED). In a recent paper, they have created a monthly database for economic research (FRED-MD), a macroeconomics data base which consists of 134 monthly US indicators. The data is automatically updated each month and can be downloaded freely, facilitating the replication of results and comparing results.

Another strength of using the FRED-MD is the fact that the data is already preprocessed. To see how it has been preprocessed see the Methodology section.

4 Methodology

In conventional Vector Autoregression (VAR) dependencies of any system variable on past realizations of itself and its covariates are modelled through linear equations. This corresponds to a particular case of the broader class of Deep Vector Autoregressions investigated here and will serve as the baseline for our analysis.

4.1 Vector Autoregression

Let \mathbf{y}_t denote the $(K \times 1)$ vector of variables at time t . Then the VAR(p) with p lags and a constant deterministic term is simply a linear system of stochastic equations of the following form:

$$\mathbf{y}_t = \mathbf{c} + \mathbf{A}_1 \mathbf{y}_{t-1} + \mathbf{A}_2 \mathbf{y}_{t-2} + \dots + \mathbf{A}_p \mathbf{y}_{t-p} + \mathbf{u}_t \quad (1)$$

The matrices $\mathbf{A}_m \in \mathbb{R}^{K \times K}$, where $m \in \{1, \dots, p\}$, contain the reduced form coefficients and $\mathbf{u}_t \in \mathbb{R}^{K \times 1}$ is a vector of errors for which $\mathbb{E}\mathbf{u}_t$, $\mathbb{E}\mathbf{u}_t \mathbf{u}_t^T = \Sigma$ and $\mathbb{E}\mathbf{u}_t \mathbf{u}_s^T = \mathbf{0}$ for all $t \neq s$. We refer to (1) as the **reduced form** representation of the VAR(p) because all right-hand side variables are predetermined (Kilian and Lütkepohl 2017).

We can restate (1) more compactly as

$$\mathbf{y}_t = \mathbf{A} \mathbf{Z}_{t-1} + \mathbf{u}_t \quad (2)$$

where $\mathbf{A} = (\mathbf{c}, \mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_p) \in \mathbb{R}^{K \times (Kp+1)}$ and $\mathbf{Z}_{t-1} = (1, \mathbf{y}_{t-1}^T, \dots, \mathbf{y}_{t-p}^T)^T \in \mathbb{R}^{(Kp+1) \times 1}$. The expression in (2) demonstrates that the VAR(p) can be considered as a **seemingly unrelated regression** (SUR) model composed of individual regressions with common regressors (Greene 2012). In fact, it is useful to note for our purposes that the VAR(p) can be estimated efficiently through equation-by-equation OLS regression. In particular, it follows from (2) that

$$y_{it} = c_i + \sum_{m=1}^p \sum_{j=1}^K a_{jm} y_{jt-m} + u_{it} \quad , \quad \forall i = 1, \dots, K \quad (3)$$

which corresponds to the key modelling assumption that at any point in time t any time series $i \in 1, \dots, K$ is just a weighted sum of past realizations of itself and all other variables in the system. This assumption makes the estimation of VAR(p) processes remarkably simple. Perhaps more importantly, the assumption of linearity also greatly facilitates inference about VARs.

For implementation purposes it is generally more useful to estimate the VAR(p) through one single OLS regression. To this end let $\tilde{\mathbf{A}} = \mathbf{A}^{-1}$ and note that (2) can be restated even more compactly as

$$\mathbf{y} = \mathbf{Z}\tilde{\mathbf{A}} + \mathbf{u}_t \quad (4)$$

with $\mathbf{y} = (\mathbf{y}_1, \dots, \mathbf{y}_T)^T \in \mathbb{R}^{T \times K}$ and $\mathbf{Z} \in \mathbb{R}^{T \times (Kp+1)}$. Then the closed form solution for OLS is simply $\tilde{\mathbf{A}} = (\mathbf{Z}^T \mathbf{Z})^{-1} \mathbf{Z}^T \mathbf{y}$ and hence

$$\mathbf{A} = \mathbf{y}^T \mathbf{Z} (\mathbf{Z} \mathbf{Z}^T)^{-1} \quad (5)$$

4.2 Deep Vector Autoregression

We propose the term Deep Vector Autoregression to refer to the broad class of Vector Autoregressive models that use deep learning to model the dependences between system variables through time. In particular, as before we let \mathbf{y}_t denote the $(K \times 1)$ vector that describes the state of system at time t . Consistent with the conventional VAR we assume that each individual time series y_{it} can be modelled as a function of lagged realizations of all variables y_{jt-p} , $j = 1, \dots, K$, $m = 1, \dots, p$. More specifically we have

$$y_{it} = f_i(\mathbf{y}_{t-1:t-p}; \theta) + v_{it} \quad , \quad \forall i = 1, \dots, K \quad (6)$$

where $\mathbf{y}_{t-1:t-p} = \{y_{jt-m}\}_{j=1, \dots, K}^{m=1, \dots, p}$ is the vector of lagged realizations, f_i is a variable specific mapping from past lags to the present and θ is a vector of parameters. While in the conventional VAR above we assumed that the multivariate process can be modelled as a system of linear stochastic equations, our proposed Deep VAR(p) can similarly be understood as a system of potentially highly non-linear equations. As we argued earlier, Deep Learning has been shown to be remarkably successful at learning mappings of arbitrary functional forms (Goodfellow, Bengio, and Courville 2016).

Note that the input and output dimensions in (6) are exactly the same as in the conventional VAR(p) model (equation (3)): f_i maps from $\mathbf{y}_{t-1:t-p} \in \mathbb{R}^{Kp \times 1}$ to a scalar. Our proposed plain-vanilla approach to Deep VARs diverges as little as possible from the conventional approach: it boils down to simply modelling each of the univariate outcomes in (6) as a deep neural network. We can restate this approach more compactly as

$$\mathbf{y}_t = \mathbf{f}(\mathbf{y}_{t-1:t-p}; \theta) + \mathbf{v}_t \quad (7)$$

where $\mathbf{f}(\cdot) = (f_1(\cdot), f_2(\cdot), \dots, f_K(\cdot))^T \in \mathbb{R}^{K \times 1}$ is just the stacked vector of mappings to univariate outcomes described in (6).

The notation in (7) gives rise to a more unified and general approach to Deep VARs that would treat the whole process as one single dynamical system to be modelled through one deep neural network \mathbf{g} :

$$\mathbf{y}_t = \mathbf{g}(\mathbf{y}_{t-1:t-p}; \theta) + \mathbf{v}_t \quad (8)$$

This approach is in fact proposed and investigated by Verstyuk (2020) in his upcoming publication. We decided to go with the approach in (7) for two reasons: firstly, the link to conventional VARs is made abundantly clear through this implementation and, secondly, we found that the equation-by-equation approach produces good modelling outcomes and is relatively easy to implement using state-of-the art software.

Finally, note that if f_i in (3) is assumed to be linear and additive for all $i = 1, \dots, K$ then we are back to the conventional VAR(p). This illustrates the point we made earlier that the linear VAR(p) is just a particular case of a Deep VAR(p). Since the model described in equations (6) and (7) is less restrictive but otherwise consistent with the conventional VAR framework, we expect that it outperforms the traditional approach towards modelling multivariate time series processes.

4.3 Deep Neural Networks - a whistle-stop tour

So far we have been speaking about deep learning in rather general terms. For example, above we have referred to our model of choice for learning the mapping $f_i : \mathbf{y}_{t-1:t-p} \mapsto y_{it}$ as a **deep neural network**. The class of deep neural networks can further be roughly divided into **feedforward neural networks** and **recurrent neural networks**. As the term suggests, the latter is generally used for sequential data and therefore our preferred model of choice. Nonetheless, below we will begin by briefly exploring feedforward neural networks first. This should serve as a good introduction to neural networks more generally and (even though we have not tested this empirically) there is good reason to believe that even Deep VARs using feedforward neural networks perform well.

4.3.1 Deep Feedforward Neural Networks

The term **deep feedforward neural network** or **multilayer perceptron** (MLP) is used to describe a broad class of models that are composed of possibly many functions that together make up the directed acyclical graph. The functions $f_i(\cdot)$ - sometimes referred as layers \mathbf{h}_i - are chained together hierarchically with the first layer feeding forward its outputs to the second layer and so on (Goodfellow, Bengio, and Courville 2016). Applied to our case, an MLP with H hidden layers can be loosely defined as follows:

$$f_i(\mathbf{y}_{t-1:t-p}; \theta) = f_i^{(H)} \left(f_i^{(H-1)} \left(\dots f_i^{(1)} (\mathbf{y}_{t-1:t-p}) \right) \right) \quad (9)$$

The depth of the MLP is defined by the number of hidden layers H , where, generally speaking, deeper networks are more complex.

The desired outputs of any $f_i^{(h)}$ that will serve as inputs for $f_i^{(h+1)}$ cannot be inferred from the training data $\mathbf{y}_{t-1:t-p}$ ex-ante, which is where the term **hidden** layer stems from. Each $f_i^{(h)}$ is typically valued on a vector of hidden units, each of them receiving a vector of inputs from $f_i^{(h-1)}$ and returning a scalar that is referred to as activation value. This approach is inspired by neuroscience, hence the term **neural** network (Goodfellow, Bengio, and Courville 2016).

4.3.2 Deep Recurrent Neural Networks

Recurrent neural networks (RNN) are based on the idea of persistent learning: a continuous process that evolves gradually and at each step uses information about its prior states instead of continuously reinventing itself and starting from scratch. To this end, RNNs develop the basic concepts underlying feedforward neural networks by incorporating feedback loops. Formally the loop is typically made explicit as follows

$$\mathbf{h}_t = f(\mathbf{h}_{t-1}, \mathbf{x}_t; \theta) \quad (10)$$

where $\mathbf{h}_t \in \mathbb{R}^{N \times 1}$ corresponds to the hidden state of the dynamical system at time t that the RNN learns (Goodfellow, Bengio, and Courville 2016), and N corresponds to the number of hidden units in each hidden layer, known as the width of the layer. In the given context we have that $\mathbf{x}_t = \mathbf{y}_{t-1:t-p}$ as specified in (7). Given some random initial hidden state vector \mathbf{h}_0 the RNN updates parameters sequentially at each time step t as follows

$$\begin{aligned} \mathbf{a}_t &= \mathbf{b} + \mathbf{W}\mathbf{h}_{t-1} + \mathbf{U}\mathbf{h}_{-1} \\ \mathbf{h}_t &= \tanh(\mathbf{a}_t) \\ \hat{\mathbf{y}}_t &= \mathbf{c} + \mathbf{V}\mathbf{h}_t \end{aligned} \quad (11)$$

where $\mathbf{b} \in \mathbb{R}^{N \times 1}$ and $\mathbf{c} \in \mathbb{R}^{K \times 1}$ are vectors of constants (biases), \tanh is the hyperbolic tangent activation function and \mathbf{W} , \mathbf{U} , \mathbf{V} are coefficient matrices, where $\mathbf{W}, \mathbf{U} \in \mathbb{R}^{N \times N}$ and $\mathbf{V} \in \mathbb{R}^{K \times N}$. Note that to simplify the notation we have omitted the layer index in (11): to be specific, \mathbf{h}_t really represents $\mathbf{h}_t^{(H)}$ (the ultimate hidden layer), \mathbf{h}_{-1} stands for $\mathbf{h}_t^{(H-1)}$ (the penultimate layer). Finally, at each step t the first layer $\mathbf{h}_t^{(0)}$ of the forward propagation corresponds to $\mathbf{y}_{t-1:t-p}$.

