

# Enhanced Bayesian Neural Networks for Macroeconomics and Finance<sup>\*</sup>

NIKO HAUZENBERGER<sup>1, 2</sup>, FLORIAN HUBER<sup>1</sup>, KARIN KLIEBER<sup>3</sup>,  
and MASSIMILIANO MARCELLINO<sup>4</sup>

<sup>1</sup>*University of Salzburg*

<sup>2</sup>*Vienna University of Economics and Business*

<sup>3</sup>*Oesterreichische Nationalbank*

<sup>4</sup>*Bocconi University, IGER, CEPR, Baffi-Carefin and BIDS*

November 7, 2022

We develop Bayesian neural networks (BNNs) that permit to model generic nonlinearities and time variation for (possibly large sets of) macroeconomic and financial variables. From a methodological point of view, we allow for a general specification of networks that can be applied to either dense or sparse datasets, and combines various activation functions, a possibly very large number of neurons, and stochastic volatility (SV) for the error term. From a computational point of view, we develop fast and efficient estimation algorithms for the general BNNs we introduce. From an empirical point of view, we show both with simulated data and with a set of common macro and financial applications that our BNNs can be of practical use, particularly so for observations in the tails of the cross-sectional or time series distributions of the target variables.

**JEL:** C11, C30, C45, C53, E3, E44.

**Keywords:** Bayesian neural networks, model selection, shrinkage priors, macro forecasting.

---

<sup>\*</sup>Corresponding author: Massimiliano Marcellino. Department of Economics, Bocconi University. Address: Via Roentgen 1, 20136 Milano, Italy. Email: [massimiliano.marcellino@unibocconi.it](mailto:massimiliano.marcellino@unibocconi.it). The authors thank Jamie Cross, Martin Feldkircher, Christian Hots-Behofsits, Michael Pfarrhofer, and the participants of the COMPSTAT 2022 conference for helpful comments and suggestions. Hauzenberger gratefully acknowledges financial support from the Jubiläumsfonds of the Oesterreichische Nationalbank (OeNB, grant no. 18304 and 18718), and Huber acknowledges financial support from the Austrian Science Fund (FWF, grant no. ZK 35) and the Jubiläumsfonds of the OeNB (grant no. 18304). The views expressed in this paper do not necessarily reflect those of the Oesterreichische Nationalbank or the Eurosystem.

# 1 Introduction

In recent decades, statistical agencies, governmental institutions and central banks increasingly collect vast datasets. Practitioners and academics rely on these datasets to form forecasts about the future, efficiently tailor policies or improve decisions at the corporate level. However, this abundance of data also gives rise to the curse of dimensionality and questions related to separating signal (i.e., extracting information from important covariates) from noise (i.e., covariates which do not convey meaningful information) are key for carrying out precise inference. Fortunately, the recent literature on statistical and econometric modeling in high dimensions using regularization-based techniques offers a range of solutions (see, e.g., [Carvalho et al., 2010](#); [Bhattacharya and Dunson, 2011](#); [Griffin and Brown, 2013](#); [Belmonte et al., 2014](#); [Huber et al., 2021](#)).

One key shortcoming, however, is that these models often assume linearity between a given response variable (or in general a vector of responses) and a possibly huge panel of covariates. The reason for this is simplicity in estimation and interpretation. Apart from these very general reasons, allowing for arbitrary functional relations in the conditional mean introduces substantial conceptual challenges. For instance, which form should the relationship, encoded by the function  $f$ , between a response  $y_t$  and a set of covariates  $\mathbf{x}_t$  take? Should  $f$  change over time?

[Hornik et al. \(1989\)](#) show that neural networks (NNs) are efficient approximating devices for learning any function  $f$  under relatively mild assumptions. And they do this successfully in fields such as robotics and signal processing. Yet, the performance of NNs in macroeconomic forecasting is not so satisfactory, see e.g. [Makridakis S. \(2018\)](#), a bit better for financial forecasting, see e.g. [Sezer O. \(2020\)](#). A possible reason for this unsatisfactory performance of NNs in economic applications is that off-the-shelf implementations of NNs in statistical packages are often tailored for applications in engineering and computer science, such as image recognition, data compression or classification tasks, rather than for economics and finance. Related to this issue, the specification of NNs remains difficult and researchers rely on cross validation to select NN features such as the form of activation functions, the number of hidden layers and/or the number of neurons.

In this paper, our goal is to blend the literature on Bayesian econometrics with recent advances in NN modeling. We focus on efficient estimation methods for NNs tailored to match features of time series commonly observed in macroeconomics and finance. More-

over, we provide methods that allow for learning the structure of the neural network without the need for cross validation. These techniques require minimal input from the researcher and are robust to mis-specification.

We achieve all this by taking a Bayesian stance. Exact Bayesian approaches to the estimation of neural networks have been proposed in, e.g., MacKay (1992), Neal (1996), Blundell et al. (2015), Gal and Ghahramani (2016), Scardapane et al. (2017), Ghosh et al. (2019), Dusenberry et al. (2020), and Cui et al. (2021). These approaches, however, typically rely on marginal likelihood comparisons to select features such as the activation functions or the number of neurons, which requires re-estimating the model many times, making the computational burden excessive, especially if interest centers on recursive forecasting. Faster computational solutions rely on approximation-based techniques, which replace exact full conditional posterior sampling with an optimization approach that searches for optimal variational densities that are close to the exact posterior. Due to its approximate nature, however, questions related to approximation accuracy typically arise. These relate not only to the precision of point estimators but also to whether sampling uncertainty is adequately taken into account.

As opposed to using variational approximations, our starting point is the literature on Markov chain Monte Carlo (MCMC)-based estimation of NNs. Recent advances in Bayesian statistics in ultra high dimensional models will be used to speed up computation of possibly very complex NNs. To adaptively select the appropriate network structure, we will adopt Bayesian shrinkage techniques and establish a connection between infinite dimensional mixture and factor models and standard approaches commonly used in the estimation of Bayesian neural networks (BNNs). Our goal is to develop methods that can be applied to datasets commonly used by macroeconomists in central banks, academia and governmental institutions. We will pay particular attention to provide techniques that are reliable, require little input from the researcher, but are flexible enough to unveil complex patterns in economic and financial data to ultimately improve decision making.

We first apply these techniques to synthetic data, to evaluate model performance in a controlled environment. Then, we consider four well known datasets, to explore the degree of nonlinearity in prominent macro and financial applications, using both cross-sectional and time series data, and for the latter both monthly, quarterly and yearly frequencies. Our simulations show that our proposed models yield good forecasts across

a large range of different data generating processes (DGPs). In actual data, we find that across the four datasets, flexible models yield more precise density forecasts. These gains in predictive accuracy are especially pronounced for extreme realizations and/or in problematic periods. Related to this finding, empirically we detect that the type and extent of nonlinearity evolves substantially over time, which is automatically taken into account by our enhanced BNN models, while in standard NN models the type and extent of nonlinearity is fixed over the sample. Focusing on the qualitative properties of the forecast gains also reveals that the superior forecasting performance in the tails arises from the capability of the NNs to explain more in-sample variation. This form of benign over-fitting has been found previously in the literature on machine learning (see, e.g., [Bartlett et al., 2020](#)), and we conjecture that it mainly stems from the fact that NNs extract efficiently information on nonlinear relations between the response variables and their predictors.

Overall, from a methodological point of view, our main contribution is to allow for a general specification of networks that can be applied to either dense or sparse datasets, and combines various activation functions, a possibly very large number of neurons, and stochastic volatility (SV) for the error term. From a computational point of view, we develop fast and efficient estimation algorithms for the general BNNs we introduce. From an empirical point of view, we show both with simulated data and with a set of common macro and financial applications that our BNNs can be of practical use, particularly so for observations in the tails of the cross-sectional or time series distributions of the target variables. This also provides empirical evidence in favor of recent theoretical macro models that allow for nonlinearities, see for example [Harding et al. \(2022\)](#) for the case of inflation explained with a nonlinear Phillips curve.

