Modeling Multivariate Time Series in Economics: from Auto-Regressions to Recurrent Neural Networks*

Sergiy Verstyuk†

This draft: 30 April 2020
First draft: 30 November 2018

Abstract

The modeling of multivariate time series in an agnostic manner, without assumptions about underlying theoretical structure is traditionally conducted using Vector Auto-Regressions. They are well suited for linear and state-independent evolution. A more general methodology of Multivariate Recurrent Neural Networks allows to capture non-linear and state-dependent dynamics. This paper takes a range of small- to large-scale Long Short-Term Memory MRNNs and pits them against VARs in an application to US data on GDP growth, inflation, commodity prices, Fed Funds rate and bank reserves. Even in a small-sample regime, MRNN significantly outperforms VAR in forecasting out-of-sample. MRNN also fares better in interpretability by means of impulse response functions: for instance, a shock to the Fed Funds rate variable generates system dynamics that are more plausible according to conventional economic theory. Additionally, the paper shows how, due to its inherent non-linearity, MRNN can discover (in an unsupervised manner) different macroeconomic regimes. Utilizing its state dependence, MRNN may also be a useful tool for policy simulations under practically relevant economic conditions (such as Zero Lower Bound).

*Acknowledgements: Jörn Boehnke, John Cochrane, Bulat Gafarov, Scott Kominers, Jun Liu; as well as the staff of FAS Research Computing at Harvard University.

†Contact address: verstyuk@cmsa.fas.harvard.edu.

Contents

1 Introduction

Artificial Neural Networks are mathematical models for pattern extraction and recognition, a task they achieve by passing their inputs through a series of non-linear transformations and mapping them to targeted outputs; hence the term Deep Learning. Over the last several years this machine learning approach demonstrated impressive progress in image, text/sound and video recognition. Success in image processing became possible due to Convolutional Neural Networks that work well for spatial data. Success in text and sound processing was made to a large extent due to Recurrent Neural Networks that work well for sequential data. This paper investigates ANNs applicability to multidimensional time series data that are encountered in economics, naturally focusing on the RNN family, and on its Long Short-Term Memory member more specifically.

Recurrent Neural Network maintains within its network the connections that look back in time. When processing sequential data, this allows RNN to account for a relatively distant history, to keep track of the context, and to handle data of multiple frequency or scales. For example, in a natural language processing application within the context of zoology-related words such as "cat" or "ears" an RNN after encountering the word "French" would predict a high probability of encountering word "bulldog" next, but within the context of gastronomy-related words such as "bread" or "butter" it would predict instead a high probability of "baguette" coming next.

In this paper we are interested in agnostic dynamic statistical modeling that does not rely on "structural" restrictions stemming from economic theory. Traditionally in economics this is implemented using Vector Auto-Regressions, a simple time-series model that we use as a benchmark. On the other hand, LSTM Multivariate Recurrent Neural Network fulfills the same role, additionally allowing for (i) highly non-linear relationships, as well as (ii) state-dependent dynamics in the data. In fact, a VAR model can be viewed as a special case of MRNN.

The data set we use here comprises GDP growth, inflation, commodity prices, Fed Funds rate, and bank reserves for the United States since 1959 (at monthly frequency, after the aggregation/disaggregation procedures described in the text).

In our exercise MRNN architectures of varying depth (from 1 to 3 hidden layers) and width (from 5 to 1000 units per layer) are considered. In the case of large-capacity NNs, we allow for substantial redundancy, since available regularization techniques such as dropout prevent overfitting quite effectively. MRNN is trained using mini-batch gradient descent. VAR is estimated by OLS, trying various lag lengths. In both cases, the process of parameter learning is guided by prediction performance on the validation sub-sample.

Basically, we run a genuine horse-race between VAR and MRNN, providing two models with the same data sets and letting them use the data in the most effective way their structure allows. Then we compare the fruitfulness of the two modeling approaches.

Our preferred LSTM MRNN architecture is not very deep, comprising 2 hidden layers with 820 units upstream and 660 units downstream. VAR specification we ended up with

comprises 3 lags.

This paper's approach is rooted in modern machine learning that focuses on predicting data out-of-sample, as opposed to more traditional econometrics that deals with fitting data in-sample. Because of this, and also for the reasons of practical interest, the first, quantitative domain for comparing two modeling methods is prediction performance. Our LSTM MRNN outperforms VAR in prediction accuracy on the testing sub-sample (i.e., in generalization) by about 8%, which amounts to 1% confidence in a formal statistical test comparing the two.

The second, qualitative domain is related to an aspect of model interpretation that is revealed by impulse responses. These are a standard exercise for VARs, and in this paper we try out a simple procedure of their implementation for MRNNs. To help identify causal effects of a shock on current and future responses of the variables, we impose a recursive ordering on variables' contemporaneous interrelationships. Specifically, we examine the commonly considered situation of a transitory increase in the Fed Funds rate, and verify which of the two models behaves more in line with economic intuition.

It turns out that following such a monetary tightening, the reactions produced by MRNN are arguably more plausible than those produced by VAR. For the output growth, VAR implies an initial acceleration, which is followed by deceleration later. MRNN implies deceleration unambiguously. As far as inflation rate is concerned, VAR implies its unambiguous acceleration. MRNN is more nuanced, it implies an inflation deceleration in the state of the world characterized by low growth, inflation and policy interest rates, but an acceleration in the state of the world characterized by high growth, inflation and policy interest rates.

Putting this into context, a pick-up in output growth in the wake of monetary contraction is hard to justify theoretically. A positive response of prices to monetary tightening is also a theoretically controversial finding, which has been referred to as a "price puzzle" in VAR literature (Sims, 1992; Uhlig, 2005), although some authors do offer variations of Fisherian arguments to justify it (Cochrane, 2018). Thus, in order to exhibit response dynamics that are more plausible, VARs usually require a careful introduction of additional theoretical ("structural") identification restrictions. However, MRNN appears to be capable of learning plausible reactions essentially from raw data, without the need to impose any theoretical structure beyond a simple recursive ordering of variables.

Next, we move beyond the comparison of the two modeling approaches, and touch upon MRNN interpretations and applications. First, some additional tools for analysis and interpretation. Apparently, MRNN has discovered in the data two different macroeconomic regimes, one characterized by low inflation and interest rates, and another characterized by high inflation and interest rates. Interestingly, it has done so under a completely "hands-off" procedure, without any special modeler's intervention (such as explicitly specifying a regime-switching model).

Second, a real-life primer on policy simulation. Using the MRNN machinery for mod-

eling conditional dynamics, we can examine the effect of monetary tightening in the special circumstances when the Fed Funds rate is close to its Zero Lower Bound. In such case, a monetary tightening shock has a relatively more negative effect on growth but less negative, even slightly positive, effect on inflation rate than in the (closest to ZLB) macroeconomic regime of low growth, inflation and policy rates.

Lastly, it is worth noting that we use a fairly small data set: just 59 years of monthly-frequency data on 5 economic variables. The parameterizations of our two models are quite rich: VAR comprises 80, while MRNN comprises 6698440 parameters. The conventional attitude is skeptical about the reliability of statistical models in such situations, especially with regard to "data-hungry" NN architectures. However, there are theoretical arguments why NNs effective capacity is in practice substantially lower than what a raw parameter count suggests (see the main body); this point is indirectly supported by strong performance of our MRNN in out-of-sample predictions. In any case, we attempted a data augmentation exercise as one way to synthetically expand the limited data set; it did not seem to bring any benefit though (more details are available in Appendix §C).

2 Literature

Artificial Neural Networks and Deep Learning is a very rapidly advancing area of knowledge. Excellent textbook treatment is available in a still relevant Bishop (2006), as well as in a less formal but more up-to-date Goodfellow et al. (2016). A very broad review is available in Schmidhuber (2015), a good review with an eye on applications to Natural Language Processing is Goldberg (2016), a review of Recurrent Neural Networks if offered in Lipton et al. (2015).

Important ANN architectures include Feed Forward Neural Networks that were historically first (single-layer due to Rosenblatt, 1958; multi-layer going back to Ivakhnenko and Lapa, 1965), Convolutional Neural Networks used for image recognition (LeNet introduced by LeCun et al., 1998; AlexNet by Krizhevsky et al., 2012; GoogLeNet by Szegedy et al., 2014), Recurrent Neural Networks used for text/sound processing (Elman, 1990). Here we focus on the latter.[1]

Recurrent Neural Network forms a network that loops to itself, which allows to account for recent history when processing sequential data. Specifically, we focus on Long Short-Term Memory variation of RNN, which has the facilities to prevent the problem of vanishing/exploding error gradients used in the "back-propagation through time" algorithm (Hochreiter and Schmidhuber, 1997b).

Apart from theoretical existence results regarding accurate approximation (Cybenko, 1989; Hornik et al., 1989; also see Kolmogorov, 1957; and Arnold, 1957), there is still

[1]Interestingly, according to Liao and Poggio (2016), RNNs are biologically-plausible models of the cortex—thus closing the loop back to seminal McCulloch and Pitts (1943), whose pioneering NN ideas lied on the interface of biology and computer science.

no consensus about the reasons for NN effectiveness. Possible explanations come from statistical physics (Mehta and Schwab, 2014), geometry (Lei et al., 2018), information theory (Schwartz-Ziv and Tishby, 2017), Bayesian statistics (Patel et al., 2015; Polson and Sokolov, 2017; Friston et al., 2018). Arguably, an important role is played by clever training and regularization techniques (such as Hinton and Van Camp, 1993; Hochreiter and Schmidhuber, 1997a; Srivastava et al., 2014; Zaremba et al., 2015; Mandt et al., 2017; and many others).

Results on consistency, convergence rates, asymptotic normality for neural network estimation are still being refined, contributions range from pioneering White (1989) as well as Chen and White (1999) to very recent Bauer and Kohler (2019), Schmidt-Hieber (forthcoming) and Farrell et al. (2019). Also see Kuan et al. (1994) for early analysis specific to RNNs, as well as Chen (2007) for an excellent treatment of sieve estimation.

Besides well-known practical applications, ANNs gain wide usage in natural science and mathematics. For instance, RNNs have been successfully applied to high-dimensional chaotic systems (Jaeger and Haas, 2004; Vlachas et al., 2018; Pathak et al., 2018). NNs also found application in cryptography (Maghrebi et al., 2016; Cagli et al., 2017), and even to data from geometry and physics that is relevant to string theory (He, 2017).

Economics is also not lagging behind. Pioneering papers dealt with macroeconomic forecasting (Swanson and White, 1997; Chen et al., 2001) as well as financial asset pricing and forecasting (Hutchinson et al., 1994; Swanson and White, 1995; White and Racine, 2001). More recently, ANNs have been used for estimation of causal relationships (Hartford et al., 2016; Farrell et al., 2019), auction design (Dütting et al., 2017), asset pricing and risk management (Heaton et al., 2016a; 2016b; Sirignano, 2016; Feng et al., 2018; Gu et al., 2018).

Considering applications to sequential data, a good source is Graves and Schmidhuber (2009), who use multidimensional LSTM RNN for handwriting recognition. A more recent paper is Goel et al. (2016), with their application of LSTM to multivariate time series from aviation industry. One application to sequential data from the field of finance is Dixon et al. (2018), although instead of RNN the authors use a simpler FFNN with lagged explanatory variables embedded into an input vector.

As a benchmark for comparison, we rely on Vector Auto-Regressions. Loosely speaking, they combine Auto-Regressive Integrated Moving Average models (Box and Jenkins, 1970) and models comprising systems of equations (Theil, 1954; Zellner, 1962; Zellner and Theil, 1962). In economics, this approach was popularized by Sims (1980). Thanks to their simplicity, VARs are still extensively used. A recent review of VAR and time series analysis more generally is available in Stock and Watson (2017); detailed reviews of the so-called structural VARs are offered by Ramey (2016) as well as Stock and Watson (2016); a textbook exposition is offered in Kilian and Lütkepohl. (2017).

In this paper we apply MRNN and VAR models to macroeconomic data from the United States, with an eye on forecasting and policy analysis. Relevant references are

provided in the text sections where these applications are discussed in detail.

3 Methodology

3.1 Framework

First, let us clarify some of the terminology used going forward. We divide the whole universe of Artificial Neural Network models into different architectures (or specifications), and we divide each architecture into its different trained instances.

Given an architecture, the process of hyperparameter tuning (a kind of meta-optimization) defines the parameters ruling the optimization process.[2] Then, optimization, or learning, algorithm executes the model's parameter training (as opposed to estimation in more traditional statistical settings). Next, regularization techniques help with the trained model's parameter refinement (note that optimization and regularization are often conducted in an alternating manner). Lastly, model selection means: (a) for a specific architecture, the choice among trained instances of the best one; and/or (b) between different architectures, the selection of the best one.

3.1.1 Architecture

Artificial Neural Networks, or just nets, are inspired by biological neural networks: units (nodes, cells or "neurons") have connections (edges or "axons", "synapses" and "dendrites") to other units, and they perform complex non-linear computations by transmitting signals from one to another. Mathematically, ANN is a model defined by the following system of equations (or architecture):

$$\begin{aligned}
\boldsymbol{h}_1 &= \boldsymbol{g}_1\left(\boldsymbol{b}_1 + \boldsymbol{W}_1\boldsymbol{x}\right), \\
\boldsymbol{h}_2 &= \boldsymbol{g}_2\left(\boldsymbol{b}_2 + \boldsymbol{W}_2\boldsymbol{h}_1\right), \\
&\cdots \\
\boldsymbol{h}_M &= \boldsymbol{g}_M\left(\boldsymbol{b}_M + \boldsymbol{W}_M\boldsymbol{h}_{M-1}\right), \\
\hat{\boldsymbol{y}} &= \boldsymbol{g}_{M+1}\left(\boldsymbol{b}_{M+1} + \boldsymbol{W}_{M+1}\boldsymbol{h}_M\right),
\end{aligned} \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^{K_x}$ is a K_x-dimensional input vector; $\boldsymbol{b}_m \in \mathbb{R}^{N_m}$ and $\boldsymbol{W}_m \in \mathbb{R}^{N_m \times N_{m-1}}$ are parameters (biases and weights, respectively); $\boldsymbol{g}_m(\cdot)$ are activation functions such that $\boldsymbol{g}_m : \mathbb{R}^{N_m} \mapsto \mathbb{R}^{N_m}$; $\boldsymbol{h}_m(\cdot)$ are outputs of hidden layers; with hidden layers $m \in \{1, \ldots, M\}$ and hidden units $N_m \in \mathbb{N}^+$ (keeping $N_0 := K_x$); as well as $\hat{\boldsymbol{y}} \in \mathbb{R}^{K_y}$ is the output vector aiming to predict the value of a random variable $\boldsymbol{y} \in \mathbb{R}^{K_y}$, with $\boldsymbol{\xi}$ being the prediction error,

$$\boldsymbol{y} := \hat{\boldsymbol{y}} + \boldsymbol{\xi}. \tag{2}$$

[2] Although sometimes hyperparameters are understood as also including the details of the architecture itself.