A shortfall of generic recurrent neural networks is that they fail to capture long-term dependencies. More specifically, if parameters are propagated over too many stages in a simple RNN it typically suffers from the problem of **vanishing gradients** (Goodfellow, Bengio, and Courville 2016). Fortunately, there exist effective extensions of the RNN, most notably the long short-term memory (LSTM), which was introduced by Hochreiter and Schmidhuber (1997) and is our model of choice for Deep VARs. The key idea underlying LSTMs is to regulate exactly how much information is propagated from one cell state vector \mathbf{C}_{t-1} to the next \mathbf{C}_t through the introduction of so called sigmoid gates:

“The LSTM [has] the ability to remove or add information to the cell state, carefully regulated by structures called gates. Gates are a way to optionally let information through.” — Olah (2015)

These regulating gate layers include a **forget gate** \mathbf{f}_t , an **input gate** \mathbf{i}_t and a **output gate** \mathbf{o}_t . Each of them are vector-values sigmoid functions whose elements $\mathbf{f}_{it}, \mathbf{i}_{it}, \mathbf{o}_{it}$ are bound between 0 and 1. Their individual purposes are implied by their names: faced with \mathbf{h}_{t-1} and $\mathbf{y}_{t-1:t-p}$, the forget gate regulates how much of each individual unit in \mathbf{C}_{t-1} is retained. Then the input gate regulates which units of \mathbf{C}_{t-1} should be updated and to what candidate values $\tilde{\mathbf{C}}_{t-1}$. Using the previous two steps the actual update is performed according to the following rule

$$\mathbf{C}_t = \mathbf{f}_t \odot \mathbf{C}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{C}}_{t-1} \quad (12)$$

where \odot indicates the element-wise product. Finally, the output gate acts like a filter on \mathbf{C}_t : the new hidden state is computed as $\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{C}_t)$ where as before we use the hyperbolic tangent as our activation function.¹ Formally, we can summarize the LSTM neural network underlying our Deep VAR framework as follows:

$$\begin{aligned} \mathbf{f}_t &= \sigma(\mathbf{b}_f + \mathbf{W}_f \mathbf{h}_{t-1} + \mathbf{U}_f \mathbf{h}_{-1}) \\ \mathbf{i}_t &= \sigma(\mathbf{b}_i + \mathbf{W}_i \mathbf{h}_{t-1} + \mathbf{U}_i \mathbf{h}_{-1}) \\ \mathbf{o}_t &= \sigma(\mathbf{b}_o + \mathbf{W}_o \mathbf{h}_{t-1} + \mathbf{U}_o \mathbf{h}_{-1}) \\ \mathbf{C}_t &= \mathbf{f}_t \odot \mathbf{C}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{b}_C + \mathbf{W}_C \mathbf{h}_{t-1} + \mathbf{U}_C \mathbf{h}_{-1}) \\ \mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{C}_t) \\ \hat{\mathbf{y}}_t &= \mathbf{c} + \mathbf{V} \mathbf{h}_t \end{aligned} \quad (13)$$

which is best understood when read from top to bottom. Once again we have simplified the notation by omitting the layer index in (11). The same notation as before applies.

4.4 Model selection

There are at least two important modelling choices to be made in the context of conventional VARs. The first choice concerns properties of the time series data itself, in particular the order of integration and cointegration. The second choice is about the lag order p . In order to arrive at appropriate decisions regarding these choices the VAR literature provides a set of guiding principles. We propose to apply these same principles to the Deep VAR, firstly because they are intuitive and simple and secondly because treating both models equally to begin with allows for a better comparison of the two models at the subsequent modelling stages.

4.4.1 Stationarity

When working with time series we are generally concerned about stationarity. Broadly speaking stationarity ensures that the future is like the past and hence any predictions we make based on past data adequately describe future outcomes. In order to state stationarity conditions in the VAR context it is convenient to restate the K -dimensional VAR(p) process in companion form as

¹For a clear and detailed exposition see Olah (2015).

$$\mathbf{Y}_t = \begin{pmatrix} \mathbf{c} \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \mathbf{A}\mathbf{Y}_{t-1} + \begin{pmatrix} \mathbf{u}_t \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad (14)$$

where $\mathbf{Y}_t = (\mathbf{y}_t^T, \dots, \mathbf{y}_{t-p+1}^T)^T \in \mathbb{R}^{Kp \times 1}$ and $\mathbf{A} \in \mathbb{R}^{Kp \times Kp}$ is referred to as the companion matrix (Kilian and Lütkepohl 2017). Stationarity of the VAR(p) follows from stability: a VAR(p) is stable if the effects of shocks to the system eventually die out. Stability can be assessed through the system's autoregressive roots or equivalently by looking at the eigenvalues of the companion matrix \mathbf{A} (Kilian and Lütkepohl 2017). In particular, for the VAR(p) in (14) to be stable we condition that the Kp eigenvalues λ that satisfy

$$\det(\mathbf{A} - \lambda \mathbf{I}_{Kp}) = 0$$

are all of absolute value less than one. Stability implies that the first and second moments of the VAR(p) process are time-invariant, hence ensuring weak stationarity (Kilian and Lütkepohl 2017).

A straight-forward way to deal with stationarity of VARs is to simply ensure that the individual time series entering the system are stationary. This usually involves differencing the time series until they are stationary: for any time series y_i that is integrated of order $I(\delta)$, there exists a δ -order difference that is stationary. An immediate drawback of this approach is the loss of information contained in the levels of the time series. Modelling approaches that take into account cointegration of individual time series can ensure system stationarity and still let individually non-stationary time series enter the system in levels (Hamilton 1994).

4.4.2 Lag order

The VARs lag order p can to some extent be thought of as the persistency of the process: past innovations that still affect outcomes in time t happened at most p periods ago. From a pure model selection perspective we can also think of additional lags in terms of additional regressors that add to the model's complexity. From that perspective, choosing a lower lag order corresponds to a form of regularization as it pertains to a more parsimonious model.

Various strategies have been proposed to estimate the true or optimal lag order p empirically (Kilian and Lütkepohl 2017). Among the most common ones are sequential testing procedures and selection based on information criteria. The former involves sequentially adding or removing lags - **bottom-up** and **top-down** testing, respectively - and then testing model outcomes in each iteration. A common point of criticism of sequential procedures is that the order tests matters (Lütkepohl (2005)).

Here we will focus on selection based on information criteria, which to some extent makes the trade-off between bias and variance explicit (Kilian and Lütkepohl 2017). In particular, it generally involves minimizing information criteria of the following form

$$C(m) = \log(\det(\hat{\Sigma}(m))) + \ell(m) \quad (15)$$

where $\hat{\Sigma}$ is just the sample estimate of the covariance matrix or errors and ℓ is a loss function that penalizes high lag orders. In particular, we have that our best estimate of the optimal lag order p is simply

$$\hat{p} = \arg \min_{m \in \mathcal{P}} C(m) \quad (16)$$

where $\mathcal{P} = [m_{\min}, m_{\max}]$. We will consider all of the most common functional choices for (15).

4.4.3 Neural Network Architecture

As mentioned above, a NN is therefore characterized for having a very large number of parameters and therefore making it relatively easy to overfit the data. This is a concern specially when the NN is trained in a relatively small dataset. In time series the amount of data to train the model is quite limited, therefore having so many parameters may result in the NN learning the noise of the function, potentially leading to overfitting which in turn results in fairly poor performance results of the model in the test set. Therefore some regularization must be added into the model in order to avoid overfitting.

As proposed in Srivastava et al. (2014), one way to reduce overfitting can be fitting a lot of NN in the data and the averaging its predictions. This is not feasible due to computational problems. What one can do instead is to randomly drop some units in each layer. This is also known as dropout, this technique simulates the training of multiple NN with different architectures in the same data set. By using dropout, we add some noise into the model. This avoids a certain layer trying to adapt to a mistake made by a previous one, hence potentially leading to complex adaptations which results in overfitting. Dropout therefore can be used as a regularization technique that makes the model more robust.

The way we implement dropout in the Deep VAR model is by specifying the probability to which a node is removed from the layer. This means that the information that arrives to that unit is forgotten.

As another effort to avoid overfitting, we use the same architecture for each layer of the NN. That is, we do not allow each layer to have different number of units, nor different activation functions nor different dropout rates, which could potentially result in the model learning the noise. In turn the Deep VAR model is characterized for having equally sized layers of size N.

Figure 1 shows the neural network architecture for the case of two lags ($p = 2$) and four variables ($K = 4$). We can see that the first layer corresponds to the inputs, that is, the input layer $\in \mathbb{R}^{Kp \times 1}$. This architecture consists of two hidden layers each made of fourteen units. Given that we are aiming at predicting four variables the output layer consists of four nodes. The edges illustrate the flow of information of the network, where each edge has an associated weight.

Finally, with respect to how the LSTMs underlying the Deep VAR are compiled, the Adam optimization algorithm Kingma and Ba (2014) is used. This algorithm can be used instead of the more traditional stochastic gradient descent to update network weights. There are several reasons to use this algorithm that are particularly appealing, among them we have its straightforward implementation, that it is computationally efficient and that the hyper-parametrization has an intuitive interpretation and typically requires little tuning. Adam is different to classic stochastic

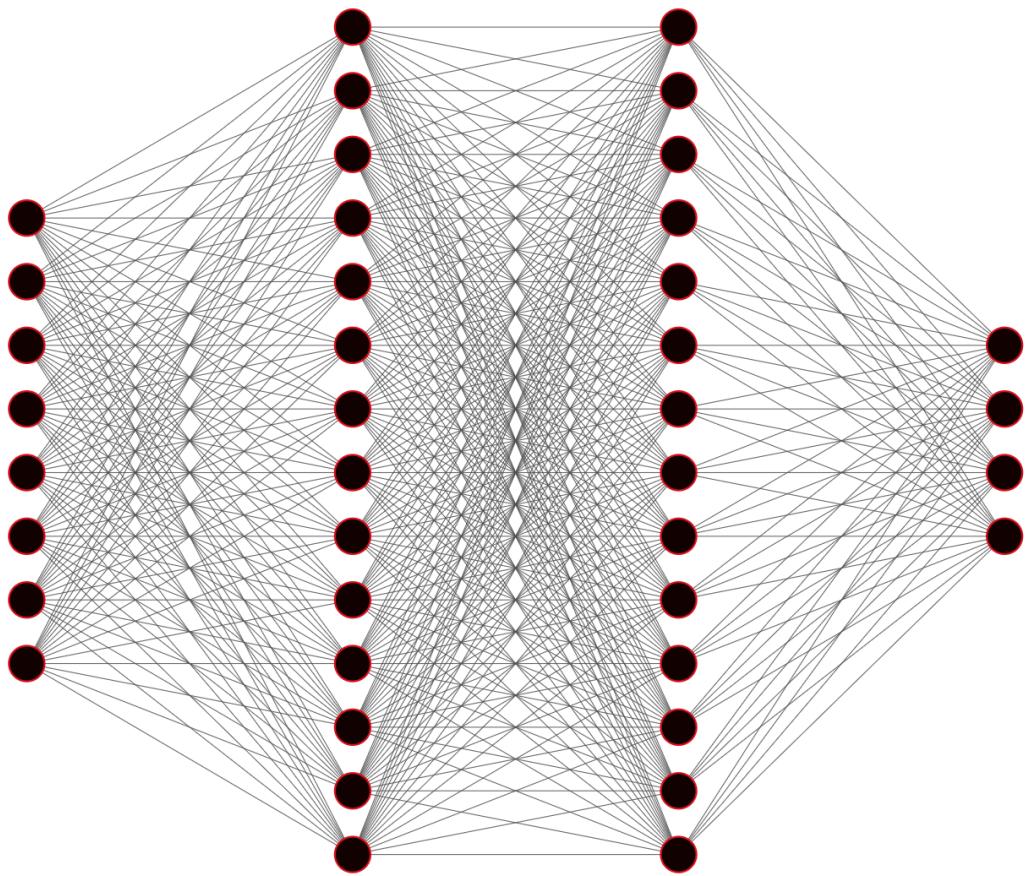


Figure 1: Neural Network Architecture.

gradient descent given that in the latter learning rates and always the same for all weight updates and hence does not change during the training.