The remainder of the paper is structured as follows. Section 2 describes the econometric model by first deriving the likelihood function, discussing prior choice and sketching the posterior simulation algorithm. The different Bayesian neural networks are then applied to synthetic data in Section 3. In Section 4 we carry out four forecasting exercises to shed light on the extent of nonlinearities in datasets commonly used in macroeconomics and finance. Finally, the last section provides a brief summary and concludes the paper. Technical details and additional empirical results are provided in the appendix.

## 2 An Enhanced Bayesian Neural Network

This section develops our Bayesian model. After discussing key model specification issues in Sub-section 2.1, we introduce suitable Bayesian regularization priors in Sub-section 2.2, discuss modeling the error variance in Sub-section 2.3 and develop posterior computation in Sub-section 2.4.

### 2.1 The model specification

Our goal is to model a macroeconomic or financial time series  $\{y_t\}_{t=1}^T$  with  $T$  denoting the length of the sample. We assume that  $y_t$  depends on a panel of  $K$  covariates, which we store in  $\mathbf{x}_t$ . In very general terms, a nonlinear regression can be written as

$$y_t = \mathbf{x}_t' \boldsymbol{\gamma} + f(\mathbf{x}_t) + \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, \sigma_t^2), \quad (1)$$

where  $\boldsymbol{\gamma}$  is a vector of  $K$  linear coefficients,  $f : \mathbb{R}^K \rightarrow \mathbb{R}$  is a function of unknown (nonlinear) form and  $\varepsilon_t$  is a Gaussian shock with zero mean and time-varying variance  $\sigma_t^2$ . The presence of a time-varying error variance implies that our model adapts to situations not learned through the conditional mean by increasing  $\sigma_t^2$  and thus increasing uncertainty surrounding predictions. It is also worth stressing that the assumption of heteroskedastic Gaussian shocks is not restrictive. In principle, it would be straightforward to incorporate more flexible error distributions based on scale-location mixtures of Gaussians (see, e.g., Escobar and West, 1995).

In macroeconomics and finance,  $f$  is often assumed to be known. For instance, if  $f(\mathbf{x}_t) = 0$  for all  $t$  we end up with a constant parameter regression model. Another commonly used model arises if  $f(\mathbf{x}_t) = \mathbf{x}_t' \boldsymbol{\gamma}_t$  with  $\boldsymbol{\gamma}_t$  denoting  $K$  time-varying parameters (TVPs). Other specifications which can be seen as special cases of Eq. (1) are threshold and Markov switching models (see, e.g., Hamilton, 1989; Tong, 1990; Teräsvirta, 1994), polynomial regression (see, e.g., McCrary, 2008; Lee and Lemieux, 2010) or models with interaction effects (see, e.g., Ai and Norton, 2003; Imbens and Wooldridge, 2009; Greene, 2010).

This brief discussion shows that the choice of  $f$  is one of the most important modeling decisions the researcher needs to take. In this paper, we follow a different route and estimate  $f$ . This can be achieved through nonparametric techniques such as Bayesian

additive regression trees (BART, see e.g., [Chipman et al., 2010](#); [Coulombe, 2020](#); [Huber et al., 2020](#)), Gaussian processes (GP, see e.g., [Williams and Rasmussen, 2006](#); [Crawford et al., 2019](#); [Hauzenberger et al., 2021](#)), splines (see, e.g., [Vasicek and Fong, 1982](#); [Engle and Rangel, 2008](#)) or wavelets (see, e.g., [Ramsey and Lampart, 1998](#); [Gallegati, 2008](#)).

In this paper, we aim to approximate  $f$  using a neural network (NN), due to the good performance of NN in many areas. We use a single hidden layer and  $Q$  neurons, so that the corresponding approximating model reads:

$$f(\mathbf{x}_t) \approx \sum_{q=1}^Q \beta_q h_q(\mathbf{x}_t' \boldsymbol{\kappa}_q + \zeta_q), \quad (2)$$

where  $\boldsymbol{\beta} = (\beta_1, \dots, \beta_Q)'$  denotes a  $Q \times 1$  vector of loadings,  $h_q(\bullet)$  a nonlinear activation function specific to neuron  $q$ ,  $\boldsymbol{\kappa} = (\boldsymbol{\kappa}_1, \dots, \boldsymbol{\kappa}_Q)$  a  $K \times Q$  matrix of weighting coefficients and  $\boldsymbol{\zeta} = (\zeta_1, \dots, \zeta_Q)'$  a  $Q \times 1$  vector of bias terms. If  $Q$  and the form of  $h_q$  are set adequately, this model can approximate any function  $f$  with arbitrary precision ([Hornik et al., 1989](#); [Hornik, 1991](#)). Even more flexibility could be achieved with a multi-layer specification, but at the cost of an even more complex nonlinear specification, which would substantially impact estimation time and complexity. Hence, we work with a single layer specification but permit  $Q$  to be very large.

The representation flexibility of the model in Eq. (2) depends on the number of neurons  $Q$  but also on the functional form of  $h_q$ . In the machine learning literature, both of these are typically treated as hyperparameters and chosen through extensive cross validation. It is also worth stressing that for each of the  $Q$  neurons, we need to estimate  $K$  coefficients in  $\boldsymbol{\kappa}_q$  plus  $Q$  bias terms. This implies that, conditional on a specific  $h_q$ , the conditional mean function features  $(1 + Q)K + Q$  free parameters. If  $T$  is moderate, this large number of parameters might translate into severe overfitting issues and a substantial computational burden. One contribution of our paper is the development of a Bayesian prior setup that decides on the complexity of the neural network automatically. Moreover, we use shrinkage techniques to avoid overfitting issues if  $Q$  and  $K$  are large.

Up to this point we did not discuss how we select  $h_q$ . Since appropriately selecting  $h_q$  is critical for producing precise forecasts ([Karlik and Olgac, 2011](#); [Agostinelli et al., 2014](#)), we treat the functional form of  $h_q$  as an additional (discrete) parameter which we estimate alongside the remaining model parameters. This is discussed in more detail in

the subsequent sub-section.

## 2.2 The priors

**Selecting the number of neurons.** Our prior setup builds on two pillars. First, we select the number of neurons  $Q$  by using insights from the literature on infinite dimensional factor models (Bhattacharya and Dunson, 2011). This literature suggests setting the number of factors to a very large value and then using a shrinkage prior to force the columns of the factor loadings matrix to zero (and thus decide on the effective number of factors). Conditional on knowing  $h_q$  ( $q = 1, \dots, Q$ ) and  $\kappa$ , the model in Eq. (2) can be viewed as a factor model with observed factors. To decide on  $Q$ , we specify the multiplicative Gamma process (MGP) prior developed in Bhattacharya and Dunson (2011) on the elements in  $\beta$ .

The MGP prior assumes that each element  $\beta_j$  arises from a Gaussian distribution:

$$\beta_q \sim \mathcal{N}(0, \phi_{\beta_q}^{-1}), \quad \phi_{\beta_q} = \prod_{r=1}^q \varrho_r, \quad \varrho_1 \sim \mathcal{G}(a_1, 1), \quad \varrho_r \sim \mathcal{G}(a_2, 1), \text{ for } r > 1.$$

Here, for suitable scaling parameters  $a_1$  and  $a_2$ , the shrinkage factor  $\phi_{\beta_q}$  (i.e., precision of  $\beta_q$ ) increases in  $q$ , implying more shrinkage for larger values of  $Q$ . Hence, if we set  $Q$  to a very large value (in all our empirical work we set  $Q = K$ ) our prior increasingly forces elements in  $\beta$  to zero. This implies that at some point  $q^* > q$ , the corresponding values of  $\beta_{q^*}$  can be safely regarded as being zero and the related neuron does not impact the likelihood function.

One shortcoming of this prior is that it only shrinks elements in  $\beta$  to zero. The probability of observing that  $\beta_q = 0$ , however, equals exactly zero. In light of sufficient shrinkage this difference between shrinkage and sparsity should only play a minor role in actual empirical work. However, if the researcher is interested in providing details on the effective number of neurons  $Q^*$ , one could apply a simple thresholding rule similar to the one proposed in Johndrow et al. (2020). Introducing a threshold  $\tau_\beta$  close to zero would allow us to compute the effective number of neurons as follows:

$$Q^* = \sum_{q=1}^Q \mathbb{I}(\phi_{\beta_q}^{-1} > \tau_\beta),$$

where  $\mathbb{I}(\bullet)$  is the indicator function which equals one if its argument is true. It is noteworthy that some point estimate of  $Q^*$  can be used in a second step to select  $Q$  efficiently.