In this paper, $K_x := K \in \mathbb{N}^+$, and $K_y := K \times H$ with $H \in \mathbb{N}^+$ (in the simplest case, H just equals 1).

The above system can be more compactly expressed as a composition of functions

$$\hat{\boldsymbol{y}} = g_M^{bW} \circ \cdots \circ g_1^{bW}(\boldsymbol{x}), \tag{3}$$

where we used the notation

$$\boldsymbol{g}_m^{bW}(\boldsymbol{x}) := \boldsymbol{g}_m\left(\boldsymbol{b}_m + \boldsymbol{W}_m \boldsymbol{x}\right). \tag{4}$$

Basically, the network passes the incoming data $\boldsymbol{x}$ through several stages of non-linear (semi-linear) transformations, and aims to predict some target $\boldsymbol{y}$. One can think of $\boldsymbol{g}_m(\cdot)$ as units, and of $\boldsymbol{W}_m$ as connections.

For the time-series setup, we are using a particular type of ANN that is known as (Multivariate) Recurrent Neural Network, in particular the specification called Long Short-Term Memory.[3] It specializes the general model along the following lines. First, random variables as well as state variables are time-indexed. Second, the input variable is just a lagged variable that is being predicted by the output, $\boldsymbol{x}_t := \boldsymbol{y}_{t-1}$. Third, $\boldsymbol{g}_{m,t}(\cdot)$ takes a special form (for readability, omitting the layers' subscripts m throughout):

$$\begin{aligned}
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t), \\
\boldsymbol{c}_t &= \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tanh(\boldsymbol{b}_c + \boldsymbol{W}_c \boldsymbol{h}_{-1,t} + \boldsymbol{U}_c \boldsymbol{h}_{t-1}), \\
\boldsymbol{o}_t &= \sigma(\boldsymbol{b}_o + \boldsymbol{W}_o \boldsymbol{h}_{-1,t} + \boldsymbol{U}_o \boldsymbol{h}_{t-1}), \\
\boldsymbol{i}_t &= \sigma(\boldsymbol{b}_i + \boldsymbol{W}_i \boldsymbol{h}_{-1,t} + \boldsymbol{U}_i \boldsymbol{h}_{t-1}), \\
\boldsymbol{f}_t &= \sigma(\boldsymbol{b}_f + \boldsymbol{W}_f \boldsymbol{h}_{-1,t} + \boldsymbol{U}_f \boldsymbol{h}_{t-1});
\end{aligned} \tag{5}$$

where $\boldsymbol{h}_t \in \mathbb{R}^N$ is in this specification interpreted as a cell hidden state vector (more precisely, due to the readability caveat above, $\boldsymbol{h}_t$ stands for $\boldsymbol{h}_{m,t}$, and $\boldsymbol{h}_{-1,t}$ denotes $\boldsymbol{h}_{m-1,t}$, while $\boldsymbol{h}_{0,t} := \boldsymbol{x}_t$); $\boldsymbol{c}_t \in \mathbb{R}^N$ is a cell memory state vector; $\boldsymbol{i}_t \in \mathbb{R}^N$, $\boldsymbol{f}_t \in \mathbb{R}^N$ and $\boldsymbol{o}_t \in \mathbb{R}^N$ are an input gate, forget gate and output gate activation vectors, respectively; and with $\boldsymbol{b}_\cdot \in \mathbb{R}^N$, $\boldsymbol{W}_\cdot \in \mathbb{R}^{N \times N_{-1}}$, and $\boldsymbol{U}_\cdot \in \mathbb{R}^{N \times N}$ being parameters.[4]

Introduction of state vectors allows the time-series models to condition on recent historical dynamics. It may be helpful to think of LSTM network as a kind of non-linear state-space model (that are very easy to work with due to Kalman-Bucy filter as described, for example, in Hamilton, 1994).

3.1.2 Mechanics

An important theoretical result about NNs is that their formulation is general enough to represent any desired function. Given at least one hidden layer and a sufficiently large but

[3] In the specification's title, "long memory" stands for slowly-changing parameters of the ANN, "short memory" for recurrent inputs to the hidden layer from its past outputs, and "long short-term memory" for the amendment of recurrent connections with additional parametric network structures.

[4] Symbol $\odot$ denotes element-wise multiplication (also called Hadamard product).

finite number of units, this is feasible by the arguments of the universal approximation theorem (Cybenko, 1989; Hornik et al., 1989; also see the representation theorem due to Kolmogorov, 1957; and Arnold, 1957).[5] A useful implication is NNs' ability to account for non-linearity of the empirical phenomena being modeled.

Size (or capacity, parameterization) of a NN is determined by its depth, that is the number of hidden layers M, and its width, that is the number of units in each layer $\{N_1, \ldots, N_M\}$. NNs are called deep if they have more than one hidden layer, otherwise they are called shallow.

The above theory requires a large enough NN, but it does not provide us with the formal method for setting a sufficient size and structuring the architecture (and, more generally, for learning the required parameters).[6] Our guiding principle is given by what is regarded as the scientific method: successful prediction on the new data. A theory or model is good only if it generalizes the existing facts in a way that is useful in predicting yet unobserved phenomena.

From the theory, we know that an insufficiently sized NN will underfit the data: the standard statistical fit measures will demonstrate poor performance, even on the observed sample that is used for training ("in-sample"). At the other extreme, a too large NN may overfit the data: statistical fit measures will demonstrate strong performance on the observed sample that was used for training, but will collapse when the model is tested on the previously unobserved data ("out-of-sample"). In the machine learning literature, we say about the latter case that the NN has memorized the data without learning its high-level structure and thus acquiring the ability to generalize the accumulated knowledge.

Hence, the strategy commonly adopted in the literature is extensive exploration that amounts to experimenting with different model specifications, using as a criterion that guides the exploration the predictive performance of the specifications considered. Usually, the predictive fit is measured on a sample reserved exclusively for such validation purposes (and that is separate from the training and testing samples).

One practical approach is to start with a generously sized network, allowing for redundancy and erring on the side of a too many parameters, and then restricting the excessive depth/width using the regularization methods described below.[7] Another practical rule of thumb is that deep architectures tend to be easier and cheaper to train, hence their popularity even though in theory having just one hidden layer is sufficient.

[5] Formally, a function that is Borel measurable and defined over finite-dimensional spaces (e.g., continuous on a compact subset of $\mathbb{R}^n$) can thus be approximated with arbitrary precision.

[6] Technically, the above theoretical result is not constructive, it is only about the existence.

[7] Existence of effective regularization methods is the reason why overfitting is less problematic than underfitting (this asymmetry is a little reminiscent of inefficiency versus inconsistency in the practice of estimating, e.g., linear regression models, hence the prudent preference for including rather too many than too few explanatory variables or "correcting" the consistent but inefficient estimates' variance-covariance matrix). Also, allowing for redundancy improves NN reliability and makes it easier to train.

Regularization: The aim of regularization is to improve generalization and reduce overfitting of a given model. The leading motive is Occam's razor principle: the heuristic postulate that good scientific theories are parsimonious. Regularization downplays the weight of the training sample in parameter learning by introducing additional information and constraints, which tends to prevent the fitting to noise and memorization at the expense of generalization.

The recent literature provides many effective regularization techniques. Some of the most popular methods are explicitly using the idea that good models should be invariant to "small" perturbations.

Dropout adds noise to the NN architecture (Srivastava et al., 2014; also see Zaremba et al., 2015). During the training process, it randomly drops from the model some units and connections (usually in the hidden layer, but possibly in the visible one too). This is a very simple yet highly effective regularization method. It can be loosely motivated as mimicking the model averaging (as the authors of Srivastava et al., 2014, originally suggested) as well as estimation under LASSO/sparsity or Ridge constraint (Wager et al., 2013).

Data augmentation adds noise to the NN inputs (the idea can be traced back to Sietsma and Dow, 1991).[8] This expands the data set with synthetic data comprising the originals that are contaminated with some (usually, Gaussian/white) noise. In such case, effectively we are dealing with a kind of background noise. But noise can also be added exploiting the structure of the data at hand. For visual data, it involves image transformations such as translations, rotations, scaling, cropping, color perturbations (LeCun et al., 1998; Simard et al., 2003; Krizhevsky et al., 2012). For audio data, it involves sound transformations such as frequency, tempo and speed perturbations (Jaitly and Hinton, 2013; Cui et al., 2015; Ko et al., 2015).

Optimization and tuning: NN models are very costly to work with: usually they are truly high-dimensional and have a large number of parameters; they have non-linear structure and non-convex objective functions; they often involve processing very large datasets. Thus, careful design of the whole computational approach is of paramount importance. This paper leaves out the discussion of the hardware and software components (necessary resources such as Graphics Processing Units and machine learning libraries are now widely available), and focuses instead on the algorithmic component that comprises data handling, minimization algorithms, meta-optimization of the latter, etc.

We briefly present some of the techniques that proved to be useful in training NN models. First and foremost, their aim is to speed up training and reduce computational costs.[9] However, many of the below techniques involve replacing a true algorithm or object

[8]Data augmentation is often viewed intuitively as an addition of carefully perturbed synthetic data to the existing data set. Formally, however, it is a regularization method that encourages fitting to relevant features of the data rather than to irrelevant noise.

[9]Hence, such techniques are often just practical tricks that are employed only out of necessity and

with some reasonable approximation, and these techniques' effectiveness is possible only if they (or the respective downstream algorithms) are invariant to perturbations and robust to random noise that necessarily show up. In other words, along the way these speed-up techniques also act as regularizers.

The loss minimization algorithms in the existing literature are usually variations of a standard gradient descent with some stochasticity added (for an early source, see Robbins and Monro, 1951; for a Bayesian interpretation see Mandt et al., 2017).

The granularity of data inputs used for gradient computations and parameter updates vary from utilizing the full data set ("batch gradient descent") to several ("mini-batch gradient descent") or single ("stochastic gradient descent") randomly chosen block(s) of observations (that is, training example(s)). Smaller batches prevent using the most efficient computational tools and exhibit noisier convergence; but such built-in stochasticity helps avoiding local minima, hence works as a regularizer (see Wilson and Martinez, 2003).

Data standardization is useful in many statistical learning problems, but in the context of NNs it is even more critical. One development of the general standardization idea that is tailored for NNs is separately standardizing the inputs to intermediate hidden layers ("batch normalization", see Ioffe and Szegedy, 2015).

Minimization algorithms make use of efficient ways for computing gradients that improve upon the brute-force implementation of canonical chain rule ("backward propagation of errors", or just "backpropagation", see Rumelhart et al., 1986).

There are also methods used for ensuring minimization algorithms' satisfactory convergence, i.e. one that is fast and avoids bad local minima. These include clever ways of updating parameters in the face of stochasticity inherent to minimization algorithms used: the learning rate is high far from the minimum in order to ensure fast learning, but falls as the minimum is approached to prevent overshooting ("learning rate decay", see Robbins and Monro, 1951; Welling and Teh, 2011); parameter updating maintains memory of recent updates in order to keep improving in the promising direction ("momentum", see Bengio et al., 2012; Sutskever et al., 2013). Also, simple as it is, an outright termination of the algorithm's iterations well before the minimum is reached saves on computational costs with little downside ("early stopping", see Prechelt, 1997). Importantly, all of the above methods contribute to regularization as well.

Many of the methods above have their own hyperparameters (random seeds, dropout probability, learning rates, etc.). The hyperparameters' initial values—and sometimes modification schedules too—are also important choices to be made.

"Overparameterization": A common question about NNs (e.g., see Zhang et al., 2017; Belkin et al., 2018) is concerned with the fact that in overwhelming majority of applications the number of NN parameters exceeds the number of training observations by several orders of magnitude — what are the underlying mechanisms that allow NNs

computational resource constraints.

with large explanatory capacity that fit arbitrarily accurately on the training sample also to perform well in the test sample, i.e. to properly generalize?

Formally, the basic explanation is that all these parameters are not entirely free. For instance, there are some trivial (as well as rather profound) symmetries and equivalences in the space of NN weights (Chen et al., 1993; Kurková and Kainen, 1994). Furthermore, parameters of properly trained NNs turn out to lie in a lower-dimensional subspace (see the "noise stability" property of Arora et al., 2018; also refer to Hochreiter and Schmidhuber, 1997a, Hinton and Van Camp, 1993, for a related argument).

Fundamentally, it is often more useful to think about NNs non-parametrically. Thus, identifiability of specific parameters is not even a relevant issue to consider, and instead we are interested in higher-level objects: in model outputs and their characteristics (e.g., forecasts and uncertainty about them).

3.2 Data

Our raw data set comprises: Real GDP, seasonally adjusted annual rate, and GDP Price Deflator, seasonally adjusted (both at quarterly frequency)[10]; Equal Weight Continuous Commodity Index (at daily frequency); Effective Fed Funds Rate and Nonborrowed Reserves (at monthly frequency). Additional details are available in Appendix §A.

Daily frequency data are aggregated into monthly by averaging, while quarterly frequency data are disaggregated into monthly using a linear state-space model (as in Bernanke et al., 1997). Our interpolation procedure is described in Appendix §B.

Next, data are transformed as follows: Real GDP, GDP Deflator and Commodity Price Index into month-over-month growth rates; Nonborrowed Reserves into monthly differences. See Figures 6 and 7 in Appendix §A for visual representations of monthly-frequency data prior and after the transformations, respectively.[11] (Finally, to prevent numerical complications, the training itself is done in terms of data series that are standardized to have a mean of 0 and a variance of 1.)

The resulting data set covers the time period from 1959:M02 to 2019:M06.

3.3 Implementation

We split those 60 years of monthly data into 3 sub-samples:

(i) training sample used for parameter learning (1959:M02 to 2011:M06);

(ii) validation sample used for adjusting the learning rate in order to prevent data memorization (2011:M07 to 2015:M06);

[10]Seasonally adjusted data are used for simplicity, because ANNs' treatment of seasonality in a small-sample regime as here is an important research problem in its own right and deserves a separate study.

[11]Notice the effect of quantitative easing policy on Nonborrowed Reserves at the end of the sample period.

(iii) testing sample that the NN does not see until we are finished with learning and start benchmarking NN's usefulness (2015:M07 to 2019:M06).

Here we have sequential data, i.e. each observation is dependent on other observations immediately prior to it. Hence, one training example includes several (specifically, 24) consecutive observations: $\{\boldsymbol{y}_{t-i}\}_{i=0}^{23}$.

Predictions over the validation and testing sub-samples are made multiple steps ahead. Non-linearity of NNs warrants direct—as opposed to iterated—forecasts for each point in the prediction horizon.[12] Each sub-sample is split in half and predictions are 24-steps-ahead, thus reserving 2×24 months for validation and 2×24 months for testing.

We use the "golden-standard" mini-batch gradient descent, with 4 training examples per batch. Then, 1 training epoch is composed of a full run through the training data set, thus feeding the training examples for all possible t: $\{\{\boldsymbol{y}_{t-i}\}_{i=0}^{23}\}_{t=24}^{629}$.