5 Empirical results

We now proceed to benchmark the proposed Deep VAR model against the conventional VAR using out macroeconomic time series data. To begin with, we compare both models in terms of their in-sample fit. For this part of the analysis the models will be strictly run under the same framing conditions. Due to the RNN’s capacity to essentially model any possible function $f_i(\cdot)$ the Deep VAR dominates the VAR in this realm. We investigate during what time periods the outperformance of the Deep VAR is particurly striking to gain a better understanding of when and why it pays off to relax the linearity constraint.

These findings with respect to in-sample performance provide some initial evidence in favor of the Deep VAR. But since a reduction in modelling bias is typically associated with an increase in variance, we are particularly interested in benchmarking the models with respect to their out-of-sample performance. To this end we split our sample into train and test subsamples. We then firstly benchmark the models in terms of their pseudo out-of-sample fit. Finally we also look at model performance with respect to n -step ahead pseudo out-of-sample forecasts.

The final part of this section relaxes the constraint on the framing conditions. In particular, we investigate how hyperparameter tuning with respect to the neural network architecture and lag length p can improve the performance of the Deep VAR.

5.1 In-sample fit

For this first empirical exercise both models are trained on the full sample. We have decided to include the post-Covid sample period despite the associated structural break, since it serves as interesting point of comparison. The optimal lag order as determined by the Akaike Information Criterium is $p = 6$, where we used a maximum possible lag of $p_{\max} = 12$ corresponding to one year. A look at the eigenvalues of the companion matrix showed that the VAR(6) is stable. The LSTMs underlying the Deep VAR model are composed of $H = 2$ that count $N = 100$ hidden units each. The dropout rate is set to $p = 0.5$.

To assess the fit of our models we use the root mean squared error (RMSE) as our preferred loss function. Figure 2 shows the cumulative RMSE of both the VAR model and Deep VAR model for each of the time series over the whole sample period. The first thing we can observe is that the RMSE of the Deep VAR is consistently flatter than the RMSE of the VAR. With respect to in-sample performance, the Deep VAR the VAR throughout the entire time period of the experimental analysis and for all of the considered variable. This empirical observation seems to confirm our expectation that the vector autoregressive process is characterized by important non-linear dependencies across time and variables that the conventional VAR fails to capture.

Figure 2 is especially useful to asses in which specific periods the Deep VAR model achieves better modelling outcomes than the VAR model. From the very beginning and across variables, we observe that the increase in cumulative loss for the VAR model is greater than for the Deep VAR model. The US economy during 1960s was influence by John F. Kennedy’s introduction of **New Economics**,

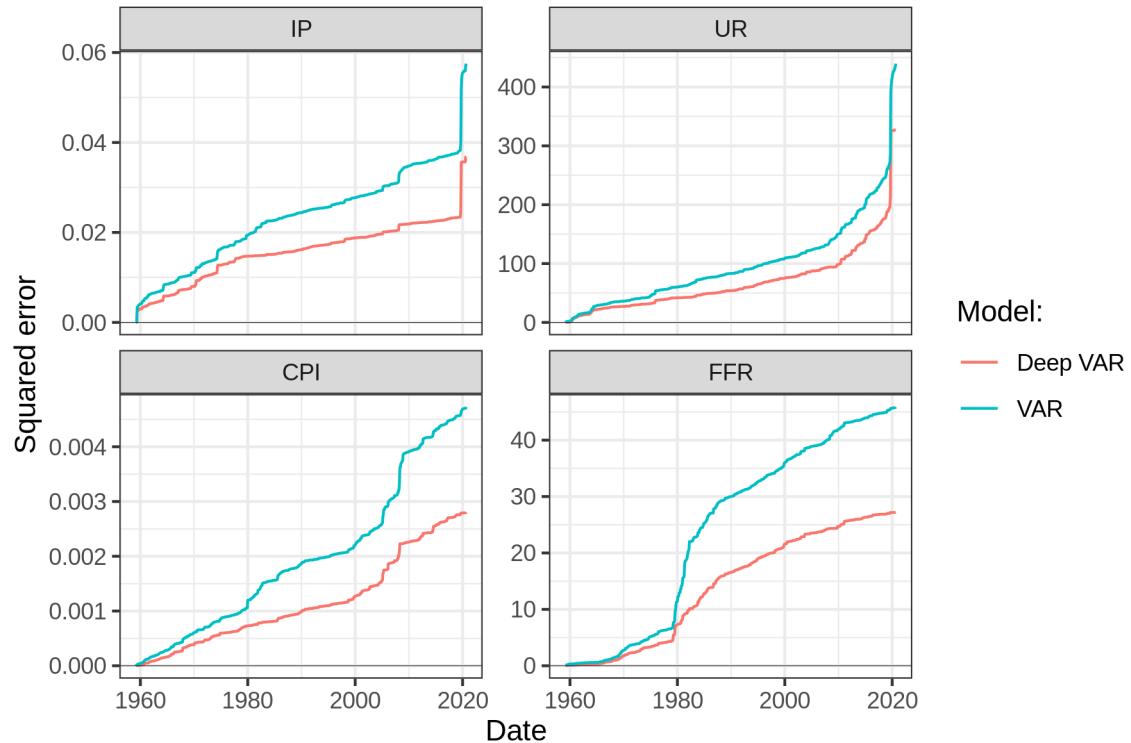


Figure 2: Comparison of cumulative loss over the entire sample period for conventional VAR and proposed Deep VAR.

which was informed by Keynesian ideas and characterized by increasing levels of inflation, a reduction in unemployment and output growth. The change in government certainly corresponded to a regime switch with respect to the economy (Perry and Tobin 2010) and in that sense it is interesting to observe that the Deep VAR appears to be doing a better job at capturing the underlying changes. The 1970s can be broadly thought of as a continuation of New Economy and loosely defined as a period of stagflation. The Deep VAR continues to outperform the VAR during that period.

The first truly interesting development we can observe in Figure 2 coincides with the onset of the Volcker disinflation period. Following years of sustained CPI growth, Paul Volcker set the Federal Reserve on course for a series of interest rate hikes as soon as he became chairperson of the central bank in August 1979. The shift in monetary policy triggered fundamental changes to the US economy and in particular the key economic indicators we are analysing here throughout the 1980s (Goodfriend and King 2005). Despite this structural break, the increase in the cumulative RMSE of the Deep VAR remains almost constant during this decade for most variables. The performance of the VAR on the other hand is unsurprisingly poor over the same period, in particularly so for the CPI and the Fed Funds Rate, which arguably were the two variables most directly affected by the change in policy. The Deep VAR also clearly dominates the VAR with respect to the output related variables (IP) and to a lesser extent unempoyment. These findings indicate that changes to the monetary transmission mechanism in response to sudden policy shifts are not well captured by a linear-additive vector autoregressive model. Instead they appear to unfold in a high-dimensional latent state space, which the Deep VAR by its very construction is designed to learn.

Following the Volcker disinflation period, Figure 2 does not reveal any clear outperformance of either of the models during the 1990s. Interestingly the dot-com bubble has little affect on either of the models, aside from a small pick-up in cumulative loss with respect to the CPI for both models. With all that noted, the Deep VAR still continuously outperforms the VAR since evidently its cumulative loss increases at a lower pace alltogther.

As the Global Financial Crisis unfolds around 2007 the pattern we observed for the Volcker disinflation reemerges, albeit to a lesser extent: there is a marked jump in the difference between the cumulative loss of the VAR and the Deep VAR, in particular so for the CPI, the Fed Funds rate and industrial production. The gap for all these variables continues to widen during the aftermath of the crisis. The Deep VAR once again does a better job at modelling the changes that the dynamical system undergoes: post-crisis US monetary policy was characterized by very low interest rates, low levels of inflation as well as the introduction of a range of non-conventional monetary policy tools including quantitative easing and forward guidance.

Finally, it is also interesting to observe how both models perform in response to the unprecedented exogenous shock that Covid-19 constitutes. Both models incur huge errors with respect to both IP and UR - the two series most significantly affected by Covid. Evidently though, the magnidtude of the errors is somewhat larger for the VAR than for the Deep VAR. This, once again seems to confirm our hypothesis that the Deep VAR model captures important non-linear dependences across time and variables that the conventional VAR fails to catpure.

As a santiy check we also visually inspected the distributional properties of the model residuals for the full-sample fit. The outcomes are broadly consistent across models: while for some variables residuals are clearly not Gaussian, we see no evidence of serial autocorrelation of residuals (see Figures 5 and 6 in the appendix).

Table 1: Root mean squared error (RMSE) for the two models across subsamples and variables.

Sample	Variable	DVAR	VAR	Ratio (DVAR / VAR)
test	IP	0.00608	0.01484	0.40957
test	UR	0.97022	1.65170	0.58741
test	CPI	0.00273	0.00342	0.79869
test	FFR	0.19964	0.23974	0.83271
train	IP	0.00528	0.00727	0.72610
train	UR	0.32325	0.43322	0.74615
train	CPI	0.00149	0.00232	0.64019
train	FFR	0.16115	0.25780	0.62509

5.2 Out-of-sample fit

In order to assess if the Deep VAR’s outperformance is a consequence of overfitting, we now repeat the previous exercise, but this time we train the models on a subsample of our date. The training sample spans from March, 1959 to October, 2008, whereas the test data goes from November, 2008 to March, 2021. This corresponds to training the model on 80 percent of the data and retaining the remaining 20 percent for testing purposes. The optimal lag order for the training subsample is $p = 7$ where we use the same criterion and maximum lag order as before. Once again we find this VAR specification to be stable.

Tables 1 shows the Root Mean Squared Error (RMSE) for the in-sample and the out-of-sample predictions of both the VAR model and the Deep VAR model. We can see that the RMSE for the Deep VAR outperforms the one for the conventional VAR for both the training data and the test data and for all time series. The fifth column of the table shows us the ratio between the RMSEs of the Deep VAR and the VAR: the lower the ratio, the better the Deep VAR compared to the VAR. With respect to the training sample, the RMSE of the Deep VAR model is consistently less than 75% of that of the conventional VAR reflecting to some extent the results of the previous sections. Turning to the test data, there is no evidence that the Deep VAR is more prone to overfitting than the VAR. For both industrial production and unemployment, the Deep VAR yields an RMSE that is around half the size of that produced by the VAR. For inflation and interest rate predictions the outperformance on the test data is less striking, but still fairly significant.

5.3 Forecasts

Up until now we have been assessing the 1-step ahead predictions of both models. In our context these predictions can be thought of as 1-month ahead nowcasts from a practical perspective. Since real-time nowcasts have grown in popularity during recent years, the results so far should be of great interest to central bankers and other practitioners. Nonetheless, there is typically also great interest in time series forecasts at longer horizons. We therefore briefly introduce n -step ahead pseudo out-of-sample forecasts in this section and revisit them again further below.

Forecasts are produced recursively both for the VAR and the Deep VAR. Specifically, we use the

Table 2: Comparison of n-step ahead pseudo out-of-sample forecasts.

Variable	VAR FRMSE	Deep-VAR FRMSE	VAR correlations	Deep-VAR correlations
IP	0.01870	0.01602	-0.30409	-0.65279
UR	0.85984	0.82785	-0.10093	0.27425
CPI	0.00946	0.00708	-0.33567	0.07823
FFR	0.52321	0.39161	-0.55935	0.01161

models we trained on the training data to recursively predict one time period ahead, concatenate the predictions to the training data and repeat the process.² This way we produce one-year ahead forecasts beginning from the first date in the test sample (October, 2008).

Table 2 shows the resulting root mean squared forecast errors (RMSFE) along with correlation between forecasts and realizations. As we can see in the table, the RMSFE of the Deep VAR is consistently lower than the one for the VAR. Regarding correlations the VAR produces forecasts that are negatively correlated with actual outcomes for all time series: in other words, when the time series evolves in one direction, the VAR forecast tends to evolve in the opposite direction. For industrial production, the Deep VAR forecast also has a highly negative correlation with the actual values. For the rest of time series the Deep VAR forecasts correlate positively with actual outcome, albeit weakly. Another general observation we made with respect to these forecasts is that the forecasts from the conventional VAR are fairly volatile, while the Deep VAR forecasts swiftly revert to steady levels (see Figures 11 and 12 in the appendix).