**Shrinking the weighting and linear coefficients.** The next model selection issue relates to the question on which elements in  $\boldsymbol{\kappa}_q$  and  $\boldsymbol{\gamma}$  should be (non-)zero. Similarly to the weights in  $\boldsymbol{\beta}$ , this is achieved through a shrinkage prior. Since  $\mathbf{x}_t$  is potentially large dimensional, standard spike and slab priors in the spirit of [George and McCulloch \(1993\)](#) and [George et al. \(2008\)](#) suffer from mixing issues ([Bhattacharya et al., 2015](#)). As a remedy, we propose using the horseshoe prior ([Carvalho et al., 2010](#)) on the elements of  $\boldsymbol{\kappa}_q$ . The horseshoe is a global local shrinkage prior that implies the following prior hierarchy on each element of  $\boldsymbol{\kappa}_q$ :

$$\kappa_{jq} \sim \mathcal{N}(0, \phi_{\kappa_{jq}}^{-1}), \quad \phi_{\kappa_{jq}}^{-1} = \lambda_{\boldsymbol{\kappa}_q}^2 \varphi_{\kappa_{jq}}^2, \quad \lambda_{\boldsymbol{\kappa}_q} \sim \mathcal{C}^+(0, 1), \quad \varphi_{\kappa_{jq}} \sim \mathcal{C}^+(0, 1),$$

with  $\lambda_{\boldsymbol{\kappa}_q}$  being a global (neuron-specific) shrinkage parameter which forces all elements in  $\boldsymbol{\kappa}_q$  towards the origin,  $\varphi_{\kappa_{jq}}$  is a local scaling parameter that allows for coefficient-specific deviations in light of strong global shrinkage (i.e., if  $\lambda_{\boldsymbol{\kappa}_q} \approx 0$ ). Both, the global and local shrinkage parameters follow a half-Cauchy distribution a priori. Precisely the same prior is used on  $\boldsymbol{\gamma}$ . This prior selects relevant predictors in  $\mathbf{x}_t$  and has been shown to work well in a range of different applications in economics (see, e.g., [Kowal et al., 2019](#); [Huber et al., 2021](#); [Huber and Pfarrhofer, 2021](#); [Carriero et al., 2022](#)).

**Choosing between activation functions.** In the literature on neural networks, the type of activation is often a hyperparameter that is inferred via cross validation. However, this is computationally demanding and has the problem that uncertainty with respect to the choice of the activation function is neglected since predictive distributions are then typically obtained from a fixed set of activation functions. Our approach differs in the sense that we treat the functions  $h_q$  as an unknown quantity and place a prior on it.

We focus on six commonly used activation functions: leakyrelu (1), softmax (2), soft-plus (3), sigmoid (4), relu (5) and tanh (6). Each of these activation functions has different implications on the flexibility of the neural network to capture nonlinearities in the data. [Table 1](#) provides a summary of the functions used.

We construct a prior that remains agnostic about which activation function describes



the data best a priori. To decide on a specific activation function, we introduce a latent discrete random variable  $\delta_q$  ( $q = 1, \dots, Q$ ) that takes integer values between one and six. The probability that  $\delta_q = m$  is set equal to:

$$\text{Prob}(\delta_q = m) = \underline{\omega}_{qm} = \frac{1}{6},$$

and thus assumes that each activation function is equally likely a priori. We store the indicators in a  $Q$ -dimensional vector  $\boldsymbol{\delta} = (\delta_1, \dots, \delta_Q)'$ . In principle, if the researcher has prior knowledge that a given activation function should be nonlinear (either through observing features of the data or theoretical knowledge), this prior can be modified and the sampling step sketched below is the same. The main implication of this prior is that one can think of the activation function  $h_q$  under the prior as a convex combination over different activation functions:

$$h_q(\mathbf{x}'_t \boldsymbol{\kappa}_q + \zeta_q) = \sum_{m=1}^6 \underline{\omega}_{qm} h_q^{(m)}(\mathbf{x}'_t \boldsymbol{\kappa}_q + \zeta_q),$$

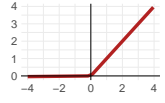

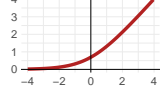
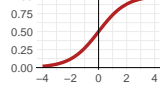


where  $h_q^{(m)}$  denotes one of the six activation functions. Hence, our approach does not decide on one specific activation function but combines all of them using the prior weights. If we confront this prior with the likelihood we end up with posterior weights which provide information on the nature of nonlinearities associated with the  $q^{th}$  neuron. Since we allow for different activation functions across neurons, we substantially increase the flexibility of the model.

We illustrate the effect different activation functions have on the function estimates using a simple univariate example. This example models the relationship between the year-on-year inflation ( $y_t$ ) and the year-on-year money growth rate ( $x_t$ ) in a nonlinear manner. These two series are obtained from the FRED-MD database ([McCracken and Ng, 2016](#)). To account for the leading effect of money growth on inflation, we specify  $x_t$  as the 18<sup>th</sup> lag of money growth (see, e.g., [Reichlin and Lenza, 2007](#); [Amisano and Fagan, 2013](#)). The corresponding nonparametric regression is given by:

$$y_t = f(x_t) + \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, \sigma^2). \quad (3)$$

We compare the effect that different activation functions have on the mean estimate  $f(x_t)$

**Table 1:** Set of activation functions.

Activation function	Equation	Plot
(1) leaky relu	$h_q^{(1)}(z_{qt}) = \begin{cases} 0.01z_{qt} & z_{qt} < 0 \\ z_{qt} & z_{qt} \geq 0 \end{cases}$	
(2) softmax	$h_q^{(2)}(z_{qt}) = \frac{\exp(z_{qt})}{\sum_{\tau=1}^T \exp(z_{q\tau})}$	
(3) softplus	$h_q^{(3)}(z_{qt}) = \ln(1 + \exp(z_{qt}))$	
(4) sigmoid	$h_q^{(4)}(z_{qt}) = \frac{1}{1 + \exp(-z_{qt})}$	
(5) rectified linear unit (relu)	$h_q^{(5)}(z_{qt}) = \max(0, z_{qt})$	
(6) hyperbolic tangent (tanh)	$h_q^{(6)}(z_{qt}) = \frac{\exp(z_{qt}) - \exp(-z_{qt})}{\exp(z_{qt}) + \exp(-z_{qt})}$	

Note: Here,  $h_q^{(m)}$  denotes one of the six activation functions.  $z_{qt}$  is a scalar, which, for example, takes the form  $z_{qt} = \mathbf{x}_t' \boldsymbol{\kappa}_q + \zeta_q$  in Eq. (2).