Our NN has a general LSTM architecture introduced earlier. More specifically, NN cells in the visible input layer take in the 5-dimensional data vector (so, $K := 5$) and pass it on untransformed. The cells in the hidden layers use Rectified Linear Units (ReLU) as the activation functions applied to their inputs. The cells in the visible output layer apply simple linear transformations as activation functions on their inputs from the last hidden layer. With the Mean Squared Error loss function ultimately used in our NN (i.e., $\mathrm{L}(\boldsymbol{y}, \hat{\boldsymbol{y}}) := (1/TK) \sum (\boldsymbol{y}_t - \hat{\boldsymbol{y}}_t)^\intercal (\boldsymbol{y}_t - \hat{\boldsymbol{y}}_t)$), in this final visible layer we are effectively just running Ordinary Least Squares regressions on the layer's inputs. To produce direct forecasts, the NN outputs a $K \times H$ object $[\hat{\boldsymbol{y}}_{\tau+1}, \ldots, \hat{\boldsymbol{y}}_{\tau+H}]$, with $K = 5$ and $H = 24$.

Within certain range, we try architectures of varying depth and width. The depth ranges from 1 to 3 hidden layers. Due to forecast horizon length, the number of outputs exceeds the number of inputs, hence we treat the width asymmetrically, varying it from 5 to 1000 neurons in the first hidden layer, but from 100 to 1000 in the last hidden layer (if there are more than one layer).

The minimization algorithm used is "adaptive moment estimation", or Adam (Kingma and Ba, 2014). Each architecture is trained for 10 epochs, with a decaying learning rate. Lastly, the dropout probability is set at 0.2.[13]

To sum up, our implementation recipe has four stages: (i) set hyperparameters (which requires some tuning); (ii) define specification, train and regularize it; (iii) repeat the previous step for a range of specifications; (iv) choose specification with the best training fit.[14]

[12]The formal reason is non-commutativity: even with one hidden layer $g_1^{bW}(\cdot)$ and independent mean-zero prediction errors $\boldsymbol{\xi}$, $\mathrm{E}_t[\boldsymbol{y}_{t+1}] = \mathrm{E}_t[g_1^{bW}(\boldsymbol{y}_t) + \boldsymbol{\xi}_{t+1}] = g_1^{bW}(\boldsymbol{y}_t)$, but already $\mathrm{E}_t[\boldsymbol{y}_{t+2}] = \mathrm{E}_t[g_1^{bW}(g_1^{bW}(\boldsymbol{y}_t) + \boldsymbol{\xi}_{t+1}) + \boldsymbol{\xi}_{t+2}] = \mathrm{E}_t[g_1^{bW}(g_1^{bW}(\boldsymbol{y}_t) + \boldsymbol{\xi}_{t+1})] \neq g_1^{bW}(\mathrm{E}_t[g_1^{bW}(\boldsymbol{y}_t) + \boldsymbol{\xi}_{t+1}]) = g_1^{bW}(g_1^{bW}(\boldsymbol{y}_t))$. See Tong (1990) for a theoretical analysis; also see Marcellino et al. (2006) for empirical comparison in the context of linear models.

[13]Additional hyperparameter settings are available upon request.

[14]We do not use as a criterion for choosing the best NN specification the validation sample fit (because our validation set is too small for that) or the best training epoch (because learning rate decay makes this

3.4 Benchmark

For comparison, we use a standard Vector Auto-Regressive model. VAR model is defined by the following vector-valued equation:

$$\hat{\boldsymbol{y}}_t = \boldsymbol{a} + \boldsymbol{A}_1 \boldsymbol{y}_{t-1} + \ldots + \boldsymbol{A}_L \boldsymbol{y}_{t-L}, \tag{6}$$

where $\boldsymbol{y}_{t-\ell}$ are lags of a K-dimensional random variable $\boldsymbol{y}_t$; while $\boldsymbol{a} \in \mathbb{R}^K$ and $\boldsymbol{A}_\ell \in \mathbb{R}^{K \times K}$ are parameters; with lag $\ell \in \{1, \ldots, L\}$; and $\hat{\boldsymbol{y}}_t$ on the left-hand side has the same interpretation as before in equation (2), with the prediction error $\boldsymbol{\epsilon}_t$.

More compactly, for $\boldsymbol{A} := [\boldsymbol{A}_1, \ldots, \boldsymbol{A}_L]$ and $\boldsymbol{x}_t := [\boldsymbol{y}_{t-1}; \ldots; \boldsymbol{y}_{t-L}]$, and following the notation from (4), this becomes

$$\hat{\boldsymbol{y}}_t = \boldsymbol{a} + \boldsymbol{A}\boldsymbol{x}_t = \mathrm{id}^{aA}(\boldsymbol{x}_t), \tag{7}$$

with the identity function defined as

$$\mathrm{id}(\boldsymbol{x}) := \boldsymbol{x}.$$ [15]

From this perspective, it becomes clear that VAR can be seen as a restricted version of a more general multivariate NN model (with MRNN and LSTM MRNN adding further enrichments).

We apply this model to the data set described earlier. Thus, $K = 5$ again. We split the sample the same way as before into training (1959:M02 to 2011:M06), validation (2011:M07 to 2015:M06) and testing (2015:M07 to 2019:M06) sub-samples. Predictions over the validation and testing sub-samples are made iteratively 24 steps ahead. Parameters are estimated by Ordinary Least Squares.

Importantly, we follow the "predictive" approach from above and choose the optimal lag length L by comparing the predictive performance on the validation set. Such an explicitly predictive approach differs from the more traditional lag selection practice in VAR literature that is based on various information criteria: the latter also hopes to achieve the generalization ability, albeit indirectly and implicitly. (In the interest of transparency, below we also provide the results for the traditional approach: it selects the lag length using Akaike Information Criterion, combining the training and validation sub-samples together into a larger estimation sample.)

4 Results

This is the main section of the paper. Here we present the training results for all NN models considered and select the best performing model. Then we demonstrate our preferred model's capabilities: (i) its out-of-sample forecasting performance, and (ii) several

extraneous). Also, our search for the best model is based on trying many different architectures rather than many different parameter starting values.

[15] In AR(1) terms, for $\boldsymbol{z}_t := [\boldsymbol{y}_t; \ldots; \boldsymbol{y}_{t-L+1}]$, $\boldsymbol{\zeta}_t := [\boldsymbol{\epsilon}_t; 0; \ldots; 0]$, $\boldsymbol{d} := [\boldsymbol{a}; 0, \ldots, 0]$ as well as $\boldsymbol{D}$ appropriately combining $\boldsymbol{A}$, $\boldsymbol{I}_K$ and $\boldsymbol{0}_K$, we can write $\boldsymbol{z}_t = \boldsymbol{d} + \boldsymbol{D}\boldsymbol{z}_{t-1} + \boldsymbol{\zeta}_t = \mathrm{id}^{dD}(\boldsymbol{z}_{t-1}) + \boldsymbol{\zeta}_t$.

practical ways of interpreting the workings of the NN "black box" (specifically, inspection of its steady states as well as computation of unconditional or conditional impulse response functions).

4.1 All nets

After tuning the hyperparameters, we trained and regularized a range of architectures, from a modest NN with just 1 hidden layer of 5 units to a fairly rich one comprising 3 hidden layers with 1000 units in each. Table 1 summarizes their fitting and prediction performance, measured in terms of Mean Squared Errors for standardized series, over the training, validation and testing samples.

Evidently, a lot of architectures demonstrate commensurate, potentially complementary performance.[16] Moreover, notice that specifications with larger capacities not only exhibit good training sample fit, but also do not necessarily underpeform in their testing sample predictions — this corroborates the effectiveness of our regularization techniques.

4.2 Best net

4.2.1 Forecasting

We take the trained NN instance with the lowest training sample MSE as our preferred NN architecture.

With the MSE of 0.5159, this turns out to be a not very deep specification featuring 2 layers, which contains 820 units upstream and 660 units downstream. Under the hood, such a specification comprises 6698440 NN parameters in total (refer to equations 1 and 5):

- $(820+820\times5+820\times820)$ + $(820+820\times5+820\times820)$ +$(820+820\times5+820\times820)$ + $(820 + 820 \times 5 + 820 \times 820) = 2709280$ parameters in the first hidden layer;
- $(660+660\times820+660\times660)$ + $(660+660\times820+660\times660)$ + $(660+660\times820+660\times660)$ + $(660 + 660 \times 820 + 660 \times 660) = 3909840$ parameters in the second hidden layer; and
- $(120 + 120 \times 660) = 79320$ parameters in the visible output layer.

The validation sample predictive fit is 2.2601 (multiple-steps-ahead, accuracy measured in MSE terms). This is worse than in some of the alternative trained architectures; the minimum is 1.9798 for a 3-layered NN with 50, 50 and 880 units from top to bottom layer. However, given that our validation sample is very small due to data limitations, it is

[16] Such a situation usually calls for ensemble averaging approaches. However, given that we are already using dropout regularization, which effectively conducts a kind of model averaging, we will abstain from additional mixing.

Table 1: Long Short-Term Memory Multivariate Recurrent Neural Nets, Prediction results

	I				II				III			
	N_1	Train.	Valid.	Test.	N_1-N_2	Train.	Valid.	Test.	N_1-N_2-N_3	Train.	Valid.	Test.
Shape of L(·)												
Max	n/a	0.7569	2.5539	1.8698	n/a	1.032	3.4149	2.5604	n/a	6.7058	4.5366	3.2659
Average	n/a	0.5763	2.2653	1.7086	n/a	0.6580	2.2253	1.6984	n/a	0.7029	2.2255	1.7151
Min	n/a	0.5217	2.0958	1.6227	n/a	0.5159	2.0188	1.5958	n/a	0.5171	1.9798	1.5976
St.Dev.	n/a	0.0402	0.1083	0.0514	n/a	0.0720	0.1288	0.0589	n/a	0.1378	0.2091	0.1094
Minimizers of L(·)												
in Training	880	0.5217	2.2525	1.6479	820-660	0.5159	2.2601	1.6556	800-940-940	0.5171	2.1034	1.6938
in Validation	10	0.6914	2.0958	1.7619	30-400	0.6537	2.0188	1.6588	50-50-880	0.7699	1.9798	1.6779
in Testing	940	0.5319	2.2328	1.6227	760-520	0.5567	2.1574	1.5958	10-10-780	0.8333	2.1106	1.5976

Notes: Vertical panels I to III present the results for NNs of different depth, from 1 to 3 hidden layers, respectively. For each depth, all architecture permutations are considered, with numbers of units chosen from the following sets: $\{5, 10, 20, 30, \ldots, 80, 90, 100, 120, 140, \ldots, 980, 1000\}$ for the first layer N_1; $\{100, 120, 140, \ldots, 980, 1000\}$ for the last layer N_2 in II or N_3 in III; and $\{\min(N_1, N_3), \lceil \frac{N_1+N_3}{2} \rceil, \max(N_1, N_3)\}$ for the middle layer N_2 in III. The rows present: descriptive statistics on the shape of the loss function L(·) in the training, validation and testing sub-samples; the results for NNs minimizing L(·) in the training, validation and testing sub-samples. Results are given in terms of MSEs (calculated for standardized series, averaged over variables and time periods). Predictions are 1-step-ahead in the training sub-sample, but direct 24-steps-ahead in the validation and testing sub-samples. Additional technical details about the implementation are provided in the main text.

Table 2: Vector Auto-Regression, Prediction results

Criterion	L	Training	Validation	Training&Validation	Testing
Validation	3	0.5203	1.8564		1.7934
AIC	11			0.5523	1.9198

Notes: The results are presented, in rows, for VARs with lag lengths $L = 3$ (selected basing on validation sample performance) and $L = 11$ (selected basing on AIC). Estimation was done by OLS. MSEs (calculated for standardized series, averaged over variables and time periods) in the training, validation and testing as well as combined training & validation sub-samples are given. Predictions are iterated 24-steps-ahead in the validation and testing sub-samples. Additional technical details about the implementation are provided in the main text.

impossible to tell apart an underperformance of the specific NN instance and a peculiarity of the chosen validation data sample.[17]

The testing sample was not available for our preferred NN at the training stage: its predictive fit ended up being 1.6556 there. Many alternative NNs performed better, with the minimum MSE being 1.5958 for a 2-layered NN with 760 top units and 520 bottom ones.

We are going to compare our best NN with the benchmark model, Vector Auto-Regression. Predictive performance on the validation sample suggested using 3 lags, i.e. VAR(3). Such specification comprises 80 parameters in total (refer to equation 6): $(5+5\times5+5\times5+5\times5)$. As can be seen from Table 2, VAR model's in-sample performance is on par with that of our preferred NN: the training sample MSE is 0.5203, and the validation sample MSE is 1.8564. However, VAR underperforms out-of-sample: for the test data, its MSE of 1.7934 is about 8% worse. More rigorously, Diebold and Mariano (1995) test for forecast accuracy with the value of 4.68 favors MRNN at 1% confidence level; modified Diebold and Mariano test that corrects size distortions (due to Harvey et al., 1997) at 4.03 reaches the same conclusion. (We prefer the VAR(3) specification as a benchmark; but in the interest of transparency, Akaike Information Criterion suggested using 11 lags, the resulting VAR(11) model in Table 2 performs poorly out-of-sample, its test-data MSE is 1.9198.)

Let us zoom closer into two models' predictions on Figure 1. Neither of the two captures high-frequency movements, instead they are trying to pin down the movements in conditional means. Failure to achieve the former, much more challenging task should not be surprising: the time series in question are inherently noisy (just think of unpredictable weather conditions), they are affected by many other factors beyond the 5-variable economic system that is being modeled (think of housing market dynamics or global economic

[17]That is why we relied on the—longer—training sample's fit for architecture choice.

environment).

One point to keep in mind is that for most practical purposes, what matters is the cumulative forecast over several periods. In such a situation, much of the high-frequency noise cancels out. Moreover, at a monthly frequency even a small uniform improvement in prediction accuracy can be very meaningful in cumulative sense.[18]

Another point pertains to our standardization convention. We conduct the modeling in terms of standardized time series, each having the variance of 1, and weight them equally in the mean-squared loss function. So, in the original scaling some of the forecasts on Figure 1 may look farther off the target than they are in the standard-deviations space that is important from the modeling perspective. E.g., NN's (as well as VAR's) Fed Funds rate predictions are not as poor as they may seem in the original scaling.

Overall, the take-away is that, at least in our exercise, MRNN outperformed VAR in out-of-sample prediction (and, thus, in generalization).

4.2.2 Steady states

Now we briefly investigate the limiting behavior of the system, which reveals some robust features of the data and the typical scenarios in the evolution of economic variables. In other words, how does the economy look on average, across many time periods?

Technically, a steady state (an equilibrium, or a "fixed point") of a multivariate time-series system is defined as some subset of the space that the system maps back to itself. A stable steady state (attractive fixed point, or a "fixed-point attractor") is a steady state that, even after some small perturbation, the system will again converge to (instead of diverging to somewhere else). Then, a region of convergence ("basin of attraction") is a neighborhood of a stable steady state that maps the outputs from its inputs in the direction toward, and will eventually reach, this particular steady state. The stable steady states characterize the asymptotic behavior of the economic system.