5.4 Varying hyperparameters

While up until now with respect to model selection we have intentionally remained strictly within the conventional VAR framework, we will now relax that constraint and vary the lag length as well as hyperparameters of the Deep VAR. In particular, we perform a grid search where we vary the number of hidden layers (1,2,5), number of hidden units per layer (50,100,150), the dropout rate (0.3,0.5,0.7) and the lag order (10, 50, 100). For each combination of parameter choices we train the two models and compute the various performance measures introduced above.³ Our expectation is that the conventional VAR is prone to overfitting and will produce poor out-of-sample outcomes for higher lag orders. For the Deep VAR we expect to interesting variation in the outcomes for different lag order and hyperparameter choices. It is not clear ex-ante that the Deep VAR should suffer from the same issue of overfitting for higher lag orders. The bulk of the corresponding visualizations can be found in the appendix.

5.4.1 Tuning the Deep VAR

To begin with, we shall forget about benchmarking for a moment and focus on the outcomes for the Deep VAR as we vary parameters. Recall that a higher number of hidden layers (depth), a

²Note that for the Deep VAR an alternative approach would be to work with a different output dimension for the underlying neural networks.

³Of course, with respect to the conventional VAR only the lag order affects outcomes.

higher number of hidden units (width) and a smaller choice for the dropout rate all correspond to an increase in neural network complexity. Consistent with this intuition we find that the in-sample loss for the Deep VAR improve as complexity increases (Figure 13): higher complexity leads to a reduction in bias and as we noted earlier the underlying recurrent neural networks should in principle be able to model arbitrary functions (Goodfellow, Bengio, and Courville 2016). Conversely, we observe exactly the opposite pattern for out-of-sample loss: as evident from Figure 14 a higher choice for the dropout rate and lower choices for the depth and width of the neural networks generally yields a smaller out-of-sample RMSE across variables.

Interestingly, both in- and out-of-sample loss tend to decrease significantly as the number of lags increases. In other words, the Deep VAR seems to be relatively insensitive to overfitting with respect to the lag order. With that in mind, we find that using standard lag order selection tools such as the AIC above may in fact not be appropriate for Deep VARs.

Finally, Figure 15 provides an overview of how pseudo out-of-sample forecasting errors behave as we vary the hyperparameters. As before we produce one-year ahead forecasts starting from the end of the 80% training sample. In this context, the pattern is less clear and varies across variables. As the lag order increases, for example, the forecast performance for the unemployment rate deteriorates. For inflation, forecasts are poor for the medium lag choice of $p = 50$ and much better for the low and high lag orders. The exact opposite relationship appears to hold for the Fed Funds Rate. With respect to the choices for the Deep VAR hyperparameters it is difficult to establish any clear pattern at all. The magnitude of differences in RMSFE is generally very small, so overall we conclude that to some extent the variation we do observe may be random.

In light of this evidence, we propose that for the purpose of hyperparameter tuning Deep VARs researchers should focus on the RMSE associated with the 1-step ahead fitted values. For the underlying data, a reasonable set of hyperparameter choices could be: 1 hidden layer, 50 hidden units and a dropout rate of 0.5.

5.4.2 Benchmark

Using the hyperparameter choices proposed above we now turn back to comparing the performance of the Deep VAR to the conventional VAR. Figure 3 shows the pseudo out-of-sample RMSE and RMSFE for both models across the different lag choices. For the sake of completeness we also include the performance measures we obtained when we initially ran both models in section 5.2 using the optimal lag order as determined by the AIC.

The first observation is that the Deep VAR outperforms the VAR across the board, reflecting our earlier findings. As expected, the VAR is subject to overfitting for when high lag order are chosen. This trend is observed both for the RMSE as well as the RMSFE. The fact that n -step ahead forecasts of the VAR are also subject to overfitting with respect to the lag order, while the Deep VAR appears unaffected, to some extent may reflect what we observed earlier: for the given data, Deep VAR forecasts swiftly converge to steady levels, while VAR forecasts are volatile, which may explain the relative outperformance of the Deep VAR. It appears that this effect is amplified for higher lag orders.

To conclude this empirical section we summarize our main findings:

1. We provide evidence that the conventional, linear VAR fails to capture important non-

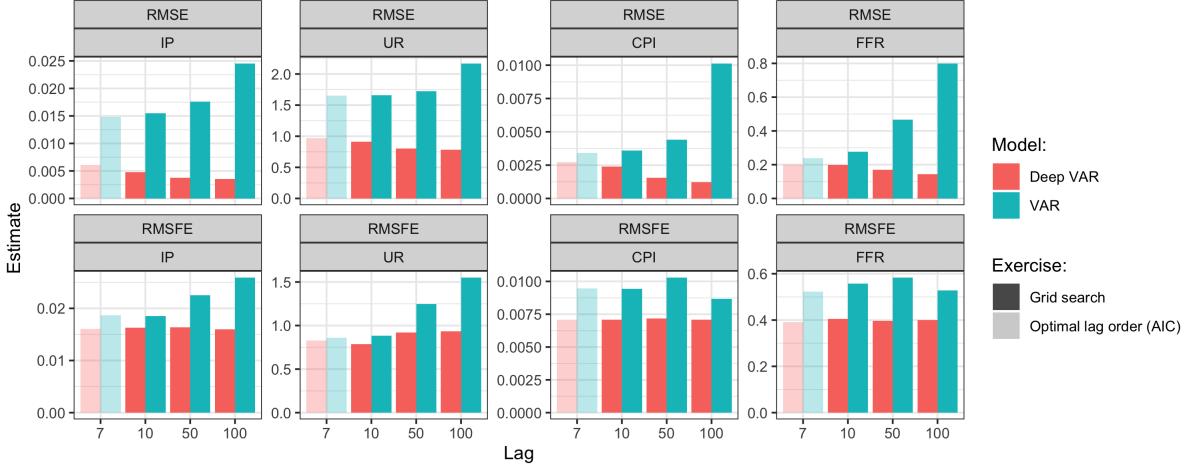


Figure 3: Pseudo out-of-sample RMSE and RMSFE for both models across the different lag choices. For the sake of completeness we also include the performance measures we obtained when we initially ran both models using the optimal lag order as determined by the AIC.

linear dependencies across time and variables that are typically used to model the monetary transmission mechanism.

2. Tapping into the broader class of Deep VARs leads to consistently better modelling outcomes.
3. Deep VARs appear to be relatively insensitive to very high lag orders at which conventional VARs are prone to overfitting.

6 Caveats and extensions

In this section we will talk about the main short-falls of our work, and potential ways to improve it.

In this paper we have developed a method to produce point forecasts which has shown to overperform the traditional VAR. However, policy makers do not only take decisions based on point estimates, but they also take into account how uncertain they are. That is why, when constructing forecasts you also want to infer confidence intervals for your point estimates. Doing that with the VAR methodology is not difficult, given that standard errors are inherent to the model, and easily attainable. This is not the case for the Deep VAR, but this does not mean that we cannot obtain them but that we need to rely on bootstrapping or montecarlo techniques to infer these bounds. This relates to the dropout rate used when fitting the Deep VAR in the test set. Given that the dropout rate specifies the number of observations that the model randomly removes at each layer, each time the model is refitted the test set will result in different but similar predictions. Threby, this can be used to get the standard errors needed for reporting the confidence intervals. This approach of refitting the model in the test set can be used for getting confidence intervals for the predictions of the test set.

Impulse response functions are another missing milestone in our paper and to which future research should be dedicated. IRFs are important to see how your model captures the dynamics of the variables in response to a shock in a given period of time. When estimating the model with the traditional VAR, computing IRFs is quite straightforward. Provided that all the roots of the autoregressive polynomial lie outside the unit circle, transforming the model into its moving average

representation and plugging the shock into the vector moving average (VMA) representation results into the IRF of the shock analysed. However, generating the IRFs is more difficult in the Deep VAR setting given that we cannot get the Deep VMA representation as a consequence of its underlying structure. Thereby, future work would be needed in this direction to be able to derive the IRFs from the Deep VAR model.

Within the VAR framework, it is quite usual to obtain the Forecast Error Variance Decomposition (FEVD) as well. This allows the researcher or the policy maker to exactly know what percentage of the variability of the forecast error of one variable is explained by the other variables in the system. However, is not clear how to get that from the Deep VAR perspective. Our intuition is that, in the same way you can use montecarlo simulations to get different estimations through the dropout rate and therefore get some variability to infer the standard error, you could also construct different forecasts. These forecasts would lead to different forecast errors and therefore potentially ending up reproducing the FEVD for the Deep VAR.

Apart from the forecasts and the previously mentioned important insights that can be derived from the VAR model and that could potentially be derived from the Deep VAR model with further work, policy makers are also concerned about the interpretability of the model. The linear relationship of inputs and outputs of the VAR allows the researcher or policy maker to assess the effect of one variable of the system to another and therefore assessing if that variable granger causes the other. In the case of the Deep VAR model, its non linear structure makes it impossible to recover this associations. There has been some research already devoted to this area. For instance the Neural Additive Vector Autorregression (NAVAR) aims to combine neural networks in the first stage of the model use a linear additive structure in the end in order to assess if one variable granger causes another. It uses a NN for each variable in the system to output the degree of causality of this variable to all the variables in the system in the form of contributions. Then, once the model has assessed the contribution of all the variables to themselves and the others, an additive structure is implemented to assess the contributions of the system to each of the variables.

7 Conclusions

The initial motivation of this study was to see whether by implementing some of the latest developments in the machine learning field, and in particular, in the deep learning domain, we could yield potential improvements in the forecasting performance of the traditional econometric techniques used in the literature. In order to do that, we introduce a new framework based on the neural network architecture while maintaining the VAR structure, which we name Deep VAR.

Towards this purpose, we compare the fit of the Deep VAR vs. the traditional VAR for monthly US economic data in the period 1959-2021. The results show a persistent and large difference in the performance of the in-sample fit. In particular, the Deep VAR achieves a much lower cumulative loss than the VAR over the entire period for the four analysed time series. This difference aggravates even more in recession periods, that is, when the time series exhibits non-linear relationships. Hence, reinforcing our initial hypothesis that by modelling the series using neural networks it is possible to capture more complex patterns.

When it comes to the out-of-sample fit, a priori it may seem that the Deep VAR is expected to overfit the data due to its high number of parameters. Although, the results clearly show that the Deep VAR achieves a much lower out-of-sample RMSE than the VAR. We reasoned that this is owed

to the regularization applied to the model, the dropout rate and the constraints regarding the neural network architecture. This method guarantees that the Deep VAR does not learn the error of the data in the training stage, and therefore leading to a robust fit for the out-of-sample performance.

Policy makers, however, are mainly interested in the forecasting performance of a given model rather than its fit. In this realm, the Deep VAR also performs better than the traditional VAR. In particular, it results in a lower forecasting root mean squared error for each of the analyzed series. To sum up, we can conclude that the Deep VAR outperforms the VAR in every domain explored; the in-sample and out-of-sample fit and the forecasting one.

The Deep VAR framework has not been introduced in the time series literature, and as any brand new methodology, it leaves a lot of unanswered questions, hence future research on Deep VAR is still missing. In particular, future research should be addressed to develop a methodology to recover standard errors for the point estimates, and therefore, coming up with confidence intervals. Furthermore, exploring a method to be able to create impulse response functions would make of the Deep VAR framework a method as complete as the VAR one, but which has proved to outperform in all possible aspects.