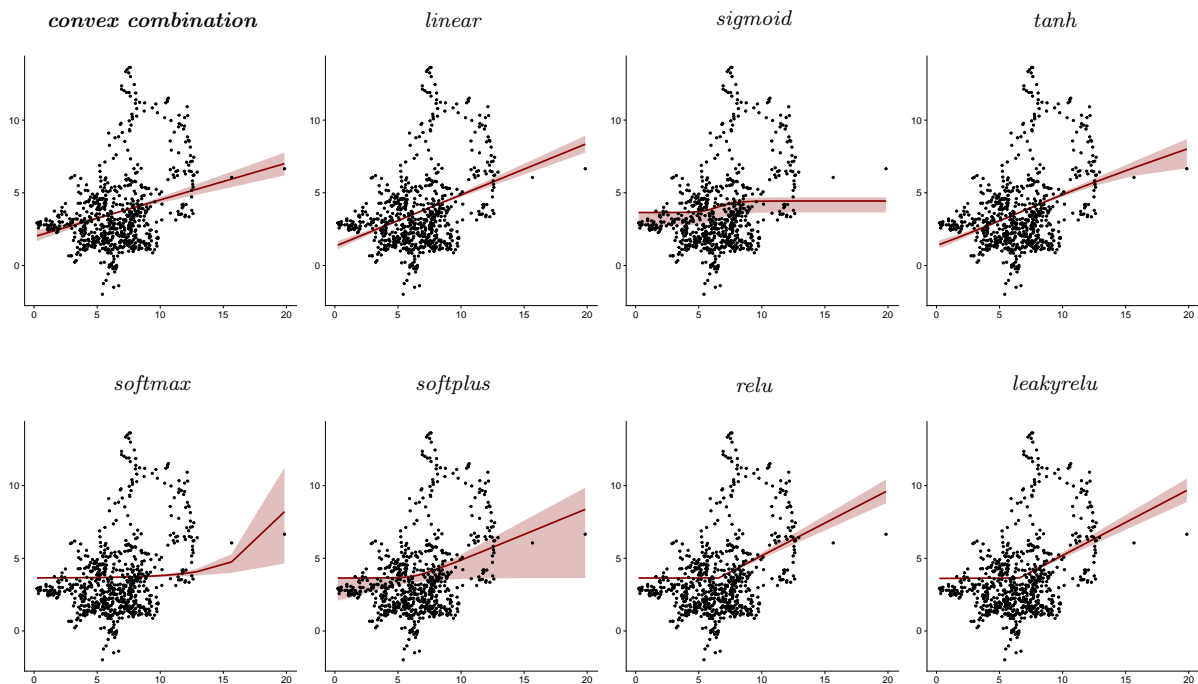
in Fig. 1 by setting  $Q = 1$ . In this figure, the (lagged) values of money growth are on the x-axis while the yearly change in inflation is on the y-axis. Considering the linear specification (i.e.,  $f(x_t) = \beta x_t$ ) suggests a positive relationship between money growth and inflation. When the activation function is nonlinear, the corresponding mean estimates also become nonlinear. These nonlinear activation functions have a common feature: in most cases, changes in inflation are small for money growth up to five percent. This relationship becomes much stronger if money growth exceeds five percent.

There are some exceptions to this pattern. For sigmoid the slope becomes stronger for values of money growth between five and eight percent and then the implied mean function becomes essentially horizontal for money growth above eight percent. In the case of the tanh activation function we observe a piecewise linear functional form that indicates a smaller slope for high values of money growth (i.e. above twelve percent). For softmax we also find a steeper slope for high levels of money growth, suggesting breaks in regression relations for values of money growth of over 16 percent.

Finally, it is worth noting that if we assume that the activation function is unknown (what is called 'convex combination' in the figure), our model recovers a mean relation that is close to linear. Notice that the slope is smaller than in the case of a simple linear activation function.

This short, stylized example illustrates that the different activation functions give rise to different, albeit similar, estimated mean relationships. Since in all our empirical work we set  $Q$  to a large value and use a large panel of covariates the models we propose are capable of extracting nonlinear features.

**Figure 1:** Nonlinearities in the relationship between inflation and money supply.



*Note:* This figure shows the relationship between inflation growth and money growth for the US and illustrates the functional forms of the activation functions specified in Table 1. The data for inflation (i.e., CPIAUCSL) and money supply (i.e., M2SL) are taken from the FRED-MD database as described in McCracken and Ng (2016). We plot the following example:  $y_t = f(x_t) + \varepsilon_t$ , where  $y_t$  is the yearly change in inflation and  $x_t$  is the 18<sup>th</sup> lag of the yearly money supply growth (see Eq. (3)).  $f$  refers to the activation functions specified in Table 1. The top-left panel 'convex combination' refers to our main specification, where we let the data decide on the form of the activation function.

## 2.3 The error variance

Neural networks often explicitly or implicitly assume that the error variance is constant. This assumption implies that the the mean function explains a constant share of variation in  $y_t$  over time. For macroeconomic data, this assumption is strong. In exceptional periods such as the global financial crisis (GFC) or the Covid-19 pandemic mean relations change and the explanatory power of certain elements in  $\mathbf{x}_t$  might deteriorate. As a solution, we

model the error variances in a time-varying manner using a standard stochastic volatility model.

Our model assumes that  $\nu_t = \log \sigma_t^2$  evolves according to an AR(1) process:

$$\nu_t = \mu_\nu + \rho_\nu(\nu_{t-1} - \mu_\nu) + \varsigma_t, \quad \varsigma_t \sim \mathcal{N}(0, \xi_\nu^2),$$

with  $\mu_\nu$  denoting the long-run level of the log-volatility,  $\rho_\nu$  the persistence parameter and  $\xi_\nu^2$  the state equation variance. This model assumes that the error variances evolve smoothly over time (if  $\rho_\nu$  is close to 1) and feature their own shock. For later convenience, we let  $\boldsymbol{\nu} = (\nu_1, \dots, \nu_T)'$  denote the full history of the log-volatilities and  $\boldsymbol{\beta}_\nu = (\mu_\nu, \rho_\nu, \xi_\nu^2)'$  the parameters of the log-volatility state equation.

## 2.4 The posteriors

In general, posterior inference in neural networks is extremely challenging. We tackle some of the issues by proposing a Markov Chain Monte Carlo (MCMC) algorithm that exploits the fact that the effective number of neurons  $Q^*$  is often much smaller than  $Q$ . If this is the case, the corresponding parameters  $\boldsymbol{\kappa}_q$  do not impact the likelihood of the model and hence we can easily simulate them from the prior. Since sampling the  $\boldsymbol{\kappa}_q$ s is typically achieved through variants of the Metropolis Hastings (MH) algorithm and this constitutes the main source of mixing issues in this general class of models, our approach reduces this problem by building on the literature on approximate Bayesian computation and assuming that if the amount of shrinkage introduced through the MGP prior on the  $j^{th}$  neuron becomes large, we can safely ignore the relevant neuron and replace the MH updating step by simply drawing from the prior (which is trivial). In our experiments, we find that this approach works extremely well and yields predictive densities which are very close to the exact ones.

At a general level, we obtain draws from the joint posterior distribution using an MCMC algorithm that cycles between the following steps (with exact details on the full conditional posterior distributions provided in Sub-section A.1 in the appendix):

- Both the linear coefficients  $\boldsymbol{\gamma}$  and the weights  $\boldsymbol{\beta}$  associated with the neurons are obtained jointly from a standard multivariate Gaussian posterior, see Eqs. (A.1) and (A.2).

- For shrinking the linear coefficients, we use the horseshoe prior (Carvalho et al., 2010) and update the corresponding hyperparameters by sampling from inverse Gamma distributions using the auxiliary sampler proposed in Makalic and Schmidt (2015), see Eqs. (A.3) to (A.6).
- The hyperparameters associated with the MGP prior are obtained through simple Gibbs updating steps, see Eqs. (A.7) to (A.8).
- Sampling from  $p(\boldsymbol{\kappa}_q|\bullet)$ , for  $q = 1, \dots, Q$ , is achieved as follows:
  - If the corresponding scaling parameter of the MGP prior exceeds a threshold very close to zero, we sample  $\boldsymbol{\kappa}_q$  using an adaptive MH step, see Eq. (A.10).
  - Otherwise,  $\boldsymbol{\kappa}_q$  is obtained by drawing from the prior.
- The shrinkage hyperparameters of the horseshoe prior on  $\boldsymbol{\kappa}_q$  are obtained from simple inverse Gamma posteriors, see Eqs. (A.11) to (A.14).
- The function  $h_q$  is simulated by first introducing an indicator  $\delta_q$  which takes integer values one to six and indicates the precise function chosen. This indicator is then simulated from a multinomial distribution, see Eq. (A.15), and the relevant functional form of the activation function is chosen.
- Draws of  $\boldsymbol{v}$  and  $\boldsymbol{\beta}_v$  are simulated using the algorithm proposed in Kastner and Frühwirth-Schnatter (2014).

We iterate through our MCMC algorithm 20,000 times and discard the first 10,000 draws as burn-in. Information on the MCMC mixing properties of our sampler is provided in Sub-section B.4 in the appendix.

## 3 Simulation exercise

### 3.1 Design of the simulation exercise

In this section, we illustrate our approach through synthetic data. We consider different data generating processes (DGPs) that range from being linear to highly nonlinear. More precisely, our set of DGPs is comprised of a deep neural network with five hidden layers and the sigmoid activation function, a shallow neural network with a single hidden layer

and a single activation function (randomly selected with an equal probability between the functions listed in Table 1), a restricted case of the latter (i.e., we assume that each neuron is informed by a single covariate in  $\mathbf{x}_t$  by setting  $\boldsymbol{\kappa} = \mathbf{I}_K$ ) as well as a linear DGP.