In the case of VAR, such investigation can be done by solving the system analytically (or iterating it forward till convergence). For MRNN, we shall rely on numerical search and simulations. Given that our MRNN is trained to predict no more than 24 steps ahead (and because direct forecasts are known to be relatively noisy in practice), we would need to relax the notion of a steady state.

Formally, we define a quasi steady state of function $\boldsymbol{s} : \mathbb{R}^K \mapsto \mathbb{R}^{K\times H}$ as any value $\boldsymbol{y}^*$ such that $||\bar{\boldsymbol{s}}(\boldsymbol{y}^*) - \boldsymbol{y}^*|| \leq \eta$ for some bound $\eta > 0$ and horizon $H \in \mathbb{N}^+$ with horizon-average of $\boldsymbol{s}(\boldsymbol{y}^*)$ denoted by $\bar{\boldsymbol{s}}(\boldsymbol{y}^*) := (1/H)\boldsymbol{s}(\boldsymbol{y}^*)\boldsymbol{1}_H$; where Euclidean norm $||\boldsymbol{x}|| := (\boldsymbol{x}^\intercal\boldsymbol{x})^{1/2}$ and function $\boldsymbol{s}(\cdot)$ is either an MRNN system (1, 2 and 5) directly predicting H steps ahead (here, $H = 24$), or a VAR model (6) iteratively predicting H steps ahead (e.g., $H \to \infty$). A conventional steady state satisfies the above definition for any $\eta > 0$

[18] Say, an improvement of merely 0.1 percentage point per month produces more than 1 percentage point improvement in the case of GDP growth forecast on a 12-months horizon.

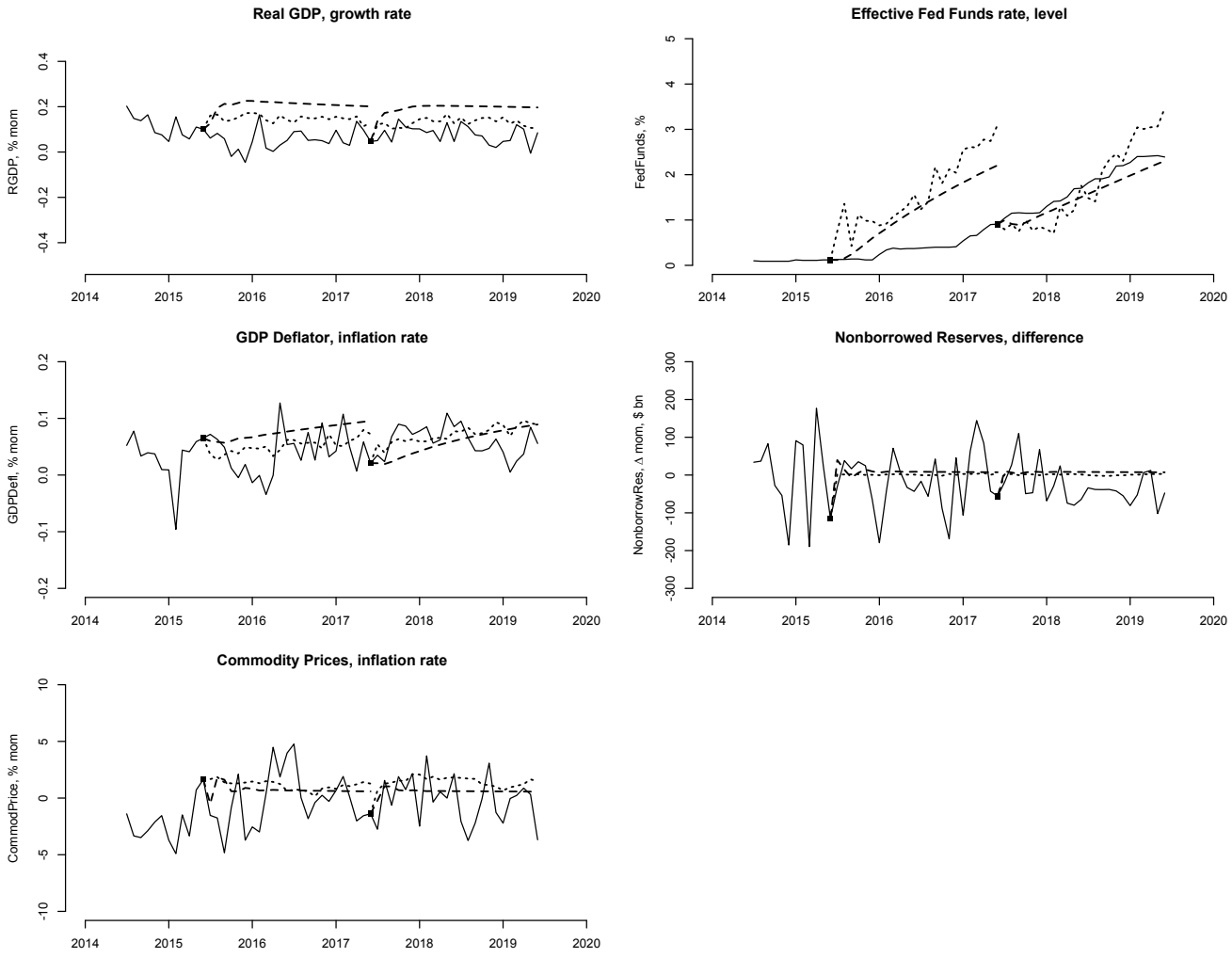

Figure 1: Predictive performance.
Multiple-steps-ahead predictions of the MRNN (dots) and VAR (dashes) models over the testing sample: first, period 2015:M07 to 2017:M06, and second, period 2017:M07 to 2019:M06; with the starting points, 2015:M06 and 2017:M06, marked (by solid squares); together with the 2014:M07 to 2019:M06 sample data (solid lines).

Table 3: Steady states

Model	RGDP	GDPDefl	CommodPrice	FedFunds	NonborrowRes
$\bar{\boldsymbol{y}}$	1.88	1.98	3.62	5.66	31.18
MRNN, low regime	1.44	1.12	7.31	4.18	3.95
MRNN, high regime	1.85	3.63	-0.14	5.84	8.26
VAR	1.87	1.91	3.87	5.35	39.17

Notes: Stable (quasi) steady state values are presented, in rows, for MRNN (low rates of Real GDP growth, GDP Deflator inflation and Fed Funds regime, as well as high rates of Real GDP growth, GDP Deflator inflation and Fed Funds regime); and also for VAR with lag length $L = 3$; additionally on top, $\bar{\boldsymbol{y}}$ provides for reference the sample averages. Variables, in columns, are Real GDP growth, GDP Deflator inflation, Commodity Price inflation (all month-over-month, in percents, annualized); Fed Funds rate (level, in percents, annualized); Nonborrowed Reserves change (monthly difference, in billions of US dollars, annualized).

and all $H \in \mathbb{N}^+$. Since quasi steady state is a relatively weaker notion, a steady state is always a quasi steady state.

Table 3 contains the results of the corresponding calculations. Naturally, VAR steady state is close to the sample mean. However, MRNN has two stable steady states: first, characterized by a combination of high growth, inflation and policy interest rates; and second, with—relatively—low growth, inflation and policy interest rates. Mathematically, existence of several steady states is a common situation in non-linear time-series models, e.g. see Tong (1990). Intuitively, this result reveals that the NN has learned about the presence of two different macroeconomic regimes, and has done so in an unsupervised manner.

The region of convergence for VAR is the whole real hyperplane $\mathbb{R}^K$. For MRNN, the region of convergence is a K-dimensional object that is computationally cumbersome to inspect in detail; but this is a straightforward task for a handful of dimensions conditionally on the rest. For example, take the low growth steady state: freezing all of the variables at their steady state values, we find that an acceleration of Real GDP growth from 1.44% to 2.12% or more sets the system on a trajectory that diverges to another, high growth steady state.

4.2.3 Impulse responses

Time-series models can also be used for measuring system reactions to shocks, as well as to conduct hypothetical policy simulations. For example, if commodity prices (think of oil) jump sharply up, what would happen to other economic variables? Or how would the economy at a particular macroeconomic juncture (think of deep recession or Zero Lower Bound) fare when the Fed Funds rate is raised?

Technically, impulse responses of a multivariate time-series system are defined as movements (responses) of all variables once the equilibrated system receives a shock (impulse) to one of the variables. In this paper, we are interested in single-period (also called transitory) shocks.[19]

Unconditionally: The leading case that we consider is when the system is at its steady state immediately before receiving the shock.

In VAR, the process works as follows. Given the covarying nature of the system variables $\boldsymbol{y}$, and thus of the shocks to these variables $\boldsymbol{\xi}$, a correct way to generate a single shock has to account for the covariance structure. Perhaps the simplest way of doing this is using a Cholesky decomposition of the shocks' variance-covariance matrix: $\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^\intercal$, where $\boldsymbol{L}$ is the resulting lower-triangular matrix. (Technically, this allows to identify contemporaneous causal links between variables in our reduced-form VAR using recursive identification restrictions.) As a result, it holds that $\boldsymbol{\xi} = \boldsymbol{L}\boldsymbol{\varepsilon}$, given some fundamental elementwise-orthogonal shock $\boldsymbol{\varepsilon}$ that has an identity variance-covariance matrix $\boldsymbol{I}$.

Thus, a transitory shock to variable k at impulse time τ is cleanly injected by (i) perturbing the k-th element of $\boldsymbol{\varepsilon}$: $\boldsymbol{\varepsilon}_\tau := \boldsymbol{e}_k$, where $\boldsymbol{e}_k$ is a column vector with, say, 1 in row k and 0 in all other rows; and then (ii) adding the manufactured shock $\boldsymbol{\xi}_\tau := \boldsymbol{L}\boldsymbol{\varepsilon}_\tau$ to the previously equilibrated system: $\boldsymbol{y}_\tau := \boldsymbol{y}^* + \boldsymbol{\xi}_\tau$, where $\boldsymbol{y}^*$ is a steady-state value. Instantaneously in the same period τ, the shock percolates to the variables with non-zero elements of lower triangular matrix $\boldsymbol{L}$; from period $\tau + 1$ onward, the whole system is affected according to $\boldsymbol{y}$'s law of motion. The impulse responses to a single-period shock for horizon $h \in \{1, \dots, H\}$ can be calculated analytically: we can write $\boldsymbol{y}_{\tau+h} = \boldsymbol{y}^* + \boldsymbol{V}_h\boldsymbol{\xi}_\tau$, where $\boldsymbol{y}^* := (\boldsymbol{I}_K - \boldsymbol{A}_1 - \ldots - \boldsymbol{A}_L)^{-1}\boldsymbol{a}$, and $\boldsymbol{V}_i := \sum_{j=1}^{\infty} \boldsymbol{V}_{i-j}\boldsymbol{A}_j$ with $\boldsymbol{V}_0 := \boldsymbol{I}_K$; which in turn gives us

$$\frac{\partial \boldsymbol{y}_{\tau+h}}{\partial \boldsymbol{\varepsilon}_\tau^\intercal} = \boldsymbol{V}_h\boldsymbol{L}.$$[20]

Alternatively, they can be calculated numerically by simulating the evolution of the estimated system (6).[21]

Unfortunately, in LSTM MRNN a similarly straightforward approach as above is not applicable. Formally, utilizing the advantages of direct forecasts, we have $[\boldsymbol{y}_{\tau+1}, \dots, \boldsymbol{y}_{\tau+H}] =$

[19] We do not consider shocks that persist for an infinite number of periods (i.e., permanent shocks). We also ignore potential asymmetry of responses and thus do not examine separately impulses in the opposite directions.

[20] We used the general formula for a moving average representation $\boldsymbol{y}_{\tau+h} = (\boldsymbol{I}_K - \boldsymbol{A}_1 - \ldots - \boldsymbol{A}_L)^{-1}\boldsymbol{a} + \sum_{i=0}^{\infty} \boldsymbol{V}_i\boldsymbol{\xi}_{\tau+h-i}$. It holds provided the operator $\mathbf{A}(\mathrm{B}) := \left(\boldsymbol{I}_K - \boldsymbol{A}_1\mathrm{B} - \ldots - \boldsymbol{A}_L\mathrm{B}^L\right)$ is invertible, where B is a backshift operator.

[21] In the case of VAR, to produce impulse responses we follow the same approach we took for multiple-steps-ahead predictions earlier: iterating 1-step responses for 24 steps ahead (rather than directly calculating 24-steps-ahead responses). Direct estimation of impulse responses in this context is known as local projections, following Jordà (2005). The equivalence of iterative VAR estimates and direct estimates by local projections for these purposes has been shown by Plagborg-Møller and Wolf (2019); also see Marcellino et al. (2006) for a related discussion.

$g^{bW}_{M,\tau} \circ \cdots \circ g^{bW}_{1,\tau}(\boldsymbol{y}_\tau)$, with $\boldsymbol{y}_\tau := \boldsymbol{y}^* + \boldsymbol{\xi}_\tau$ if steady state $\boldsymbol{y}^*$ is taken as a starting point. Then the impulse response is given by expression

$$\frac{\partial \boldsymbol{y}_{\tau+h}}{\partial \boldsymbol{\varepsilon}^\intercal_\tau} = \frac{\partial \hat{\boldsymbol{y}}_{\tau+h}}{\partial g^{bW}_{M,\tau}(\cdot)} \frac{\partial g^{bW}_{M,\tau}(\cdot)}{\partial g^{bW}_{M-1,\tau}(\cdot)} \cdots \frac{\partial g^{bW}_{1,\tau}(\cdot)}{\partial \boldsymbol{y}^\intercal_\tau} \boldsymbol{L},$$

which is a highly non-linear functional that is conditional on the values of state variables. Firstly, although cumbersome to deal with analytically, it is easy to evaluate numerically using the trained NN, since all the necessary ingredients are readily available from back-propagation procedure's implementation at the time of training.[22] Secondly, functions $g^{bW}_{m,\tau}(\cdot)$ are state-dependent: they comprise not only observed variables $\boldsymbol{y}$, but also unobserved state variables $\boldsymbol{c}$ and $\boldsymbol{h}$, and the shocks should perturb both the observed as well as the unobserved variables.[23] However, precise injection of shocks to the state variables is a challenging task given the complexity of a typical LSTM's network of hidden units and connections.

One simple indirect way that works in practice is as follows. The shock $\boldsymbol{\xi}_\tau$ is manufactured using Cholesky decomposition as above. During the usual "warmup session" that is necessary for initialization of the state of the trained LSTM before any usage (and in our case is carried out by feeding the NN with variables' steady-state values and discarding the outputs), the shock $\boldsymbol{\xi} := \boldsymbol{\xi}_\tau$ is added on top of the usual "warmup" inputs, thus shifting the corresponding state variables in the appropriate direction. Then, at impulse time τ, the system receives an input of the starting point with the added shock, $\boldsymbol{y}_\tau := \boldsymbol{y}^* + \boldsymbol{\xi}_\tau$, and predictions for H steps ahead are produced.