References

- Bernanke, Ben. 1990. "The Federal Funds Rate and the Channels of Monetary Transmission." National Bureau of Economic Research.
- Brock, William A, David Arthur Hsieh, Blake Dean LeBaron, William E Brock, and others. 1991. *Nonlinear Dynamics, Chaos, and Instability: Statistical Theory and Economic Evidence*. MIT press.
- Cortes, Corinna, and Vladimir Vapnik. 1995. "Support-Vector Networks." *Machine Learning* 20 (3): 273–97.
- Dorffner, Georg. 1996. "Neural Networks for Time Series Processing." In *Neural Network World*. Citeseer.
- Fix, E, and J Hodges. 1951. "An Important Contribution to Nonparametric Discriminant Analysis and Density Estimation." *International Statistical Review* 3 (57): 233–38.
- Friedman, Milton, and Anna Jacobson Schwartz. 2008. *A Monetary History of the United States, 1867-1960*. Princeton University Press.
- Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.
- Goodfriend, Marvin, and Robert G King. 2005. "The Incredible Volcker Disinflation." *Journal of Monetary Economics* 52 (5): 981–1015.
- Greene, William H. 2012. "Econometric Analysis 7th Ed (International)." Pearson.
- Hamilton, James Douglas. 1994. *Time Series Analysis*. Princeton university press.
- Hamzaçebi, Coşkun. 2008. "Improving Artificial Neural Networks' Performance in Seasonal Time Series Forecasting." *Information Sciences* 178 (23): 4550–59.
- Ho, Tin Kam. 1995. "Random Decision Forests." In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, 1:278–82. IEEE.
- Hochreiter, Sepp, and Jürgen Schmidhuber. 1997. "Long Short-Term Memory." *Neural Computation* 9 (8): 1735–80.
- Joseph, Andreas, Eleni Kalamara, George Kapetanios, and Galina Potjagailo. 2021. "Forecasting Uk Inflation Bottom Up."
- Kihoro, J, RO Otieno, and C Wafula. 2004. "Seasonal Time Series Forecasting: A Comparative Study of ARIMA and ANN Models."
- Kilian, Lutz, and Helmut Lütkepohl. 2017. *Structural Vector Autoregressive Analysis*. Cambridge University Press.
- Kingma, Diederik P, and Jimmy Ba. 2014. "Adam: A Method for Stochastic Optimization." *arXiv Preprint arXiv:1412.6980*.
- Kydland, Finn E, and Edward C Prescott. 1982. "Time to Build and Aggregate Fluctuations." *Econometrica: Journal of the Econometric Society*, 1345–70.
- Lucas Jr, Robert E. 1976. "Econometric Policy Evaluation: A Critique." In *Carnegie-Rochester Conference Series on Public Policy*, 1:19–46. North-Holland.

- Lütkepohl, Helmut. 2005. *New Introduction to Multiple Time Series Analysis*. Springer Science & Business Media.
- McCulloch, Warren S, and Walter Pitts. 1990. “A Logical Calculus of the Ideas Immanent in Nervous Activity.” *Bulletin of Mathematical Biology* 52 (1-2): 99–115.
- Olah, Chris. 2015. “Understanding LSTM Networks.” <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>.
- Perry, George L, and James Tobin. 2010. *Economic Events, Ideas, and Policies: The 1960s and After*. Brookings Institution Press.
- Romer, Christina D, and David H Romer. 1989. “Does Monetary Policy Matter? A New Test in the Spirit of Friedman and Schwartz.” *NBER Macroeconomics Annual* 4: 121–70.
- Sims, Christopher A, and others. 1986. “Are Forecasting Models Usable for Policy Analysis?” *Quarterly Review* 10 (Win): 2–16.
- Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting.” *The Journal of Machine Learning Research* 15 (1): 1929–58.
- Verstyuk, Sergiy. 2020. “Modeling Multivariate Time Series in Economics: From Auto-Regressions to Recurrent Neural Networks.” Available at SSRN 3589337.
- Zhang, G Peter. 2003. “Time Series Forecasting Using a Hybrid ARIMA and Neural Network Model.” *Neurocomputing* 50: 159–75.
- Zhang, Guoqiang, B Eddy Patuwo, and Michael Y Hu. 1998. “Forecasting with Artificial Neural Networks:: The State of the Art.” *International Journal of Forecasting* 14 (1): 35–62.

A Tables

	IP	UR	CPI	FFR
min	-0.14	-10.60	-0.01	-1.76
max	0.06	5.10	0.01	1.61
mean	0.00	0.02	0.00	-0.00
sd	0.01	0.81	0.00	0.27
var	0.00	0.66	0.00	0.08
median	0.00	0.00	-0.00	0.00
IQR	0.01	0.70	0.00	0.28

Table 3: Summary statistics for the full sample.

	IP	UR	CPI	FFR
min	-0.04	-2.70	-0.01	-1.76
max	0.06	2.50	0.01	1.61
mean	0.00	0.01	0.00	-0.00
sd	0.01	0.63	0.00	0.27
var	0.00	0.40	0.00	0.08
median	0.00	0.00	-0.00	0.00
IQR	0.01	0.63	0.00	0.28

Table 4: Summary statistics pre Covid.

	IP	UR	CPI	FFR
min	-0.14	-10.60	-0.01	-0.63
max	0.06	5.10	0.01	0.35
mean	-0.00	0.68	0.00	0.01
sd	0.05	4.04	0.00	0.23
var	0.00	16.31	0.00	0.05
median	0.01	1.60	0.00	0.06
IQR	0.01	2.20	0.00	0.10

Table 5: Summary statistics post Covid.

B Figures

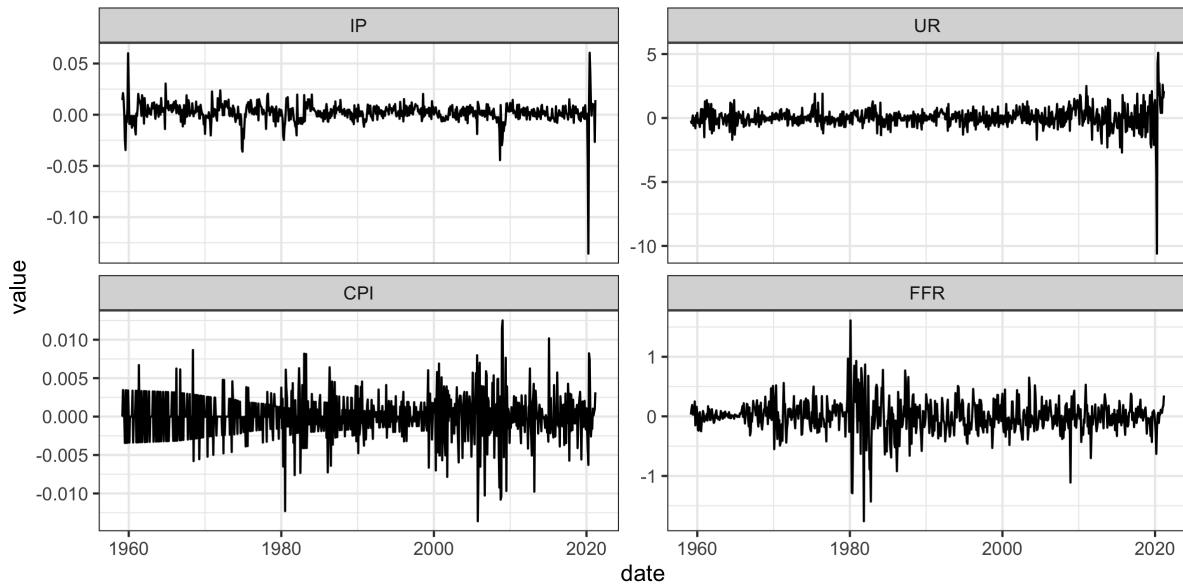


Figure 4: Time series

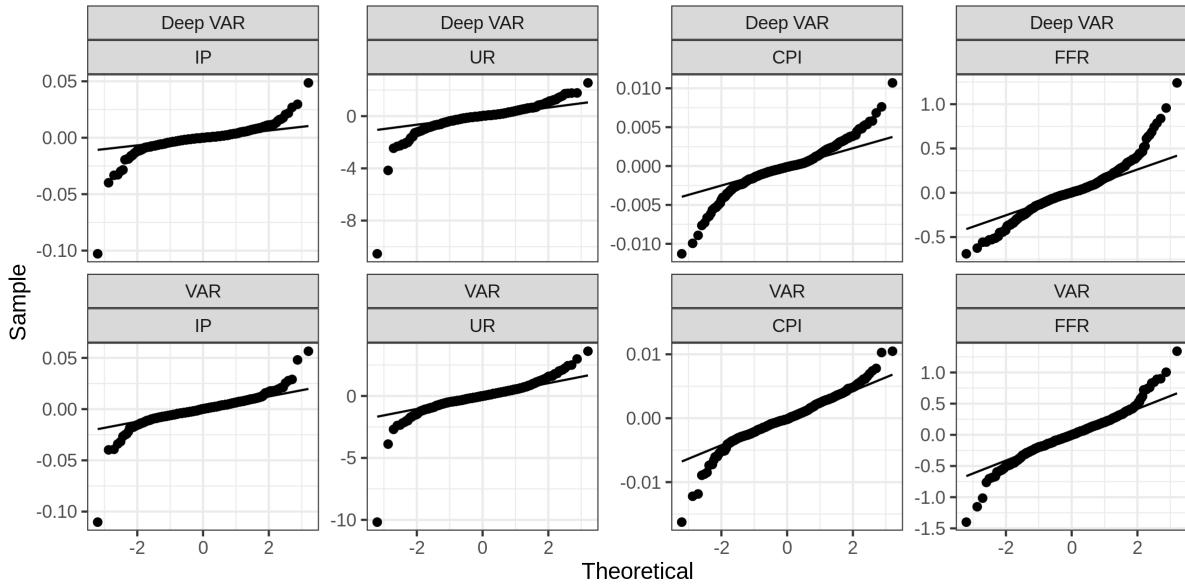


Figure 5: Quantile-quantile plots of full-sample residuals.

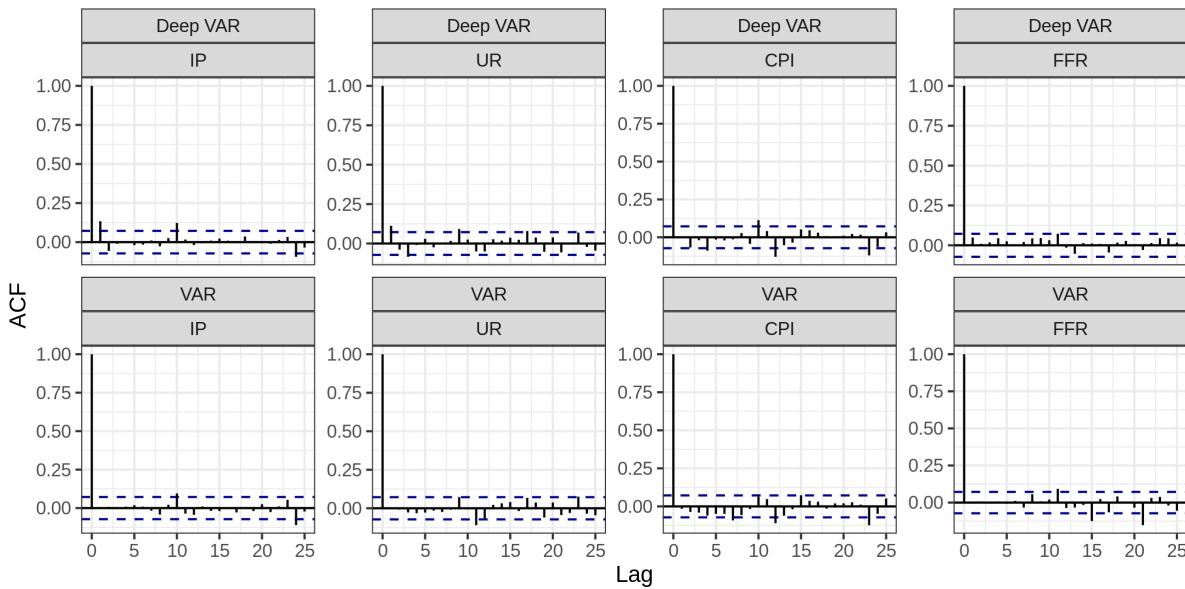


Figure 6: ACF plots of full-sample residuals.