In general, our DGP assumes that:

$$y_t = f(\mathbf{x}_t' \boldsymbol{\kappa}_{true})' \boldsymbol{\beta}_{true} + v_t, \quad v_t \sim \mathcal{N}(0, \sigma_{true}^2), \quad (4)$$

$$x_{jt} \sim \mathcal{N}(0, 1), \text{ for } j = 1, \dots, K \text{ and } K \in [30, 60], \quad (5)$$

where  $f : R^K \mapsto R^Q$  is either linear (with  $\boldsymbol{\kappa}_{true} = \mathbf{I}_K$ ), a deep NN (with five hidden layers, the sigmoid activation function and the nonlinear coefficients  $\text{vec}(\boldsymbol{\kappa}_{true}) \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{K^2})$  for each layer), a shallow neural network (with a single hidden layer, a randomly selected activation function and  $\text{vec}(\boldsymbol{\kappa}_{true}) \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{K^2})$ ) and a shallow neural network that restricts  $\boldsymbol{\kappa}_{true} = \mathbf{I}_K$  such that each covariate is associated with a single neuron. The number of neurons  $Q(= K)$  is set to imply a medium-scale network ( $Q = 30$ ) or a large one ( $Q = 60$ ). The  $Q$ -dimensional vector of linear coefficients  $\boldsymbol{\beta}_{true}$  captures the notion of the MGP prior and also allows us to distinguish between a truly sparse and a truly dense specification. Elements in  $\boldsymbol{\beta}_{true}$  are defined according to  $\beta_{j,true} \sim N(0, c\alpha^j)$ , where  $\alpha = 0.95$  denotes a coefficient-specific discount factor, while  $c = 4$  in case of the sparse DGP and  $c = 0.04$  in case of the dense DGP. In the dense model 90 percent of the neurons are active whereas in the sparse specification only one out of ten neurons gets a non-zero weight. For the variance, we also choose a large ( $\sigma_{true}^2 = 1$ ) and a small ( $\sigma_{true}^2 = 0.1$ ) specification.

To investigate the performance of the model we carry out a forecasting exercise using synthetic data. We simulate data of length  $T = 200$  and use 100 randomly selected observations to train the different models. Forecast distributions are then computed for the remaining 100 periods in 20 replications. In our simulations we consider five competing models. The first is our Bayesian NN with an activation function common to all neurons (labeled BNN). The second one assumes that each neuron features its own activation function (labeled BNN-NS). Our third model is a restricted version of the BNN which assumes that each neuron is informed by a single covariate in  $\mathbf{x}_t$  (labeled BNN-R). Finally, we estimate a neural network using backpropagation (labeled BNN-BP), which is described in more detail in Sub-section A.2 in the appendix. The benchmark is the linear model with a horseshoe prior and SV, which typically performs very well in forecasting

applications.

### 3.2 BNN forecasting performance with simulated data

Table 2 summarizes the forecasting performance of our proposed model for the DGPs described above in terms of the relative root mean squared errors (RMSEs) to the linear model, so that a relative RMSE smaller than one indicates over-performance with respect to the linear model. To investigate the tail forecasting performance within a controlled environment, the parentheses below include the 25th and 75th quantile scores (QSs). These are relative to the linear model and values smaller than unity indicate outperformance of the corresponding nonlinear model.

The final row shows the mean of relative RMSEs and QSs across all DGPs and for each of the models we consider. This serves as a rough overall measure of predictive accuracy across DGPs. When we focus on the last row of the table we find that BNN yields RMSEs which are, on average, close to the ones obtained from the linear model. The QSs tell a similar story with values that are only slightly smaller than one. Considering the forecast performance of BNN-NS and BNN-R we observe a better average forecasting accuracy (both in terms of RMSEs and QSs). Especially in the right tail we observe a substantial outperformance relative to the linear value, with BNN-R yielding right tail forecasts that are around 11 percent more precise than the ones from the linear model with SV.

These average loss measures mask important differences in predictive accuracy across different DGPs. Honing in on the different cases we find substantial heterogeneity in forecasting performance. In some cases (especially when the DGPs feature nonlinearities) we observe RMSE ratios close to but below one. If the DGP is linear the ratios are close to, but above, one. Specifications such as BNN-NS and BNN-R exhibit a stronger performance. For nonlinear DGPs, these models yield RMSE ratios much below unity (with gains reaching around 50 percent relative to the linear model). Even if the DGP is linear, both models yield RMSEs close to one (and only slightly above one).

When our focus is on tail forecast performance we find a rather consistent story. BNNs with neuron-specific activation functions and restricted variants produce tail predictions which are much more precise than the ones arising from the linear model if the DGP features nonlinearities. If the DGP is linear, differences in QSs indicate only small losses in forecasting accuracy, signaling that the network is doing a good job in learning the true

**Table 2: Synthetic.** Point forecast performance for 100 hold-out observations.

K	DGP		Model				
	Sparsity	Noise	BNN	BNN-NS	BNN-R	BNN-BP	Linear model
Deep NN DGPs							
60	Dimension reduction	L	0.99 (0.97),(0.98)	<b>0.98</b> (0.95),(0.98)	1.00 (0.96),(0.99)	1.09	1.46 (1.93),(1.81)
		S	0.99 (0.97),(0.99)	<b>0.98</b> (0.97),(0.98)	1.01 (0.97),(0.99)	1.10	1.06 (1.31),(1.27)
Shallow NN DGPs							
30	Dense	L	0.84 (0.82),(0.77)	0.84 (0.82),(0.81)	0.99 (0.98),(0.99)	<b>0.64</b>	4.00 (4.65),(4.70)
		S	0.85 (0.81),(0.81)	0.73 (0.68),(0.68)	0.99 (0.98),(0.99)	<b>0.60</b>	3.87 (4.49),(4.51)
	Sparse	L	1.00 (0.93),(1.00)	<b>0.96</b> (0.91),(0.96)	0.99 (0.98),(0.99)	0.98	4.25 (5.20),(4.46)
		S	1.00 (0.92),(1.03)	<b>0.94</b> (0.90),(0.96)	0.99 (0.99),(0.99)	0.99	4.13 (5.06),(4.30)
60	Dense	L	1.00 (0.98),(0.99)	1.01 (0.95),(1.01)	0.99 (0.96),(0.98)	<b>0.91</b>	8.63 (10.17),(9.91)
		S	1.00 (0.97),(0.97)	1.00 (0.97),(0.96)	0.99 (0.96),(0.98)	<b>0.90</b>	8.56 (10.07),(9.78)
	Sparse	L	<b>0.97</b> (0.95),(0.95)	1.00 (0.95),(0.99)	0.98 (0.96),(0.97)	1.00	7.51 (8.86),(8.68)
		S	<b>0.98</b> (0.94),(0.98)	1.00 (0.97),(0.98)	0.99 (0.96),(0.97)	1.02	7.40 (8.69),(8.43)
Restricted shallow NN DGPs							
30	Dense	L	1.02 (1.01),(1.01)	0.95 (0.93),(0.95)	<b>0.91</b> (0.87),(0.90)	1.01	1.16 (1.35),(1.42)
		S	0.99 (0.97),(0.97)	0.73 (0.72),(0.69)	<b>0.57</b> (0.55),(0.53)	0.91	0.63 (0.70),(0.72)
	Sparse	L	1.05 (1.00),(1.02)	0.97 (0.95),(0.96)	<b>0.87</b> (0.84),(0.84)	1.03	1.25 (1.52),(1.46)
		S	1.03 (1.00),(1.04)	0.83 (0.82),(0.81)	<b>0.48</b> (0.45),(0.47)	1.04	0.74 (0.87),(0.81)
60	Dense	L	1.00 (0.99),(0.98)	0.84 (0.84),(0.81)	<b>0.79</b> (0.81),(0.75)	1.13	1.38 (1.50),(1.65)
		S	0.95 (0.94),(0.94)	0.46 (0.45),(0.44)	<b>0.38</b> (0.38),(0.35)	1.05	0.97 (1.03),(1.08)
	Sparse	L	1.00 (0.99),(0.97)	0.93 (0.93),(0.90)	<b>0.84</b> (0.83),(0.80)	1.06	1.37 (1.62),(1.65)
		S	0.97 (0.98),(0.95)	0.75 (0.77),(0.72)	<b>0.46</b> (0.44),(0.42)	1.14	0.90 (1.01),(1.00)
Linear DGPs							
30	Dense	L	<b>1.02</b> (1.01),(1.02)	1.04 (1.03),(1.03)	1.04 (1.03),(1.03)	1.23	1.01 (1.08),(1.08)
		S	<b>1.03</b> (1.03),(1.02)	1.03 (1.03),(1.03)	1.06 (1.06),(1.04)	1.92	0.32 (0.32),(0.33)
	Sparse	L	1.09 (1.08),(1.09)	1.02 (1.02),(1.02)	<b>1.02</b> (1.01),(1.01)	1.21	1.09 (1.19),(1.12)
		S	1.01 (1.01),(1.01)	<b>1.01</b> (1.00),(1.01)	1.04 (1.03),(1.03)	2.13	0.35 (0.36),(0.34)
60	Dense	L	<b>1.02</b> (1.02),(1.02)	1.04 (1.04),(1.03)	1.05 (1.06),(1.04)	1.46	1.04 (1.10),(1.07)
		S	<b>1.04</b> (1.03),(1.04)	1.04 (1.04),(1.05)	1.10 (1.10),(1.11)	2.81	0.33 (0.33),(0.31)
	Sparse	L	1.01 (1.00),(1.02)	<b>1.01</b> (1.00),(1.01)	1.03 (1.02),(1.03)	1.34	1.17 (1.21),(1.20)
		S	<b>1.00</b> (1.01),(1.01)	1.01 (1.02),(1.02)	1.06 (1.07),(1.08)	2.89	0.38 (0.36),(0.37)
All DGPs			0.99 (0.97),(0.98)	0.93 (0.91),(0.91)	<b>0.91</b> (0.89),(0.90)	1.25	2.50 (2.92),(2.83)