Reactions of the VAR and two MRNN models to a 1 percentage point (i.e., 100 basis points) increase in the Fed Funds rate are presented in two forms.[24,25] First, temporal evolution of the responses is shown in Figures 2, 3 and 4.[26] The figures include pointwise confidence bands, obtained by blocks-of-blocks bootstrap (in the terminology of Kilian and Lütkepohl, 2017).[27] Second, the same results are summarized in cumulative annualized terms in Table 4. The following observations can be made:[28]

[22]Explicit modeling of a function together with its derivatives by NNs was studied in, for example, Hornik et al. (1990).

[23]In a non-degenerate LSTM MRNN, the state of the system matters. Therefore, without also shocking the unobserved state variables, the shocked observed variable and the whole system quickly revert to the unconditional mean (in our case, almost immediately).

[24]Note that given the existence of more than one stable steady state in MRNN, the impulse responses offered by MRNN are in certain sense "local", as opposed to the "global" responses provided by VAR.

[25]Before constructing impulse response functions, the LSTM MRNN architecture selected before (in §4.2.1) has been trained for additional 10 epochs (amounting to 20 epochs of training in total).

[26]In the case of MRNN, since direct forecast values are somewhat noisy in practice, in order to avoid contamination with their noise we plot the difference between responses to a positive shock $\boldsymbol{\xi}_\tau := \boldsymbol{L}\boldsymbol{e}_4 = \boldsymbol{L} \times [0; 0; 0; 1; 0]$ and to a zero shock $\boldsymbol{\xi}_\tau := \boldsymbol{L} \times [0; 0; 0; 0; 0]$, rather than the outright response to a positive shock $\boldsymbol{\xi}_\tau := \boldsymbol{L} \times [0; 0; 0; 1; 0]$.

[27]Block size used was 48, structural shocks were defined by Cholesky decomposition of the residual variance-covariance matrix estimated in the actual sample.

[28]We use the wording "short term" and "long term" in the sense of presented impulse response horizons

(i) For the real output, economic intuition suggests contraction/slowdown in response to transitory Fed Funds rate increase (monetary contraction reduces nominal aggregate demand, affecting not only nominal, but also real quantities). VAR implies, after a brief growth acceleration in the short term, a slowdown in the long term. MRNN (for either regime) implies a growth slowdown that starts in the short term and continues into the long term.

(ii) For the output price inflation rate, economic intuition suggests deceleration in response to transitory Fed Funds rate increase (monetary contraction reduces nominal aggregate demand, initiating disinflationary pressure). VAR implies an acceleration of output inflation in the short as well as in the long term. MRNN (in low regime) implies no reaction in the short term, but a long-term inflation deceleration; MRNN (in high regime) implies unchanged inflation in the short as well as in the long term, but demonstrates inflation acceleration in the medium term.

(iii) For commodity prices, economic intuition suggests contraction/deceleration in response to a Fed funds rate increase (tighter monetary supply increases the relative price of currency in terms of goods). VAR implies commodity inflation deceleration that starts in the short term and continues into the long term. MRNN (low regime) implies commodity inflation deceleration both in the short and in the long term. MRNN (high regime) implies a short-term acceleration followed by a long-term deceleration, with the magnitudes that cancel each other out.

(iv) For Fed Funds rate, a stable time-series system should deliver gradual vanishing of its response to the initial impulse (as long as the shock is transitory). VAR implies a gradual mean-reversion of Fed Funds rate after a shock. MRNN (for either regime) also implies a gradual reversion to the respective steady state.

(v) For the nonborrowed reserves that banks hold with the central bank, economic intuition suggests their contraction/slowdown in response to a Fed Funds rate increase (tighter monetary policy increases the opportunity cost of holding reserves). VAR implies a slowdown in reserve accumulation that starts in the short term and continues into the long term. MRNN (for either regime) implies a slowdown over both the short and the long term.

Apart from relatively mild quantitative differences, the key qualitative differences between MRNN and VAR demonstrated in this exercise concern the responses of Real GDP and GDP Price Deflator growth rates. The effect of monetary contraction on output growth is unambiguously negative in MRNN model, in either of its two economic regimes. But the effect is positive before turning negative in VAR model. The effect of monetary

(rather than, say, transitory and permanent effects). For instance, long-term response should not be confused with a new long-run steady state.

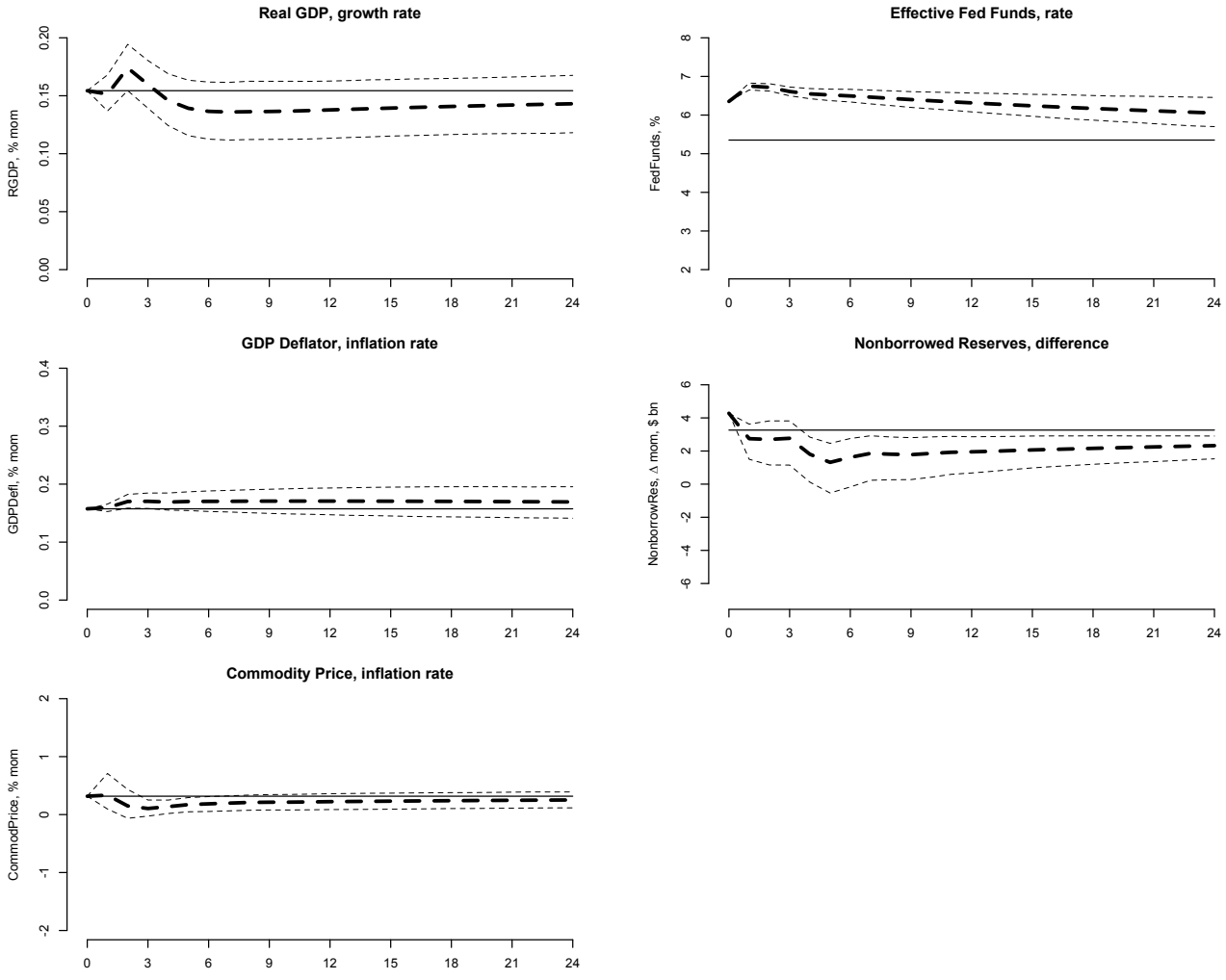

Figure 2: Impulse response functions: VAR.
Reactions in the VAR model (thick dashes) to a shock of +1 p.p. in the Effective Fed Funds rate variable, together with the 70% confidence intervals (thin dashes) and the steady state values (solid lines).
Units on the horizontal axis: months.

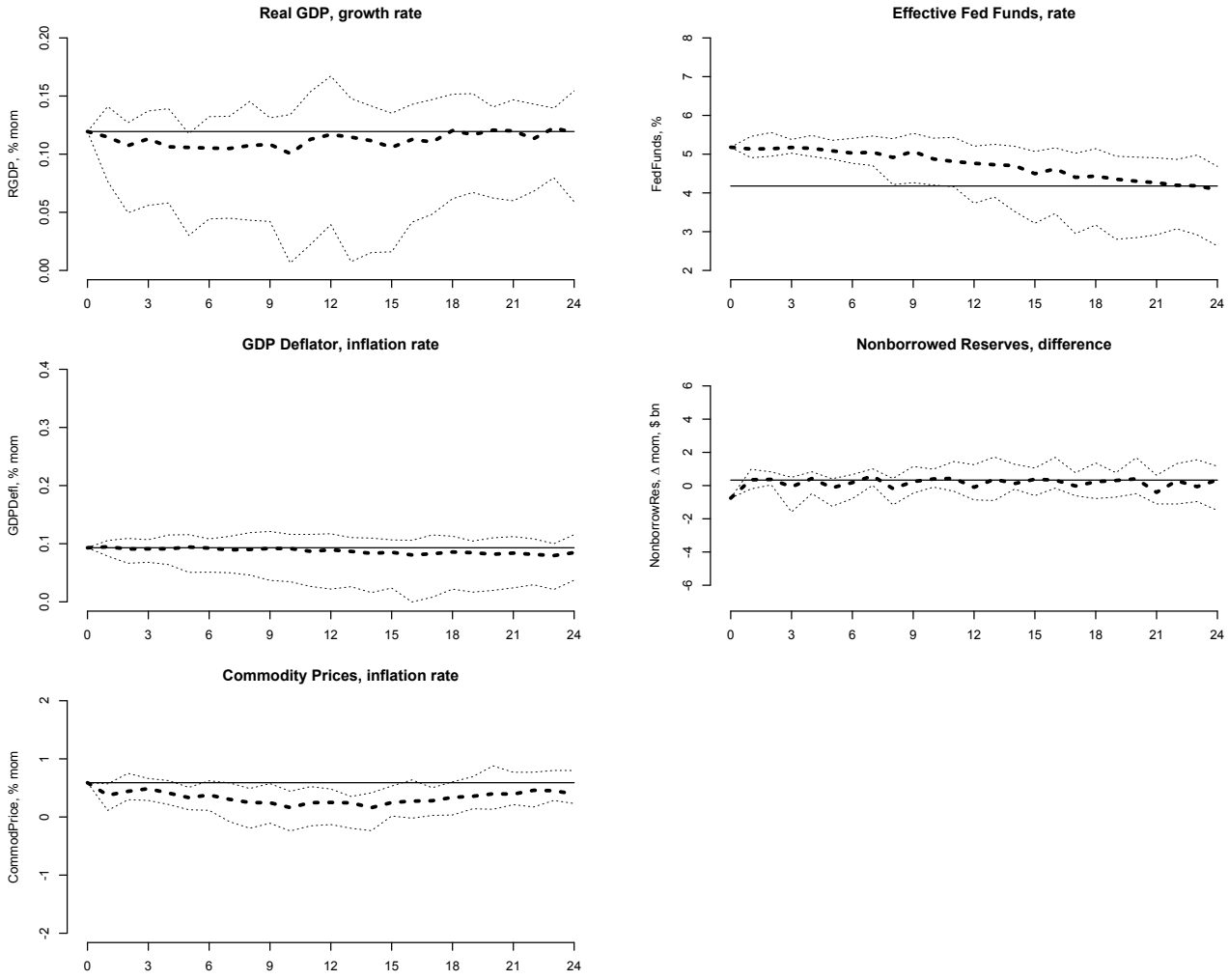

Figure 3: Impulse response functions: MRNN, low regime.
Reactions in the MRNN model (thick dots) to a shock of +1 p.p. in the Effective Fed Funds rate variable, together with the 70% confidence intervals (thin dots) and the low regime's steady state values (solid lines). Units on the horizontal axis: months.

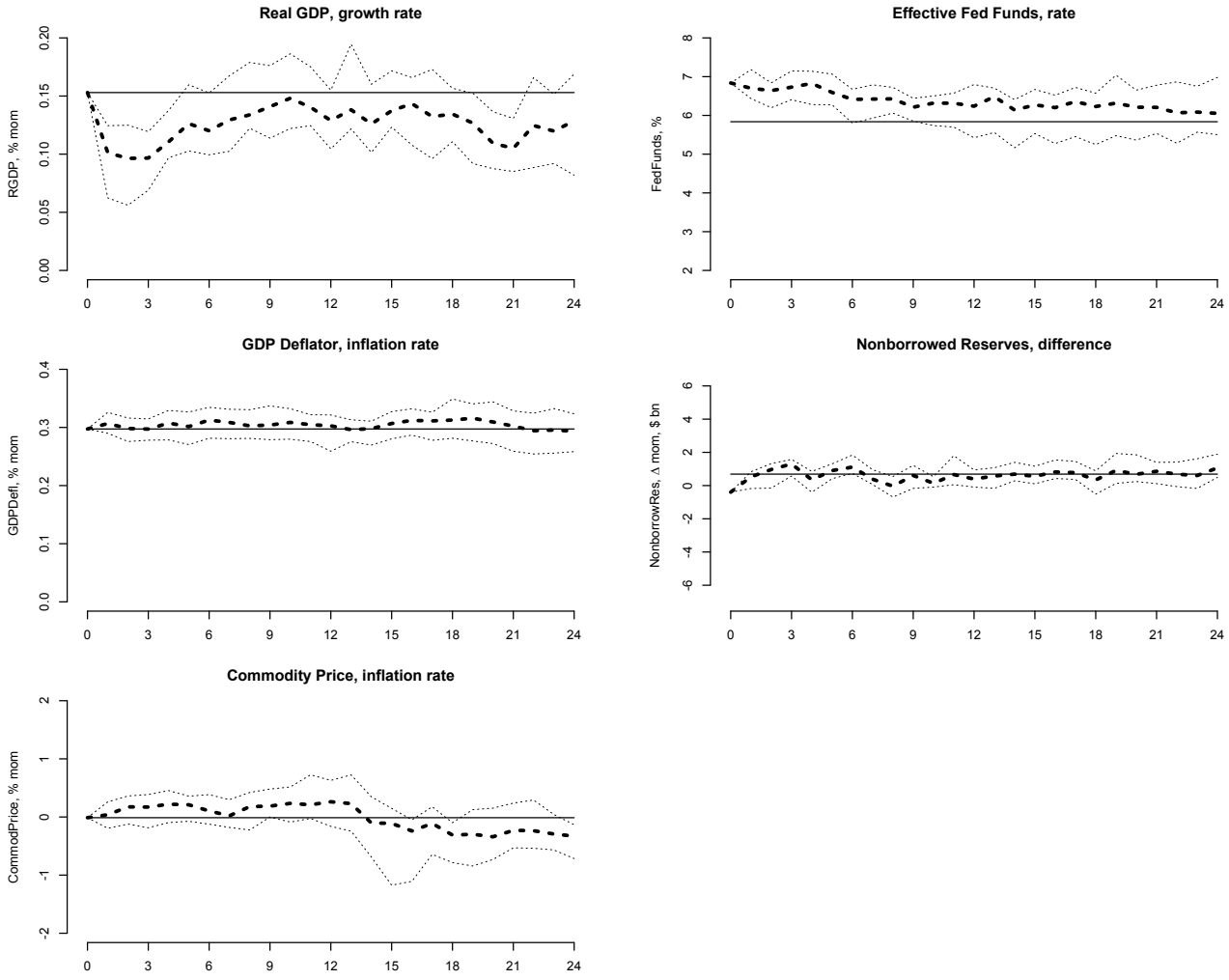

Figure 4: Impulse response functions: MRNN, high regime.
Reactions in the MRNN model (thick dots) to a shock of +1 p.p. in the Effective Fed Funds rate variable, together with the 70% confidence intervals (thin dots) and the high regime's steady state values (solid lines). Units on the horizontal axis: months.