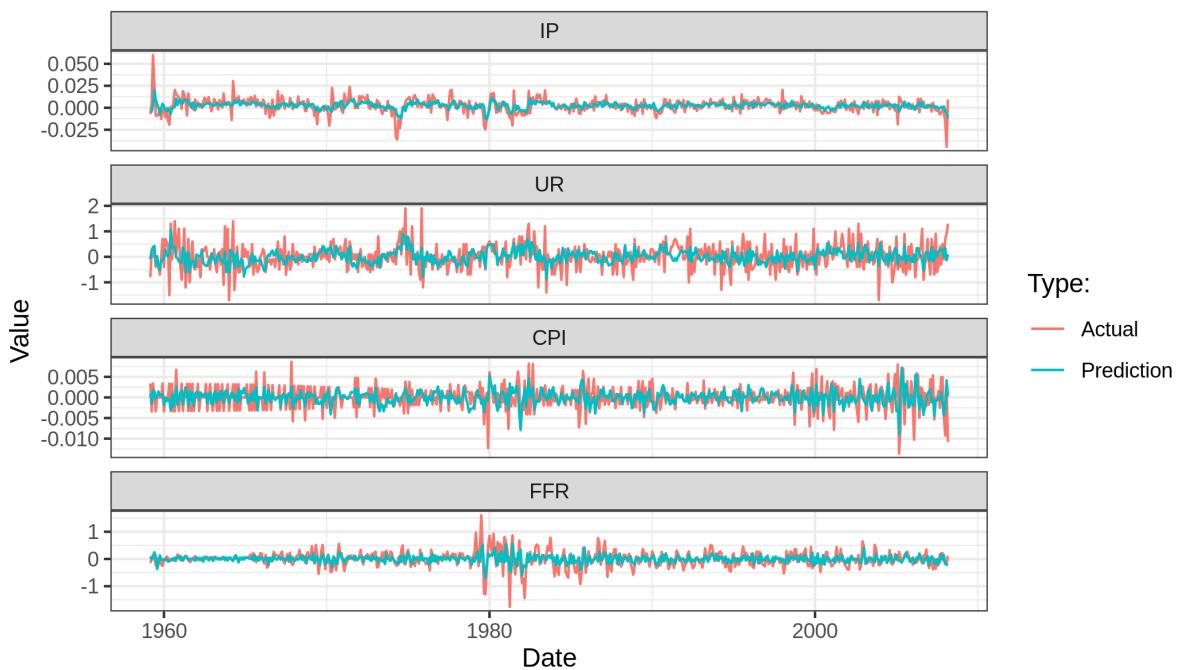


Figure 7: VAR fitted values plotted against observed values for the training sample.

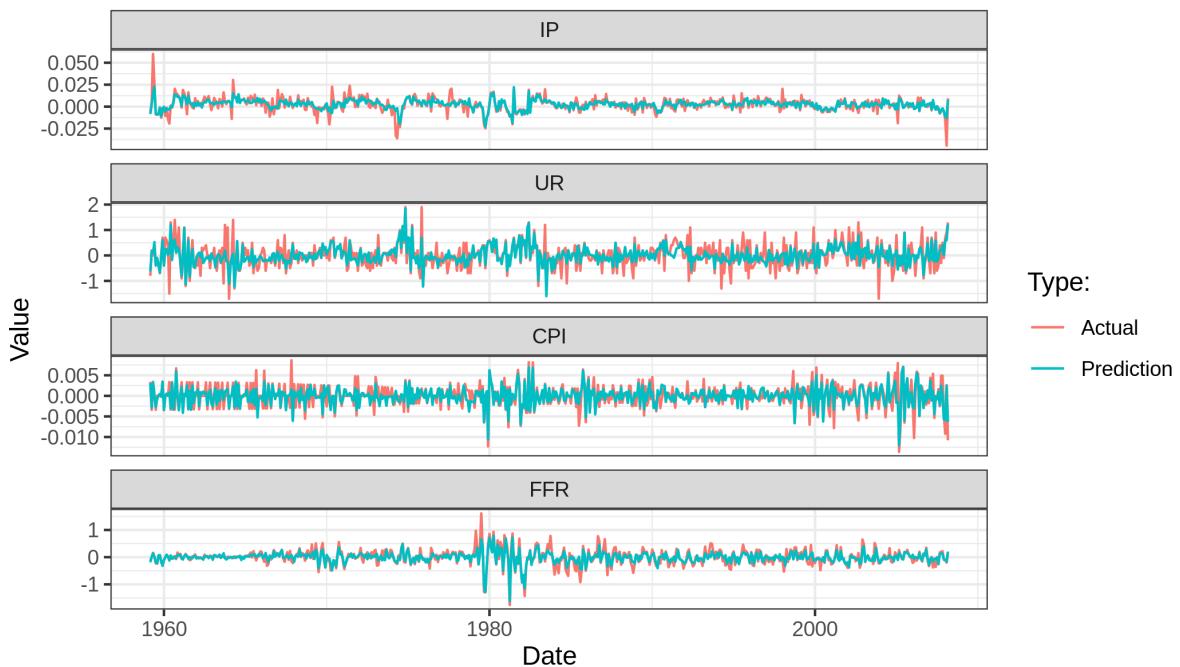


Figure 8: Deep VAR fitted values plotted against observed values for the training sample.

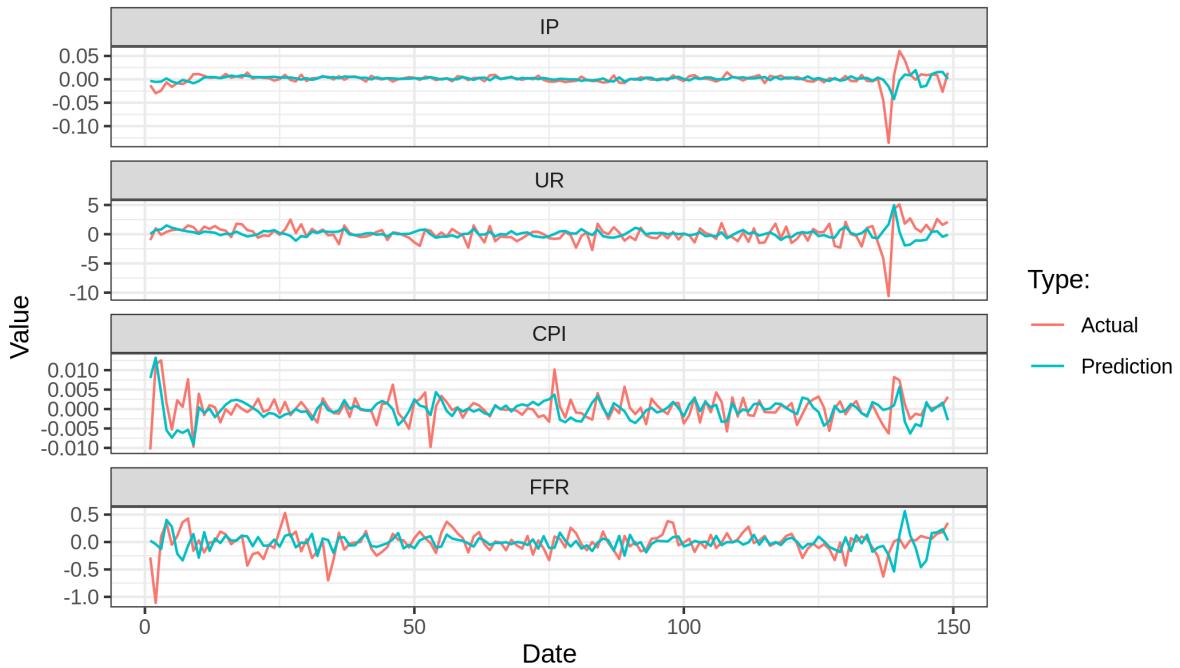


Figure 9: VAR fitted values plotted against observed values for the test sample.

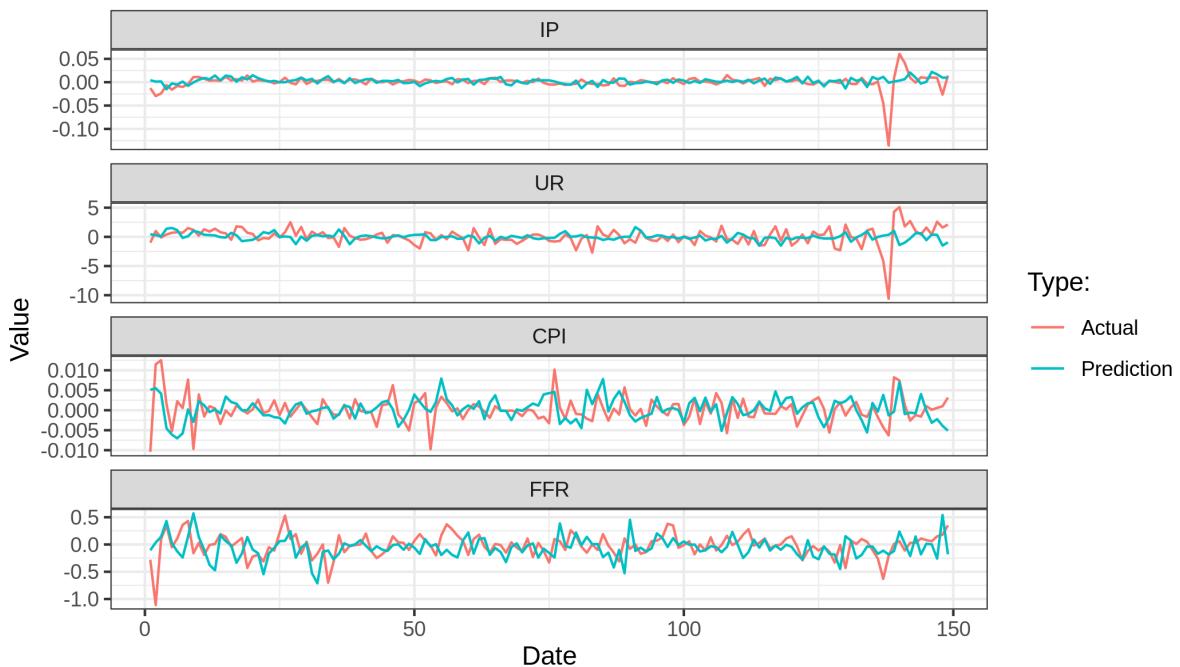


Figure 10: Deep VAR fitted values plotted against observed values for the test sample.

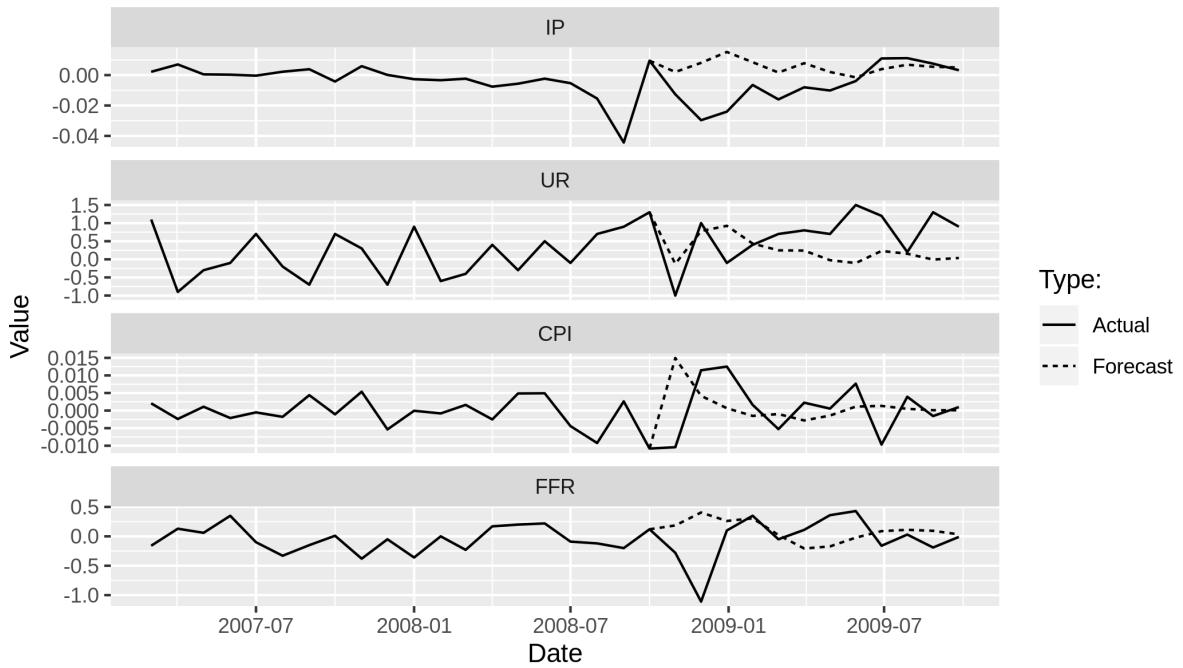


Figure 11: VAR n-step ahead forecasts plotted against observed values. Forecasts are for the first year of the test sample.