*Note:* The table shows root mean squared errors (RMSEs) relative to the benchmark linear model. The numbers in parentheses show the 25/75 quantile scores. In bold we mark the best performing model for each case. The grey shaded area gives the actual RMSE scores of the benchmark. Results are averaged across the hold-out.



structure of the DGP even in simulations from the linear model. Overall, this ranking in terms of Qs suggests that BNNs are better in capturing tail events, with the latter happening more often with BNN-type DGPs.

Finally, in terms of the effects of the types of DGP, a dense structure leads often to slightly larger gains than a sparse structure; the performance for 30 regressors is generally better than for 60 (with the exception of the restricted shallow NN DGP, likely due to its simplified structure); and there are typically little differences between a small and a large error variance, but small is better for the linear model and with the restricted shallow NN DGP. It is also worth mentioning that with the linear DGP, small is better than larger variance for the linear model, but not for the various BNNs, suggesting that the latter are less affected by the size and volatility of the shocks due to their more flexible specification for the conditional mean.

## 4 Empirical Applications

We now consider the performance of the BNN models in four different topical empirical applications, see Sub-section 4.1 and the summary in Table 3 for more details, and carry out extensive forecasting exercises in Sub-section 4.2. These allow us to assess whether, from a predictive viewpoint, allowing for nonlinearities of an unknown form is preferable relative to simpler model specifications. We also include in the set of competing models an alternative very flexible nonparametric specification, Bayesian additive regression trees (BART, see e.g. Chipman et al. (2010) and Sub-section A.3 in the appendix), to assess whether or not BNN can outperform it. In Sub-section 4.3 we investigate the role of the various activation functions / types of nonlinearity. In Sub-section 4.4 we focus on the effective number of neurons, as a summary measure of model complexity. Finally, in Sub-section 4.5 we relate the in-sample and out-of-sample results. Section B.2 in the appendix presents additional results related to the relative performance of each model for each dataset, also assessing the statistical significance of the differences in RMSE and LPL by means of the Diebold and Mariano (1995) test. It also contains results for different forecast horizons.

## 4.1 The four applications

In this sub-section, we provide details on the different applications and the datasets involved.

**Macro A: modeling and forecasting key macroeconomic variables.** The first application focuses on modeling and forecasting key US macroeconomic variables, using the popular FRED-MD database proposed in [McCracken and Ng \(2016\)](#). To gain a comprehensive picture of the importance of nonlinearities in large macro datasets, our forecasting exercise includes the consumer price (CPIAUCSL) inflation rate as specified in [Stock and Watson \(1999\)](#) and labeled as **Macro A.1**, the monthly growth rate of industrial production (INDPRO), labeled as **Macro A.2**, and the monthly growth rate of employment (CE16OV), labeled as **Macro A.3**. The sample ranges from January 1960 to December 2020 and includes 120 economic and financial variables for the large covariate set. We present results obtained from estimation based on smaller sets of variables in the appendix.

We compute the one- and three-month-ahead predictive distributions for our hold-out sample, which starts in January 2000 and ends in December 2020 (i.e., 252 monthly hold-out periods). These forecasts are obtained recursively, meaning that we use the data through January 2000 as a training sample and then forecast one- and three-month-ahead, where the latter are obtained using the direct method (see, e.g., [Marcellino et al., 2006](#)). After obtaining the corresponding predictive densities, the sample is expanded by a single month. This procedure is repeated until the end of the sample is reached.

**Macro B: long-term economic growth in a cross-section of countries.** In the second application we study the presence of nonlinearities using the standard cross-sectional dataset proposed in [Barro and Lee \(1994\)](#). This dataset comprises a wide range of cross-country characteristics over the period 1960 to 1985. As dependent variable, we model the average growth rate of GDP per capita and regress it on the initial level of the exogenous variables to avoid endogeneity issues ([Barro and Lee, 1994](#)).

To investigate the extent of nonlinearities, we randomly pick 45 countries and predict the remaining 45 countries. This is repeated 100 times. Since flexible models should do better in the tails, we also consider a case in which our hold-out includes only countries

with GDP growth rates outside the 25<sup>th</sup> and 75<sup>th</sup> percentiles (i.e., countries with very low/very high growth rates of GDP per capita).

**Macro C: modeling and forecasting quarterly exchange rate returns.** The recent literature on forecasting exchange rates suggests that accounting for nonlinearities increases predictive accuracy and helps to outperform simple benchmarks such as the random walk (see, e.g., [Wright, 2008](#); [Rossi, 2013](#); [Huber and Zörner, 2019](#); [Beckmann et al., 2020](#)). We investigate this claim more carefully and forecast the USD/GBP exchange rate using our set of models. We construct a medium-sized application using a kitchen sink regression with 19 fundamentals including macroeconomic variables (i.e., unemployment rate and real GDP), price indices (i.e., producer and consumer price index as well as the oil price), short- and long-term interest rates, monetary supply measures and stock market variables (i.e., S&P 500 and VIX). Additional results based on smaller, theoretically motivated models can be found in the appendix.

Our sample starts in the first quarter of 1990 and ends in the last quarter of 2019. We again use a recursive forecasting design and focus on one- and four-steps-ahead predictions. The initial training sample ranges from 1990Q1 to 1999Q1. The remaining observations are left for forecast validation (i.e., 80 quarterly observations).

**Finance: forecasting the equity premium.** In our finance application we assess the effect of controlling for nonlinearities on estimating and predicting US aggregate stock returns. We use the dataset described in [Welch and Goyal \(2008\)](#) which includes 16 predictors covering macroeconomic as well as interest rates and stock market variables. The variable to predict is the US equity premium measured by the continuously compounded returns on the S&P 500 index.<sup>1</sup> The sample spans the period 1948 to 2020 at a yearly frequency. Our hold-out starts in 1965 and ends in 2020 (i.e., 56 yearly observations). It is worth stressing that our forecast design implies a small number of observations in the initial training sample. This serves to illustrate how neural networks learn in light of very short time series.