Table 4: Impulse Responses, Annualized averages

Model	RGDP	GDPDefl	CommodPrice	FedFunds	NonborrowRes
MRNN, low regime					
no shocks	1.44	1.12	7.31	4.18	3.95
+FF shock	1.36	1.06	4.14	4.72	1.92
MRNN, high regime					
no shocks	1.85	3.63	-0.14	5.84	8.26
+FF shock	1.52	3.71	-0.18	6.37	7.43
MRNN, at ZLB					
no shocks	0.73	0.77	-3.93	0.10	533.60
+FF shock	0.49	0.83	-4.29	0.61	533.25
VAR					
no shocks	1.87	1.91	3.87	5.35	39.17
+FF shock	1.73	2.05	2.70	6.33	26.13

Notes: Impulse responses to a shock of +1 p.p. in the Effective Fed Funds rate variable, together with baselines of no shocks, are presented, in horizontal panels, for MRNN (low rates of Real GDP growth, GDP Deflator inflation and Fed Funds regime, then high rates of Real GDP growth, GDP Deflator inflation and Fed Funds regime, as well as at Zero Lower Bound); and also for VAR with lag length $L = 3$. Impulse response values are appropriate averages over the 24-month horizon. Variables, in columns, are Real GDP growth, GDP Deflator inflation, Commodity Price inflation (all month-over-month, in percents, annualized); Fed Funds rate (level, in percents, annualized); Nonborrowed Reserves change (monthly difference, in billions of US dollars, annualized).

tightening on inflation is negative in MRNN's regime characterized by low growth, inflation and policy interest rates; though it is positive in MRNN's regime characterized by high growth, inflation and policy interest rates. The effect is unambiguously positive in VAR model. Arguably, the system responses are more plausible from economic theory perspective in the case of MRNN. (A cautionary note applies: our confidence bands are rather large, in particular those growth and inflation responses that contradict economic intuition are not statistically significant.)

Conventional vs. Fisherian views on inflation: The effect of monetary policy shock on output growth is not too dissimilar in the presented models—at least in the long-term—and is roughly in agreement with economic theory. However, its effect on price inflation differs substantially between the models, economic regimes and horizons, with the sign varying from positive to negative; satisfactory theoretical understanding is also lacking in this respect.

Conventional economic theory predicts that transitory Fed Funds rate increase leads to lower inflation (through the nominal aggregate demand channel), even though it may not necessarily materialize in the very short term (e.g., see Christiano et al., 2005).

However, the so-called "price puzzle", implying positive response of inflation to monetary policy rates, has been observed in many studies (first in a VAR model due to Sims, 1992).

Addressing this challenge, in addition to important progress on the side of econometric methodology[29], economic theory resurrected the classical Fisherian arguments (real interest rate is determined by economic fundamentals and can not be easily manipulated, higher nominal interest rate ultimately translates into higher inflation rate). The latter predict that permanent Fed Funds rate increase leads to higher inflation in the long, and possibly even in the short term (Cochrane, 2018; Williamson, 2016).[30]

Nevertheless, as far as inflation response to a transitory monetary policy shock goes, the conventional views are still more prevalent.

Interpreting the two regimes: Now, let us get back to the issue of two stable steady states that the MRNN has identified earlier (in §4.2.2).

Looking at Table 4, the low growth, inflation and policy rates regime seems to capture the trough stage of the business cycle, while the high growth, inflation and policy rates regime corresponds to the peak stage.

Interestingly, in the latter macroeconomic regime that we have associated with business cycle peak, monetary policy tightening results in much stronger growth reduction, but without any benefit of disinflation (at least not at the horizon we consider). Since in the peak regime inflation rate is already relatively high, such a response suggests that respective steady state may also feature problems with anchoring inflation expectations and central bank credibility.

[29] Apart from the effect of some omitted variables, a sensible interpretation of the "price puzzle" finding is that simple, theoretically unrestricted VARs "confuse" correlation with causation. Historically, a high rate of inflation has been associated with high nominal interest rates, and though higher inflation indeed causes a central bank to increase the policy interest rate (one direction of causality), such an increase itself causes inflation rate to subside (another, opposite direction of causality). With the aim of clarifying contemporaneous causal links between variables in VARs, a large literature on structural identification restrictions has been born (leading to so-called structural VARs, which "decompose" reduced-form VARs by applying identification schemes that are mainly based on economic theory as opposed to relatively atheoretical and "mechanistic" schemes such as recursive identification or identification by heteroscedasticity). It includes approaches based on formal theoretical models (see e.g. King et al., 1991; Galí, 1992; Bernanke and Mihov, 1998) as well as those based on more agnostic theoretical assumptions (for example, sign restrictions, as in Uhlig, 2005; Antolín-Díaz and Rubio-Ramírez, 2018; Uribe; 2018).

[30] Although Fisherian arguments are, strictly speaking, not applicable to the case of transitory shocks, their absolute irrelevance hinges on economic agents' accurate real-time distinction between the two shocks and time discounting considerations.

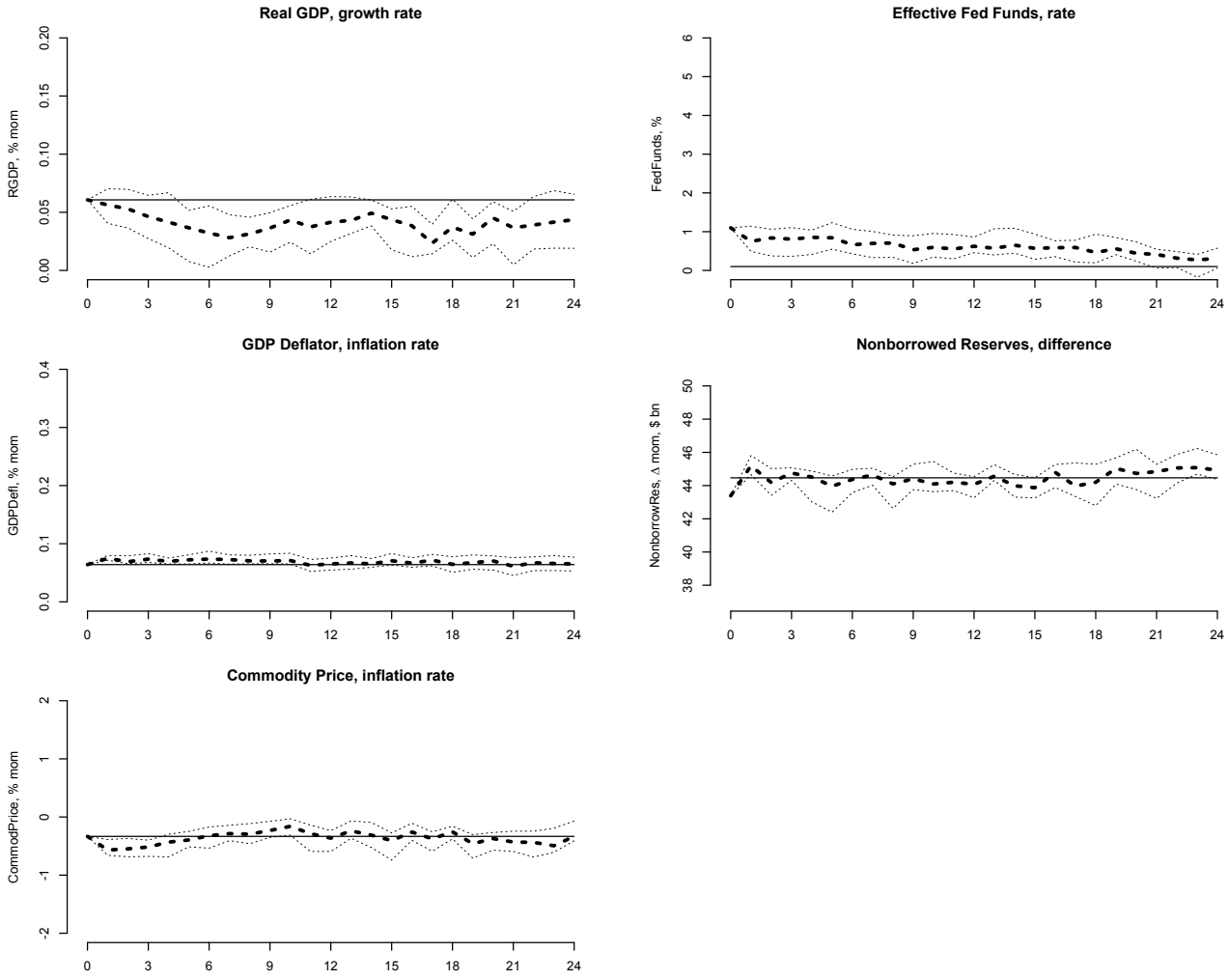

Figure 5: Impulse response functions: MRNN, at Zero Lower Bound.
Reactions in the MRNN model (thick dots) to a shock of +1 p.p. in the Effective Fed Funds rate variable, together with the 70% confidence intervals (thin dots) and the 2011:M01 to 2012:M01 sample means (solid lines).
Units on the horizontal axis: months.

Conditionally (Zero Lower Bound): Another case that we examine is the impulse responses when the system is placed into specific economic environment, with the situation around Zero Lower Bound being particularly interesting. In contrast to VAR, LSTM MRNN may be especially useful in this case, given the latter's ability to capture state-dependent dynamics.

We pick January 2011 to January 2012 as the time interval when the Federal Reserve's main policy instrument, Fed Funds rate, was at historically low levels, and was constrained from below by the interest rate of zero that promises no surplus return to the lender of funds (i.e., was at Zero Lower Bound). MRNN impulse responses are computed the same way as before, but now we initialize the model by historical data (January 2009 to December 2011) rather than by the steady state values, and trace the multiple-steps-ahead effect of a hypothetical Fed Funds rate shock added to actual historical data immediately afterwards (January 2012). The results are presented in Figure 5 as well as Table 4.

Following the Fed Funds rate increase, real output growth decelerates in the short as well as in the long term, while output price inflation rate accelerates in the short term with no effect in the long term. Commodity price inflation turns more negative in the short and in the long term. Fed Funds rate gradually reverts to its unshocked path. Nonborrowed reserve accumulation barely changes and stays at historically high levels.

Comparing these responses to those in the low growth, inflation and policy rates regime, we note three main differences. Monetary policy tightening around ZLB results in (i) relatively larger output slowdown, (ii) inflation acceleration instead of deceleration (in particular, with deflation risks not materializing), as well as (iii) commodity prices that are already falling accelerating their decline even more.

5 Conclusion

In modeling half a century of US economic time-series data, Multivariate Recurrent Neural Networks outperformed mainstream Vector Auto-Regressions in terms of forecasting accuracy and interpretability of their results. Higher precision of its out-of-sample predictions suggests that MRNN acquires stronger generalization ability. Also, MRNN discovers different macroeconomic regimes under a "hands-off" procedure, without any special modeler's intervention. Moreover, MRNN appears capable of learning theoretically plausible causal effects from the raw data, without the need to impose sophisticated structural identification restrictions beyond a simple recursive ordering of variables. Arguably, nonlinear architecture of NNs and the prominence of predictive criterion in their training, as well as Long Short-Term Memory MRNN's capability to account for conditional dynamics of the modeled system's state variables make them well equipped for such tasks.

A Data inputs

List of data inputs including series names and details, units, seasonal adjustment, frequency; sources (Federal Reserve Bank of St. Louis Economic Data / Datastream codes):

1.0. Real Gross Domestic Product, Billions of Chained 2012 Dollars, Seasonally Adjusted Annual Rate, Quarterly; U.S. Bureau of Economic Analysis (GDPC1).

1.1. Industrial Production Index, Index 2012=100, Seasonally Adjusted, Monthly; Board of Governors of the Federal Reserve System (INDPRO).

1.2. Employment Level, Thousands of Persons, Seasonally Adjusted, Monthly; U.S. Bureau of Labor Statistics (CE16OV).

1.3. Real Disposable Personal Income, Billions of Chained 2012 Dollars, Seasonally Adjusted Annual Rate, Monthly; U.S. Bureau of Economic Analysis (DSPIC96).

2.0. Gross Domestic Product: Implicit Price Deflator, Index 2012=100, Seasonally Adjusted, Quarterly; U.S. Bureau of Economic Analysis (GDPDEF).

2.1. Producer Price Index for All Commodities, Index 1982=100, Not Seasonally Adjusted, Monthly; U.S. Bureau of Labor Statistics (PPIACO).

2.2. Consumer Price Index for All Urban Consumers: All Items in U.S. City Average, Index 1982-1984=100, Seasonally Adjusted, Monthly; U.S. Bureau of Labor Statistics (CPIAUCSL).

2.3. Personal Consumption Expenditures: Chain-type Price Index, Index 2012=100, Seasonally Adjusted, Monthly; U.S. Bureau of Economic Analysis (PCEPI).

3.0. Thomson Reuters Equal Weight Continuous Commodity Index, Price Index, Not Seasonally Adjusted, Daily; Refinitiv (NYFECRB).

4.0. Effective Federal Funds Rate, Percent, Not Seasonally Adjusted, Monthly; Board of Governors of the Federal Reserve System (FEDFUNDS).

5.0. Nonborrowed Reserves of Depository Institutions, Millions of Dollars, Not Seasonally Adjusted, Monthly; Board of Governors of the Federal Reserve System (NONBORRES).