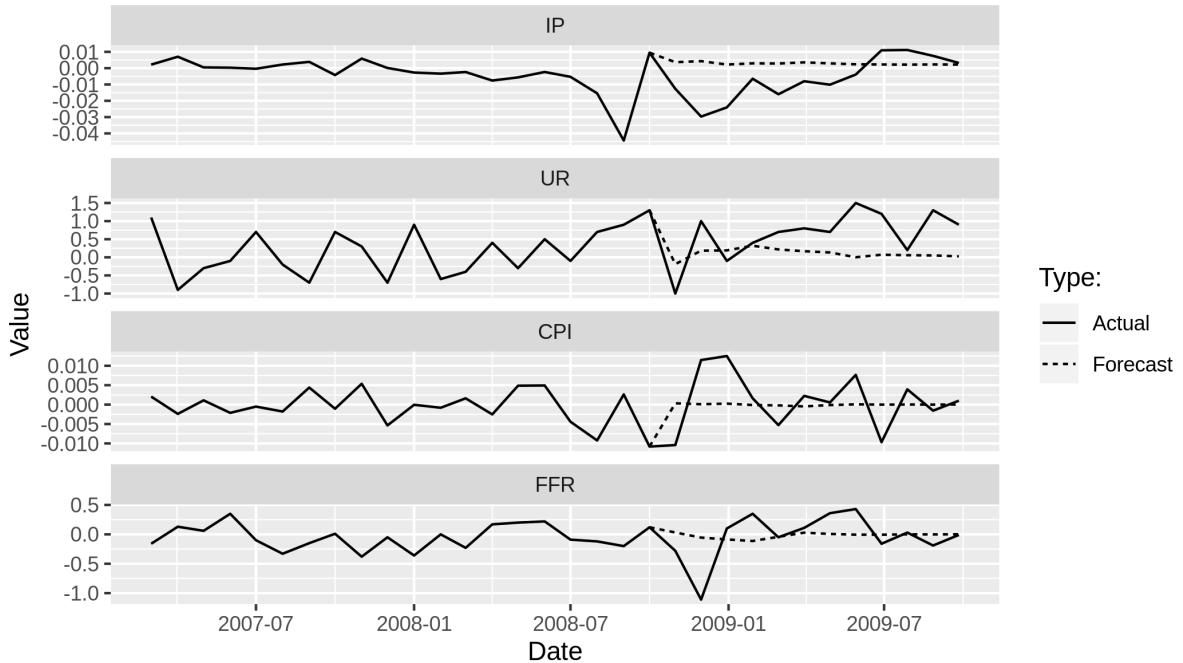


Figure 12: Deep VAR n-step ahead forecasts plotted against observed values. Forecasts are for the first year of the test sample.

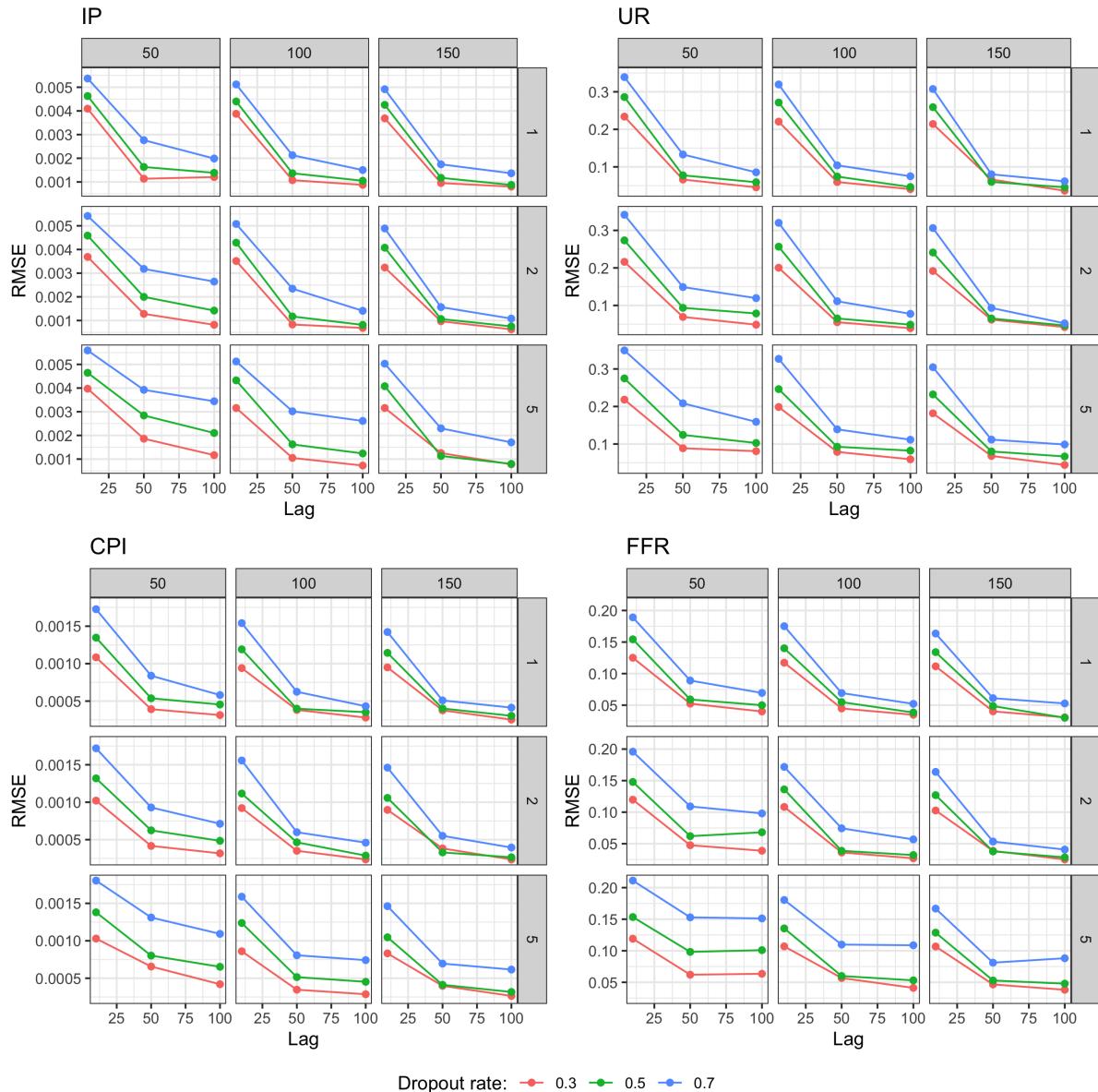


Figure 13: Train sample RMSE for Deep VAR for different variables.

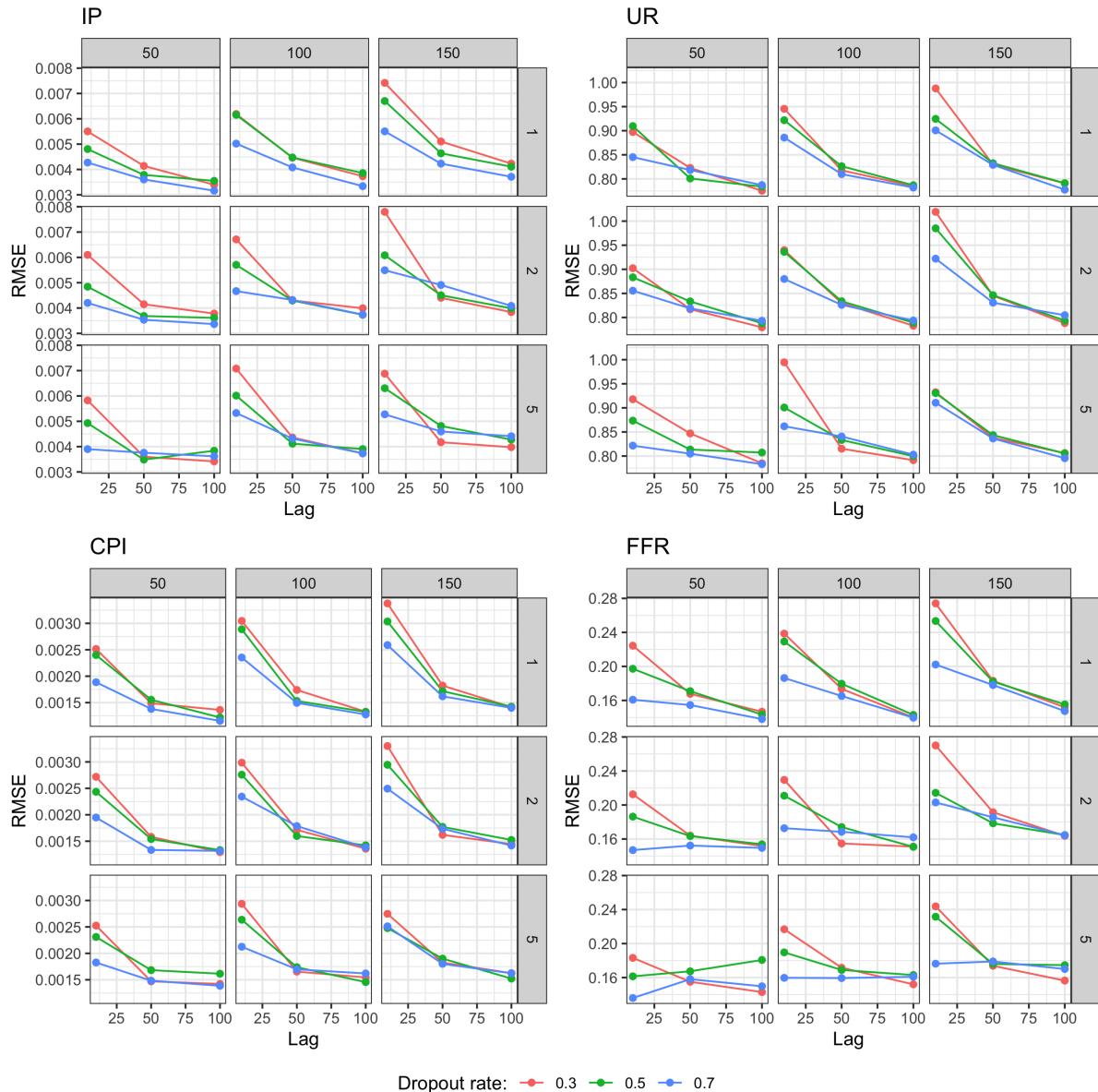


Figure 14: Test sample RMSE for Deep VAR for different variables.