---

<sup>1</sup>In the appendix we forecast the equity premium using univariate models, each including a variable deemed as relevant by the literature. Those are inflation (see, e.g., [Fama and Schwert, 1977](#); [Campbell and Vuolteenaho, 2004](#)), the term spread (see, e.g., [Campbell, 1987](#); [Fama and French, 1989](#)), dividend yield (see, e.g., [Fama and French, 1988](#); [Hodrick, 1992](#)) and the dividend price ratio (see, e.g., [Campbell and Shiller, 1988](#); [Lewellen, 2004](#)).

Table 3 summarizes the applications, provides information on the datasets, the sample and additional details on the forecasting exercises.

**Table 3:** Empirical applications.

	Dependent variable	Set of predictors	Sample	Range	Horizon	Hold-out	Source
<b>Macro A</b>	A.1) Industrial production A.2) Inflation A.3) Employment	Large (120 economic & financial variables)	Monthly data for the US	1960M1 to 2020M12	one-step- and three-steps-ahead	2000M1 to 2020M12	McCracken and Ng (2016)
<b>Macro B</b>	Average economic growth rate	60 country-specific characteristics	Cross-section	90 countries	100 random samples	45 countries	Barro and Lee (1994)
<b>Macro C</b>	USD/GBP exchange rate returns (qoq)	20 exchange rate determinants	Quarterly data for the US and UK	1990Q1 to 2019Q4	one-step- and four-steps-ahead	2000Q1 to 2019Q4	Wright (2008); Rossi (2013)
<b>Finance</b>	Equity premium	16 economic & financial variables	Annual data for the US	1948 to 2020	one-year-ahead	1965 to 2020	Welch and Goyal (2008)

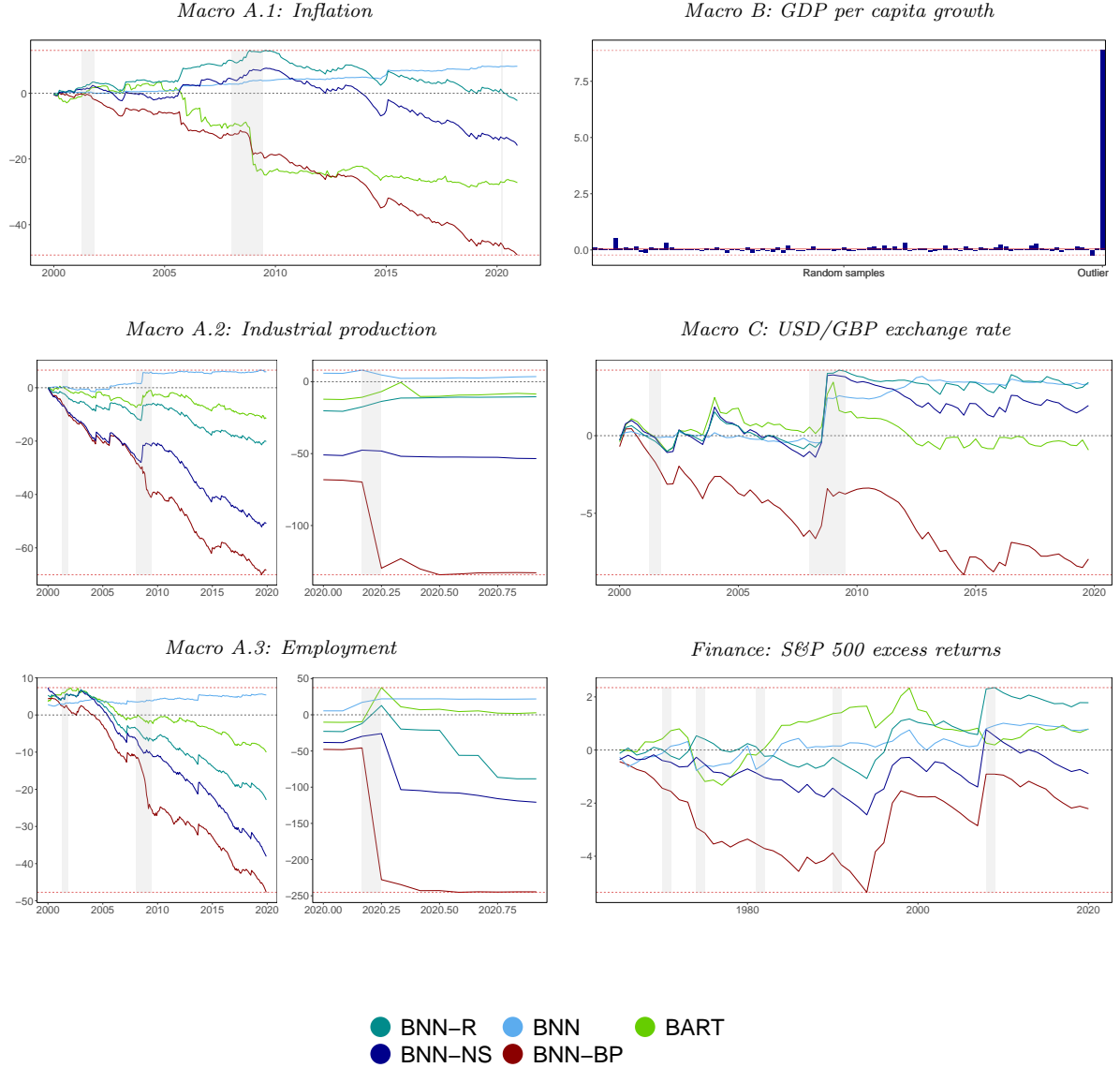
*Note:* The table gives an overview of the different empirical applications with which we test our proposed Bayesian neural network approach and present results in Sub-section 4.2. For more details on all datasets used for evaluating the performance of our BNNs we refer to Sub-section B.1 in the appendix.

## 4.2 Out-of-sample predictive accuracy

In this section we present the main forecasting results. For all four applications we focus on one particular dataset (**Large** for Macro A, **all 20 determinants** for Macro C and **all 16 variables** for Finance) and the one-step-ahead horizon. Results for the other smaller datasets and higher-order forecasts are provided in Section B.2 in the appendix. For the former, the forecasting performance is often worse (or not better) than that based on the large dataset, indicating that large information sets matter also when included in nonlinear models.

Figure 2 shows cumulative one-step-ahead LPLs relative to the linear benchmark model over time and across all empirical applications considered, except for Macro B for which the LPLs are not cumulated, with positive values indicating that the nonlinear models are better than the linear benchmark. We start our discussion with the results for the Macro A dataset (Macro A.1 - Macro A.3). For all three focus variables under consideration a single best performing model emerges: the BNN with a common activation function. The other versions of the BNN often perform slightly worse and are outperformed by the linear benchmark model with SV (where SV plays an important role to enhance the density performance of the linear model). Notice, however, that across all three target variables we find that during recessions (and sometimes during the pandemic) the BNNs yield more precise density predictions than the linear model. The worst per-

**Figure 2:** Cumulative LPLs against the benchmark for the one-step-ahead predictive densities.



*Note:* For the time series applications we plot the evolution of cumulative log predictive likelihoods (LPLs) against the benchmark for the one-step-ahead predictive densities. Here, we choose the linear model of each specification as our benchmark to highlight the effect of controlling for nonlinearities. Note that this is in contrast to the tables in Section B.2 in the appendix where we choose a global benchmark for each application. The red dashed lines denote the max./min. LPLs at the end of the hold-out sample, while the gray shaded areas indicate the NBER recessions. For Macro B we plot relative LPLs against the linear model. The red dashed lines denote the max./min. LPLs.

forming model is also consistent across target variables: BNN-BP. The dismal performance of BNN-BP is, similarly to our synthetic data exercise, driven by too narrow predictive bounds. This claim is evidenced by the fact that BNN-BP performs poorly during extreme periods (with the slope of the relative LPL curves becoming much steeper). BART is also dominated by BNN for all variables and sample periods, suggesting that BNN can provide an even more flexible specification, at least for these variables. Moreover, from the appendix (see, Table B.2 and Table B.3), it turns out that BNN works well for all variables with respect to the linear model also when predicting three-months-ahead.