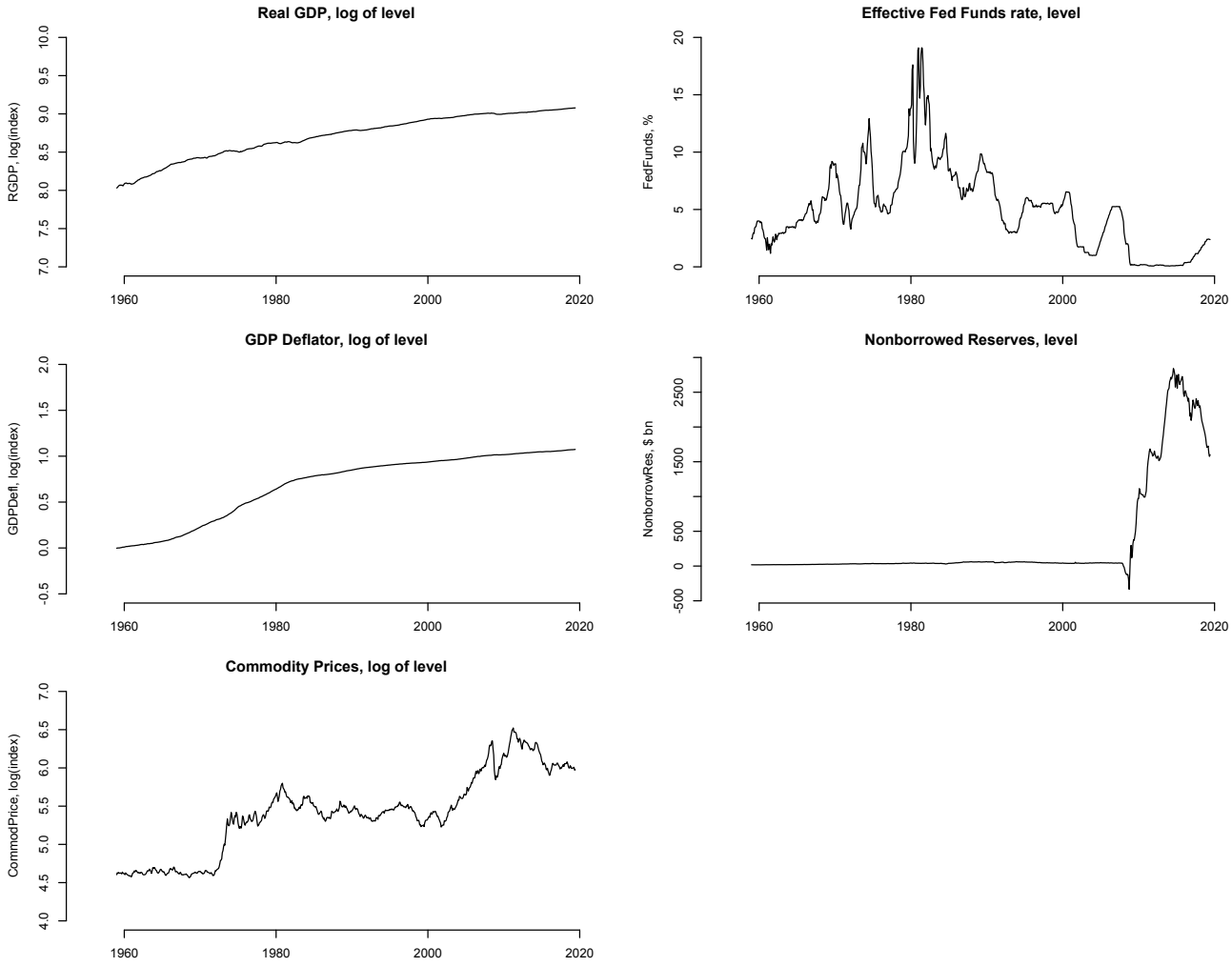

Figure 6: Plots of the original data inputs.
Variables included: logarithm of Real GDP, logarithm of GDP Price Deflator, logarithm of Commodity Price Index, Effective Fed Funds Rate and Nonborrowed Reserves.
Time period covered: 1959:M01 to 2019:M06.

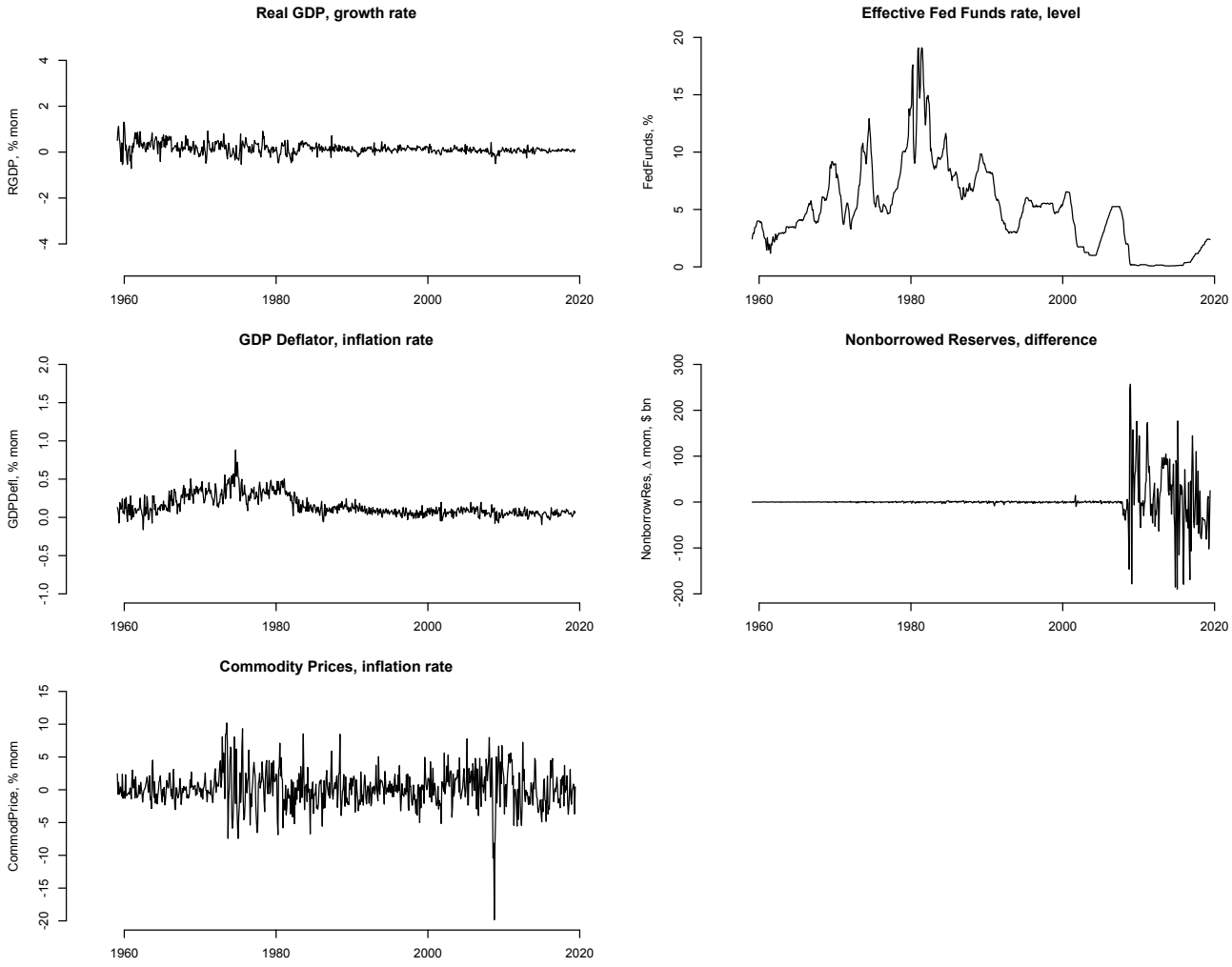

Figure 7: Plots of the transformed data inputs.
Variables included: Real GDP MoM growth rate, GDP Price Deflator MoM growth rate, Commodity Price Index MoM growth rate, Effective Fed Funds Rate and Nonborrowed Reserves monthly difference.
Time period covered: 1959:M02 to 2019:M06.

B Data interpolation: quarterly into monthly

Real GDP is observed at quarterly frequency, and the interpolation is based on the information contained in Industrial Production Index, Employment level and Real Disposable Personal Income that are observed monthly. Similarly, GDP Deflator is observed quarterly, and is interpolated using Producer Price Index, Consumer Price Index and Personal Consumption Expenditures Price Index that are observed monthly.

Denoting the interpolated quarterly variables as y_t, and the interpolating monthly variables as $x_{i,t}$, $i = 1, 2, 3$, the state-space system is formulated as

$$\begin{aligned} y_t &= \beta_0 + \beta_1 x_{1,t} + \beta_2 x_{2,t} + \beta_3 x_{3,t} + u_t, \\ u_t &= \rho u_{t-1} + \epsilon_t, \end{aligned}$$

where the measurement equation contains variable y_t that is observed only at the last months' of each quarter, while variables $x_{i,t}$ are observed for every month; the state equation contains variable ϵ_t that is unobserved and is assumed to be distributed as $\epsilon_t \sim \mathcal{N}(0, \sigma_\epsilon^2)$; with β_0, β_1, β_2, β_3, ρ and σ_ϵ^2 being constant parameters.

After standardization, or normalization, defined as $z_t := 1 + \ln(z_t/z_3)$, for $z \in \{y, x\}$ and where $t = 1, 2, \ldots, 726$ (i.e., from 1959:M01 to 2019:M06), we estimate two interpolating models for the levels of Real GDP and GDP Deflator independently by maximum likelihood using the Kalman–Bucy filter (analogously to Bernanke et al., 1997).

C Data augmentation

As a robustness check, in particular to address possible concerns about relying on a very limited data set, we attempt a data augmentation exercise.

Since in our modeling approach we abstain from imposing any structure on our data, the augmentation strategy we chose is noise contamination. Specifically, we (i) divide the existing (standardized) time series into blocks of 50 consecutive observations; (ii) copy the data set 10 times; (iii) randomly shuffle the blocks in the enlarged data set; (iv) add Gaussian noise (with zero mean and a variance-covariance observed in the training sample) scaled by 0.1; (v) additionally paste the last block of the (uncontaminated) training data set at the end of the time series. Thus, we extend the training data set tenfold, leaving the validation and test samples untouched.

Keeping the MRNN architecture chosen in the main text, on augmented data set the model was trained exactly the same way as before. When we applied the trained net to the uncontaminated data set, the measures of fit on all three sub-samples underperformed those of our original MRNN.

Our data augmentation approach did not seem to offer any advantage over using just the existing data as is. However, a different approach may prove to be more fruitful, hence verification of different augmentation settings or perhaps even trying a more structural augmentation strategy is still a worthwhile exercise.

References

[1] Akaike, Hirotugu. (1974) "A New Look at the Statistical Model Identification", *IEEE Transactions on Automatic Control*, 19(6): 716–723.

[2] Antolín-Díaz, Juan, and Juan F. Rubio-Ramírez. (2018) "Narrative Sign Restrictions for SVARs", *American Economic Review*, 108(10): 2802–2829.

[3] Arnold, V. I. (1957) "On Functions of Three Variables", *Proceedings of the USSR Academy of Sciences*, 114: 679–681.

[4] Arora, Sanjeev, Rong Ge, Behnam Neyshabur, Yi Zhang. (2018) "Stronger Generalization Bounds for Deep Nets via a Compression Approach", arXiv:1802.05296.

[5] Bauer, Benedikt, and Michael Kohler. (2019) "On Deep Learning as a Remedy for the Curse of Dimensionality in Nonparametric Regression", *Annals of Statistics*, 47(4): 2261–2285.

[6] Belkin, Mikhail, Siyuan Ma, Soumik Mandal. (2018) "To Understand Deep Learning We Need to Understand Kernel Learning", arXiv:1802.01396.

[7] Bengio, Yoshua, Nicolas Boulanger-Lewandowski and Razvan Pascanu. (2012) "Advances in Optimizing Recurrent Networks", arXiv:1212.0901.

[8] Bernanke, Ben S., Mark Gertler, and Mark Watson. (1997) "Systematic Monetary Policy and the Effiects of Oil Price Shocks", *Brookings Papers on Economic Activity*, 1: 91–157.

[9] Bernanke, Ben S., and Ilian Mihov. (1998) "Measuring Monetary Policy", *Quarterly Journal of Economics*, 113(3): 869–902.

[10] Bishop, Christopher M. (2006) *Pattern Recognition and Machine Learning*, Springer.

[11] Box, George, Gwilym Jenkins. (1970) *Time Series Analysis: Forecasting and Control*, San Francisco: Holden-Day.

[12] Cagli, Eleonora, Cécile Dumas, and Emmanuel Prouff. (2017) "Convolutional Neural Networks with Data Augmentation Against Jitter-based Countermeasures", Proceedings of the 19th International Conference Cryptographic Hardware and Embedded Systems, 45–68.

[13] Chen, An Mei, Haw-minn Lu and Robert Hecht-Nielsen. (1993) "On the Geometry of Feedforward Neural Network Error Surfaces", *Neural Computation*, 5(6): 910–927.

[14] Chen, Xiaohong. (2007) "Large Sample Sieve Estimation of Semi-Nonparametric Models". In: James J. Heckman and Edward E. Leamer (eds.), *Handbook of Econometrics,*, edition 1, volume 6B, chapter 76, pages 5549–5632, Elsevier.

[15] Chen, Xiaohong, Jeffrey Racine, and Norman R. Swanson. (2001) "Semiparametric ARX Neural-Network Models with an Application to Forecasting Inflation", *IEEE Transactions on Neural Networks*, 12(4): 674–683.

[16] Chen, Xiaohong, and Halbert White. (1999) "Improved Rates and Asymptotic Normality for Nonparametric Neural Network Estimators", *IEEE Transactions on Information Theory*, 45(2): 682–691.

[17] Christiano, Lawrence J., Martin Eichenbaum, and Charles L. Evans. (2005) "Nominal Rigidities and the Dynamic Effects of a Shock to Monetary Policy", *Journal of Political Economy*, 113(1): 1–45.

[18] Cochrane, John H. (2018) "Michelson-Morley, Fisher, and Occam: The Radical Implications of Stable Quiet Inflation at the Zero Bound", *NBER Macroeconomics Annual*, 32(1): 113–226.

[19] Cui, Xiaodong, Vaibhava Goel and Brian Kingsbury. (2015) "Data Augmentation for Deep Neural Network Acoustic Modeling", *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(9): 1469–1477.

[20] Cybenko, George. (1989) "Approximation by Superpositions of a Sigmoidal Function", *Mathematics of Control, Signals, and Systems*, 2: 303–314.

[21] Diebold, Francis X., and Roberto S. Mariano. (1995) "Comparing Predictive Accuracy", *Journal of Business and Economic Statistics*, 13: 253–263.

[22] Dixon, Matthew F., Nicholas G. Polson, Vadim O. Sokolov. (2018) "Deep Learning for Spatio-Temporal Modeling: Dynamic Traffic Flows and High Frequency Trading", arXiv:1705.09851.