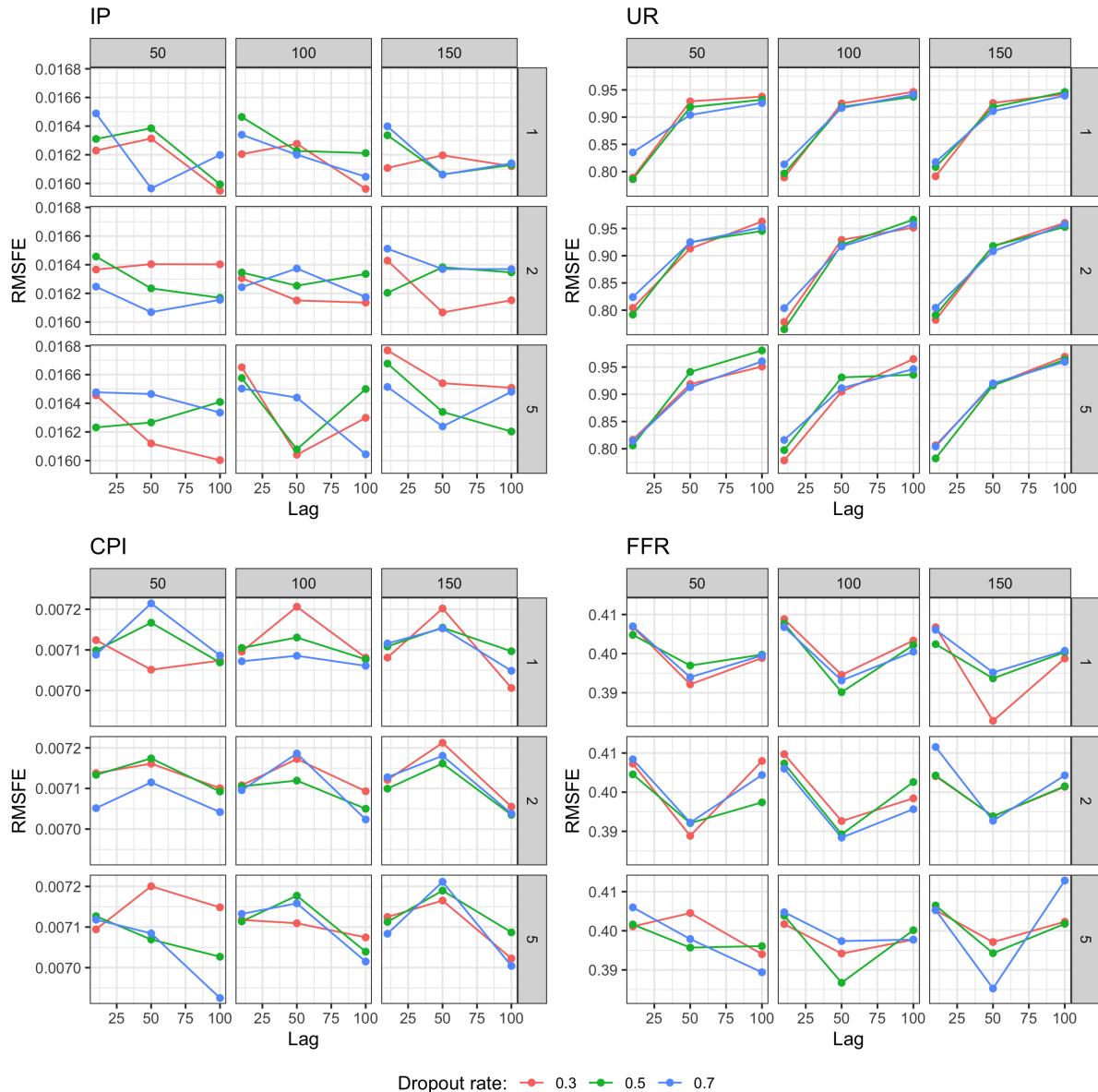


Figure 15: Pseudo out-of-sample RMSFE for Deep VAR for different variables.

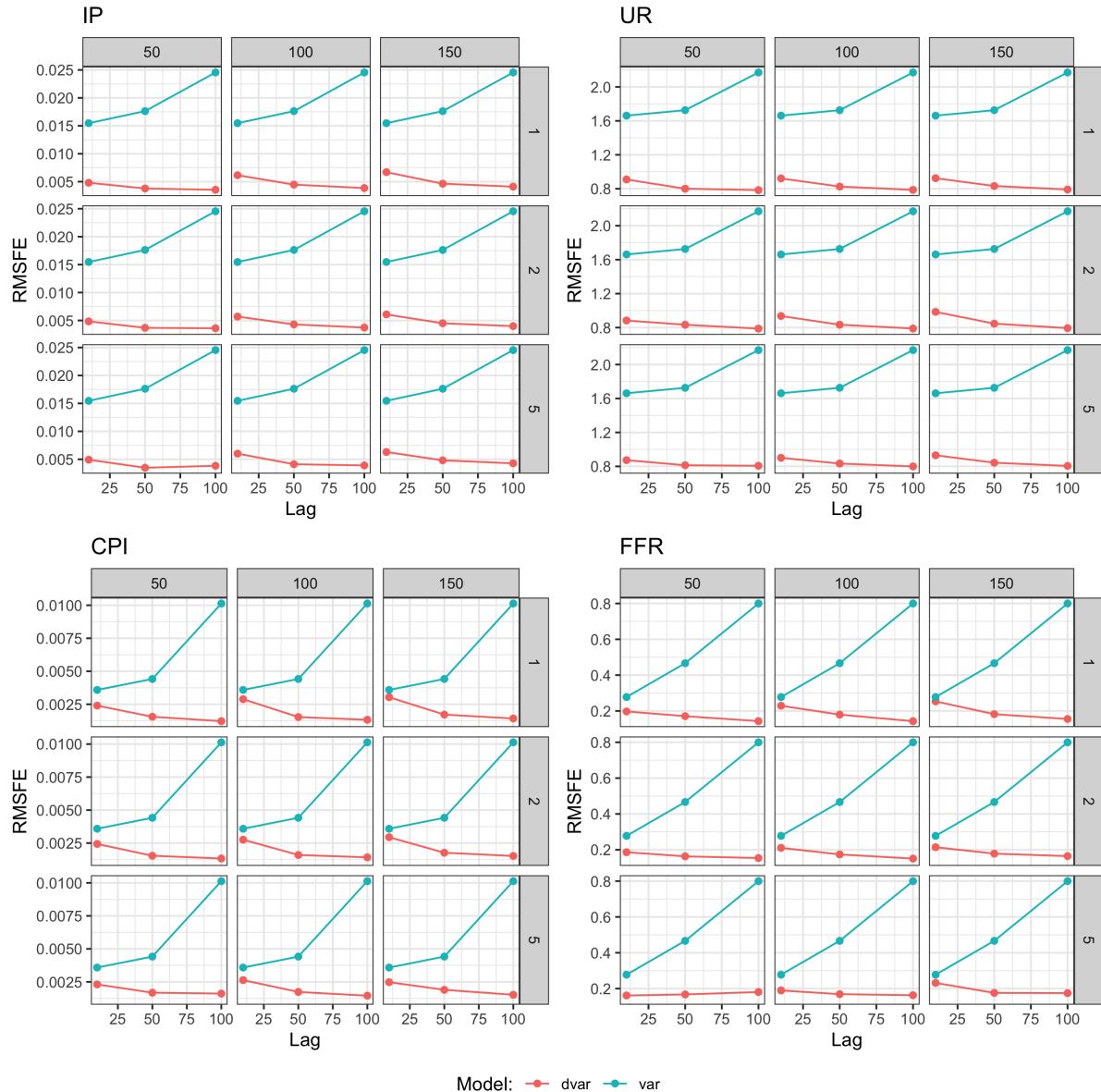


Figure 16: Comparison of out-of-sample RMSE for conventional VAR and Deep VAR for different variables.

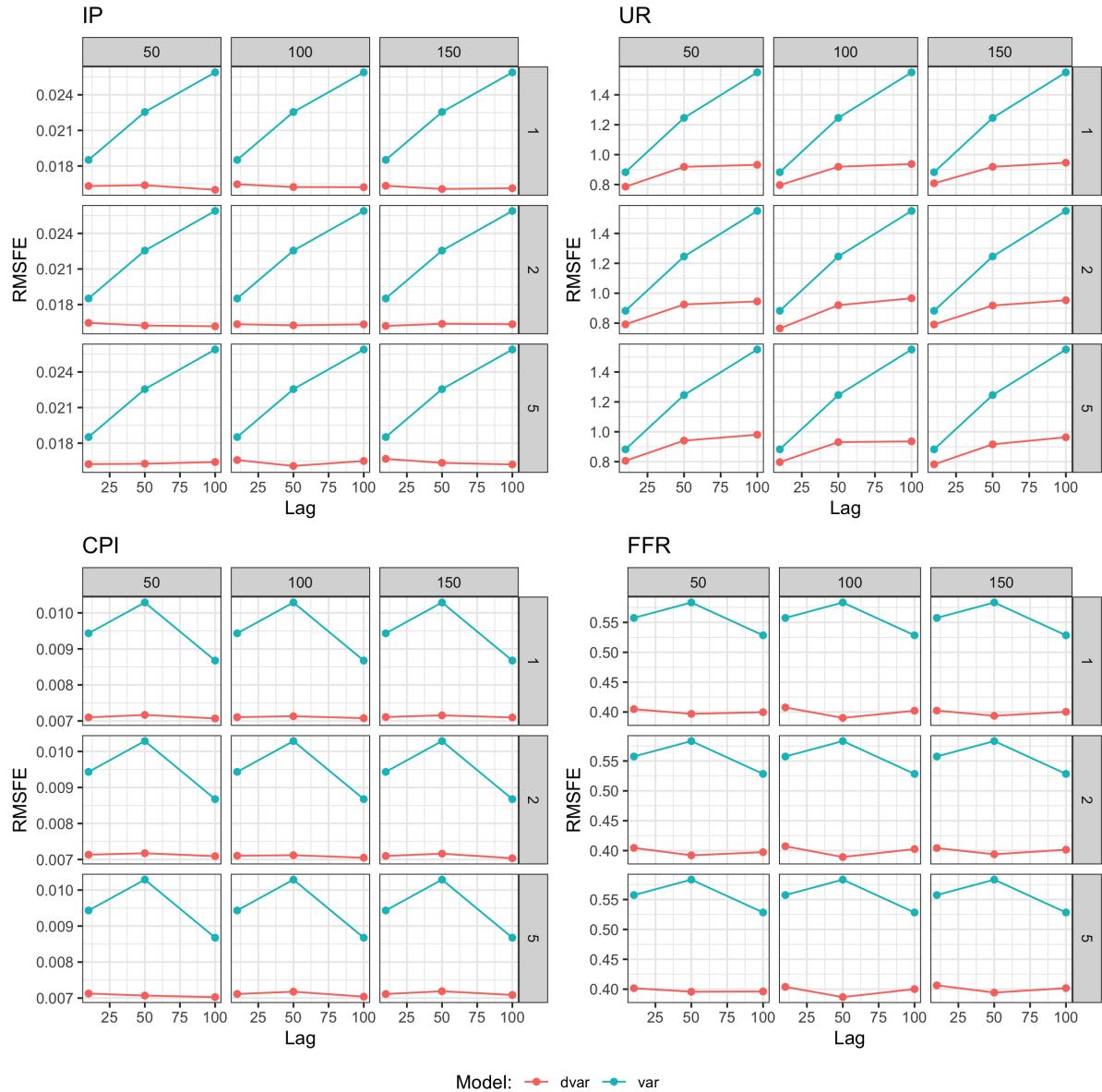


Figure 17: Comparison of pseudo out-of-sample RMSFE for conventional VAR and Deep VAR for different variables.

C R Code and Package

All code used for the empirical analysis presented in this article can be found on the corresponding GitHub repository. Researchers interested in using Deep VARs more generally for their own empirical work may find the R `deepvars` package useful which is being maintained by one of the authors. The package is still under development and as of now only available on GitHub. To install the package in R simply run:

```
devtools::install_github("pat-alt/deepvars", build_vignettes=TRUE)
```

Package vignettes will take you through the basic package functionality. Once the package has been installed simply run `utils::browseVignettes()` to access the documentation.