[23] Dütting, Paul, Zhe Feng, Harikrishna Narasimhan, David C. Parkes. (2017) "Optimal Auctions through Deep Learning", arXiv:1706.03459.

[24] Elman, Jeffrey L. (1990) "Finding Structure in Time", *Cognitive Science* 14(2): 179-211.

[25] Farrell, Max H., Tengyuan Liang, Sanjog Misra. (2019) "Deep Neural Networks for Estimation and Inference", arXiv:1809.09953.

[26] Feng, Guanhao, Jingyu He, Nicholas G. Polson. (2018) "Deep Learning for Predicting Asset Returns", arXiv:1804.09314.

[27] Friston, Karl J., Richard Rosch, Thomas Parr, Cathy Price, Howard Bowman (2018) "Deep Temporal Models and Active Inference", *Neuroscience and Biobehavioral Reviews*, 90: 486–501.

[28] Galí, Jordi. (1992) "How Well Does the IS-LM Model Fit Postwar U.S. Data?", *Quarterly Journal of Economics*, 107(2): 709–738.

[29] Goel, Hardik, Igor Melnyk, Nikunj Oza, Bryan Matthews, Arindam Banerjee. (2016) "Multivariate Aviation Time Series Modeling: VARs vs. LSTMs", Manuscript.

[30] Goldberg, Yoav. (2016) "A Primer on Neural Network Models for Natural Language Processing", *Journal of Arti*

cial Intelligence Research, 57: 345–420.

[31] Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. (2016) *Deep Learning*, MIT Press.

[32] Graves, Alex, and Jürgen Schmidhuber. (2009) "Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks", *Advances in Neural Information Processing Systems*, 21.

[33] Gu, Shihao, Bryan Kelly and Dacheng Xiu. (2018) "Empirical Asset Pricing via Machine Learning", Manuscript.

[34] Hamilton, James D. (1994) *Time Series Analysis*, Princeton: Princeton University Press.

[35] Hartford, Jason, Greg Lewis, Kevin Leyton-Brown, Matt Taddy. (2016) "Counterfactual Prediction with Deep Instrumental Variables Networks", arXiv:1612.09596.

[36] Harvey, David, Stephen Leybourne, and Paul Newbold. (1997) "Testing the Equality of Prediction Mean Squared Errors", *International Journal of Forecasting*, 13(2): 281–291.

[37] He, Yang-Hui. (2017) "Machine-learning the String Landscape", *Physics Letters B*, 774, pp. 564–568.

[38] Heaton, J. B., N. G. Polson, J. H. Witte. (2016a) "Deep Learning in Finance", arXiv:1602.06561.

[39] Heaton, J. B., N. G. Polson, J. H. Witte. (2016b) "Deep Portfolio Theory", arXiv:1605.07230.

[40] Hinton, Geoffrey E., and Drew Van Camp. (1993) "Keeping Neural Networks Simple by Minimizing the Description Length of the Weights", Proceedings of the sixth annual conference on Computational learning theory: 5–13.

[41] Hochreiter, Sepp, and Jürgen Schmidhuber. (1997a) "Flat Minima", *Neural Computation*, 9(1): 1–42.

[42] Hochreiter, Sepp, and Jürgen Schmidhuber. (1997b) "Long Short-Term Memory", *Neural Computation* 9(8): 1735–1780.

[43] Hornik, Kurt, Maxwell Stinchcombe and Halber White. (1989) "Multilayer Feedforward Networks are Universal Approximators", *Neural Networks*, 2: 359–366.

[44] Hornik, Kurt, Maxwell Stinchcombe and Halber White. (1990) "Universal Approximation of an Unknown Mapping and Its Derivatives Using Multilayer Feedforward Networks", *Neural Networks*, 3: 551–560.

[45] Hutchinson, James M., Andrew W. Lo and Tomaso Poggio. (1994) "A Nonparametric Approach to Pricing and Hedging Derivative Securities Via Learning Networks", *Journal of Finance*, 49(3): 851–889.

[46] Jaeger, Herbert, and Harald Haas. (2004) "Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication", *Science*, 304: 708–80.

[47] Jaitly, Navdeep, and Geoffrey E. Hinton. (2013) "Vocal Tract Length Perturbation (VTLP) Improves Speech Recognition", Proceedings of the 30 th International Conference on Machine Learning.

[48] Jordà, Òscar. (2005) "Estimation and Inference of Impulse Responses by Local Projections", *American Economic Review*, 95(1): 161–182.

[49] Ioffe, Sergey, Christian Szegedy. (2015) "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", arXiv:1502.03167.

[50] Ivakhnenko, A. G., and V. G. Lapa. (1965) *Cybernetic Predicting Devices*, CCM Information Corporation.

[51] Kilian, Lutz, and Helmut Lütkepohl. (2017) *Structural Vector Autoregressive Analysis*, Cambridge University Press.

[52] King, Robert G., Charles I. Plosser, James H. Stock and Mark W. Watson. (1991) "Stochastic Trends and Economic Fluctuations", *American Economic Review*, 81(4): 819–840.

[53] Kingma, Diederik P., and Jimmy Ba. (2014) "Adam: A Method for Stochastic Optimization", arXiv:1412.6980.

[54] Ko, Tom, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. (2015) "Audio Augmentation for Speech Recognition", INTERSPEECH: 3586–3589.

[55] Kolmogorov, A. N. (1957) "On the Representation of Continuous Functions of Many Variables by Superposition of Continuous Functions of One Variable and Addition", *Proceedings of the USSR Academy of Sciences*, 114, 953–956.

[56] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. (2012) "Imagenet Classification with Deep Convolutional Neural Networks", *Advances in Neural Information Processing Systems*, 25: 1097-1105.

[57] Kuan, Chung-Ming, Kurt Hornik, and Halbert White. (1994) "A Convergence Result for Learning in Recurrent Neural Networks", *Neural Computation*, 6: 420–440.

[58] Kurková, Věra, and Paul C. Kainen. (1994) "Functionally Equivalent Feedforward Neural Networks", *Neural Computation*, 6(3): 543–558.

[59] LeCun, Yann, Léon Bottou, Yoshua Bengio and Patrick Haffner. (1998) "Gradient-Based Learning Applied to Document Recognition", *Proceedings of the IEEE*, 86(11), 2278–2324.

[60] Lei, Na, Zhongxuan Luo, Shing-Tung Yau, David Xianfeng Gu. (2018) "Geometric Understanding of Deep Learning", arXiv:1805.10451.

[61] Liao, Qianli, and Tomaso Poggio. (2016) "Bridging the Gaps Between Residual Learning, Recurrent Neural Networks and Visual Cortex", arXiv:1604.03640.

[62] Lipton, Zachary C., John Berkowitz, Charles Elkan. (2015) "A Critical Review of Recurrent Neural Networks for Sequence Learning", arXiv:1506.00019.

[63] Maghrebi, Houssem, Thibault Portigliatti, Emmanuel Prouff. (2016) "Breaking Cryptographic Implementations Using Deep Learning Techniques", Proceedings of the 6th International Conference Security, Privacy, and Applied Cryptography Engineering, 3–26.

[64] Mandt, Stephan, Matthew D. Hoffman, David M. Blei. (2017) "Stochastic Gradient Descent as Approximate Bayesian Inference", arXiv:1704.04289.

[65] Marcellino, Massimiliano, James H. Stock and Mark W. Watson. (2006) "A Comparison of Direct and Iterated Multistep AR Methods for Forecasting Macroeconomic Time Series", *Journal of Econometrics*, 135, 499–526.

[66] McCulloch Warren, S., and Walter Pitts. (1943) "A Logical Calculus of the Ideas Immanent in Nervous Activity", *Bulletin of Mathematical Biophysics*, 5: 115–133.

[67] Mehta, Pankaj, David J. Schwab. (2014) "An Exact Mapping Between the Variational Renormalization Group and Deep Learning", arXiv:1410.3831.

[68] Patel, Ankit B., Tan Nguyen, Richard G. Baraniuk. (2015)"A Probabilistic Theory of Deep Learning", arXiv:1504.00641.

[69] Pathak, Jaideep, Alexander Wikner, Rebeckah Fussell, Sarthak Chandra, Brian R. Hunt, Michelle Girvan, and Edward Ott. (2018) "Hybrid Forecasting of Chaotic Processes: Using Machine Learning in Conjunction with a Knowledge-based Model", *Chaos*, 28: 041101.

[70] Plagborg-Møller, Mikkel, and Christian K. Wolf. (2019) "Local Projections and VARs Estimate the Same Impulse Responses", Manuscript.

[71] Polson, Nicholas G., and Vadim O. Sokolov. (2017) "Deep Learning: A Bayesian Perspective", arXiv:1706.00473.

[72] Prechelt, Lutz. (1998) "Early Stopping — But When?" *Neural Networks: Tricks Of The Trade*, 1524: 55–69.

[73] Ramey, Valerie A. (2016) "Macroeconomic Shocks and Their Propagation". In: John B. Taylor and Harald Uhlig (eds.), *Handbook of Macroeconomics*, edition 1, volume 2A, chapter 2, pages 71–162, Elsevier.

[74] Robbins, H. and Monro, S. (1951) "A Stochastic Approximation Method", *Annals of Mathematical Statistics*, 22(3): 400–407.

[75] Rosenblatt, Frank. (1958) "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain", *Psychological Review*, 65(6): 386–408.

[76] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. (1986) "Learning Representations by Back-Propagating Errors", *Nature*, 323(6088): 533–536.

[77] Schmidhuber, Jürgen. (2015) "Deep Learning in Neural Networks: An Overview", *Neural Networks*, 61: 85–117.

[78] Schmidt-Hieber, Johannes. (forthcoming) "Nonparametric Regression Using Deep Neural Networks with ReLU Activation Function", *Annals of Statistics*.

[79] Schwartz-Ziv, Ravid, Naftali Tishby. (2017) "Opening the black box of Deep Neural Networks via Information", arXiv:1703.00810.

[80] Sietsma, Jocelyn, and Robert J. F. Dow. (1991) "Creating Artificial Neural Networks That Generalize", *Neural Networks*, 4: 67–79.

[81] Simard, Patrice Y., Dave Steinkraus, John C. Platt. (2003) "Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis", Proceedings of the Seventh International Conference on Document Analysis and Recognition.

[82] Sims, Christopher A. (1980) "Macroeconomics and Reality", *Econometrica*, 48(1): 1–48.

[83] Sims, Christopher A. (1992) "Interpreting the Macroeconomic Time Series Facts: The Effects of Monetary Policy", *European Economic Review*, 36: 975–1000.

[84] Sirignano, Justin A. (2016) "Deep Learning for Limit Order Books", arXiv:1601.01987.

[85] Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov. (2014) "Dropout: A Simple Way to Prevent Neural Networks from Overfitting", *Journal of Machine Learning Research*, 15: 1929–1958.

[86] Stock, James H., Mark W. Watson. (2016) "Dynamic Factor Models, Factor-Augmented Vector Autoregressions, and Structural Vector Autoregressions in Macroeconomics". In: John B. Taylor and Harald Uhlig (eds.), *Handbook of Macroeconomics*, edition 1, volume 2A, chapter 8, pages 415–525, Elsevier.

[87] Stock, James H., and Mark W. Watson. (2017) "Twenty Years of Time Series Econometrics in Ten Pictures", *Journal of Economic Perspectives*, 31(2): 59–86.

[88] Sutskever, Ilya, James Martens, George Dahl, Geoffrey Hinton. (2013) "On the Importance of Initialization and Momentum in Deep Learning", Proceedings of the 30th International Conference on Machine Learning,

[89] Swanson, Norman R., and Halber White. (1995) "A Model-Selection Approach to Assessing the Information in the Term Structure Using Linear Models and Artificial Neural Networks", *Journal of Business and Economic Statistics*, 13(3): 265–275.

[90] Swanson, Norman R., and Halber White. (1997) "A Model Selection Approach to Real-Time Macroeconomic Forecasting Using Linear Models and Artificial Neural Networks", *Review of Economics and Statistics*, 79(4): 540–550.

[91] Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich. (2014) "Going Deeper with Convolutions", arXiv:1409.4842.

[92] Theil, Henri. (1954) *Linear Aggregation of Economic Relations*, Amsterdam: North-Holland Publishing Company.

[93] Tong, Howell. (1990) *Non-linear Time Series: A Dynamical System Approach*, Oxford: Clarendon Press; New York: Oxford University Press.

[94] Uhlig, Harald. (2005)"What Are the Effects of Monetary Policy on Output? Results from an Agnostic Identification Procedure", *Journal of Monetary Economics*, 52: 381–419.

[95] Uribe, Martín. (2018) "The Neo-Fisher Effect: Econometric Evidence from Empirical and Optimizing Models", Manuscript.

[96] Vlachas, Pantelis R., Wonmin Byeon, Zhong Y.Wan, Themistoklis P. Sapsis and Petros Koumoutsakos. (2018) "Data-driven Forecasting of High-dimensional Chaotic Systems with Long Short-term Memory Networks", *Proceedings of the Royal Society A*, 474: 20170844.

[97] Wager, Stefan, Sida Wang, and Percy S Liang. (2013) "Dropout Training as Adaptive Regularization". In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, 26: 351–359.

[98] Welling, Max, and Yee Whye Teh. (2011) "Bayesian Learning via Stochastic Gradient Langevin Dynamics", Proceedings of the 28th International Conference on Machine Learning.

[99] White, Halbert. (1989) "Learning in Artificial Neural Networks: A Statistical Perspective", *Neural Computation*, 1: 425–464.

[100] White, Halbert and Jeffrey Racine. (2001) "Statistical Inference, The Bootstrap, and Neural-Network Modeling with Application to Foreign Exchange Rates", *IEEE Transactions on Neural Networks*, 12(4): 657–673.

[101] Williamson, Stephen. (2016) "Neo-Fisherism: A Radical Idea, or the Most Obvious Solution to the Low-Inflation Problem?", Federal Reserve Bank of St. Louis Regional Economist, July: 5–9.

[102] Wilson, D. Randall, and Tony R. Martinez. (2003) "The General Inefficiency of Batch Training for Gradient Descent Learning", *Neural Networks*, 16: 1429–1451.

[103] Zaremba, Wojciech, Ilya Sutskever, Oriol Vinyals. (2015) "Recurrent Neural Nerwork Regularization", arXiv:1409.2329.

[104] Zellner, Arnold. (1962) "An Efficient Method of Estimating Seemingly Unrelated Regressions and Tests for Aggregation Bias", *Journal of the American Statistical Association*, 57(298): 348–368.

[105] Zellner, Arnold, and Henri Theil. (1962) "Three-Stage Least Squares: Simultaneous Estimation of Simultaneous Equations", *Econometrica*, 30: 54–78.

[106] Zhang, Chiyuan, Samy Bengio, Moritz Hardt, Benjamin Recht, Oriol Vinyals. (2017) "Understanding Deep Learning Requires Rethinking Generalization", arXiv:1611.03530.