Zooming into the different focus variables reveals some interesting patterns over time. For inflation, we find that BNN-R and BNN-NS provide good density predictions prior and through to the GFC. In both cases, the forecasting gains then deteriorate. The BNN with a single activation function provides inflation forecasts which are close to the one of the linear model until late 2005 and then becomes consistently better (without any sudden declines in relative forecasting accuracy).

Considering output forecasts (measured through industrial production) suggests that nonlinear models do well during the GFC. All models (except BNN-BP) improve upon the linear regression model during that particular episode. Again, BNN performs best overall, with most gains being driven by the superior performance through 2008/2009 and, to a somewhat lesser extent, in the years from 2005 to 2007. During the pandemic this pattern is somewhat less pronounced.

When focusing on employment, we find that especially in the early part of the sample (up to 2005) most nonlinear models do well. However, from late 2005 onward we observe a steady decline in relative cumulative LPLs. These are, again, visible for all models except for the BNN, which emerges as the model that produces the most precise density predictions.

Since there is some time variation in the LPL gains from the BNN specifications relative to the benchmark linear model, we have run the [Giacomini and Rossi \(2010\)](#) test to assess whether or not the outperformance can be considered as stable over time. Results reported in Figure B.5 in the appendix indicate that the null hypothesis of stability is not rejected, except for employment during the Covid-19 period.

Next we turn to the cross-sectional economic growth dataset (Macro B). Since this is a cross-section we do not show cumulative LPLs but, for each (randomly chosen) hold-out,

the corresponding log predictive likelihood of the single best performing model (in this case BNN-NS) against the linear regression model. Notice that in this exercise no model features stochastic volatility. When we randomly drop countries from the dataset and train the model based on the remaining countries, we obtain LPLs which are very often slightly better than the ones of the linear benchmark. However, there are also cases where the neural network performs worse than the linear model. But this only happens rarely and in the vast majority of hold-outs BNN-NS yields superior forecast densities. When we focus on countries that display extreme average growth rates of income per capita, predictive gains become enormous (more so for BNN-NS than for BART, as shown in Figure B.2 the appendix).

The results based on the first two datasets suggest that our Bayesian variants of a neural network do particularly well when time series display sharp changes or observations are extreme. Exchange rate dynamics also share features such as sudden changes in the level and/or volatility clustering. Hence, nonlinear techniques should be well suited for producing accurate forecasts. This indeed comes out from the figure associated with Macro C. Early in the sample, the different BNNs are slightly worse than the linear benchmark (with BNN-BP, again, being substantially inferior). However, as time passes by, flexible models seem to become better and sharply improve upon the linear specification in late 2004 and during the GFC. In this specific example, BNN-R and BNN exhibit a very similar predictive performance. Most of the gains from using a nonlinear model are again obtained during times of economic stress. In particular, the US dollar sharply appreciated vis-à-vis the pound during the GFC and all neural network models are quick to adapt to this pattern. Table B.5 in the appendix shows that in terms of RMSEs there are little or no gains from the various BNN specifications, confirming that the LPL gains do not come from the center of the distributions. Moreover, there are some LPL gains also at the one-year horizon, and each single fundamental driver is associated with very similar LPL gains (and no or very small RMSE gains).

Finally, we consider whether BNNs do well when forecasting yearly equity returns. As opposed to previous datasets, we find that the nonparametric BART model performs extraordinary well until the GFC in 2008 (with a period of somewhat weaker predictive accuracy in the late 1970s and early 1980s). The BNN improves upon the benchmark from 1984 onward but displays little variation in terms of LPLs over the hold-out sample.

This is in contrast to BNN-R which sharply improves upon the linear benchmark model in the 1990s and then displays a superior performance (across all models considered) during the financial crisis. Among the individual predictors, Table B.6 in the appendix shows that the dividend price ratio works best, with some gains both in terms of RMSE and LPL, but the differences with the other single drivers are small.

Collecting similarities across all four forecasting exercises yields a story that BNNs work well during recessionary episodes or when the target variable takes on extreme values (i.e. values that are located in the tails). This result holds irrespective of whether the data is monthly (and thus features much more high frequency variation), quarterly, yearly or if we focus on a cross-sectional dataset. The key question, which we are going to investigate in the next two sub-sections, is what determines this strong performance in the tails.

### 4.3 Which activation functions?

One of the questions that arises is which activation functions give rise to the good forecasts during extreme periods. We answer this by computing the posterior inclusion probability (PIP) that a given function is chosen by our algorithm. This is obtained by taking draws from the posterior of  $\delta_q$  and then computing

$$Prob(\delta_q = m | \text{Data}) = \frac{\sum_{s=1}^S (\delta_q = m)}{S},$$

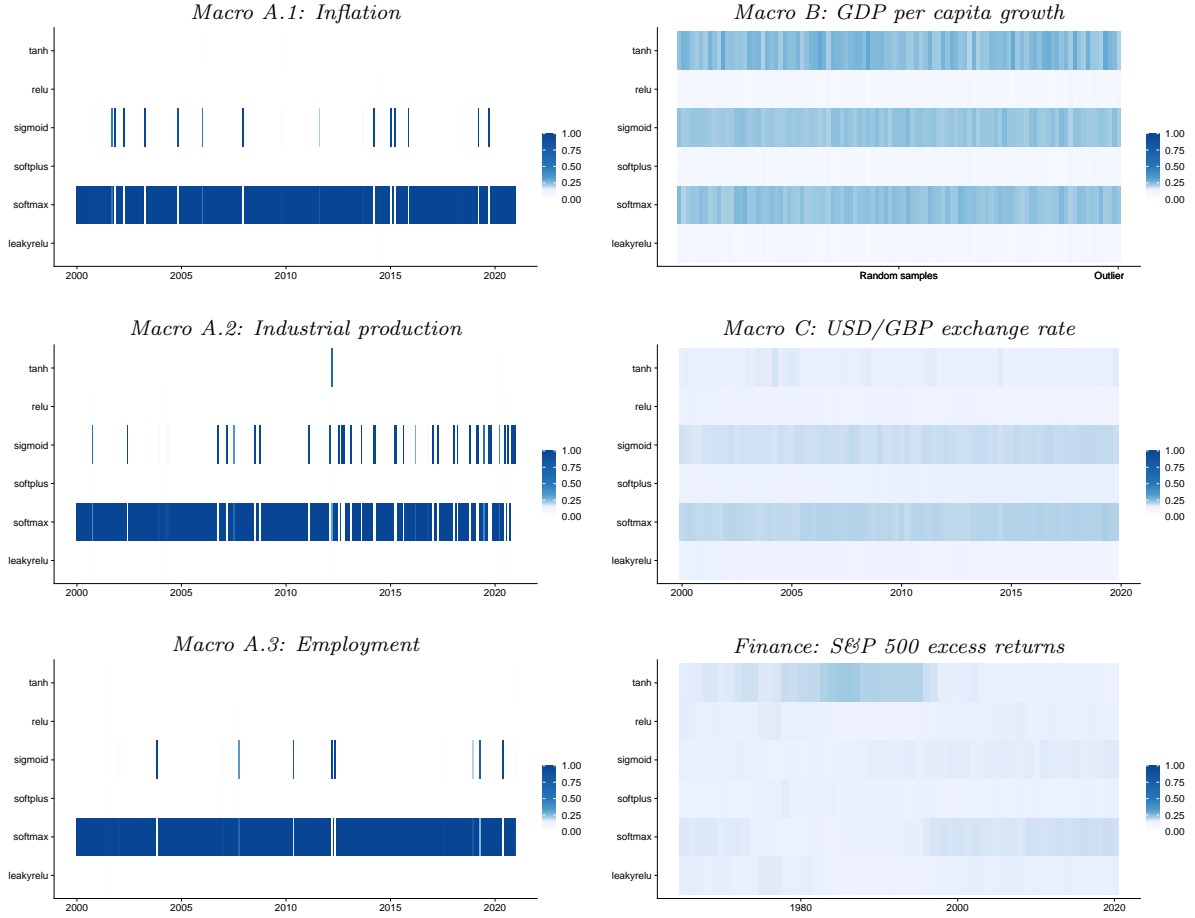
where  $S$  is the number of retained draws from the posterior. In case of the neuron-specific specification we report averages across neurons to simplify the exposition.

Figure 4 shows the PIPs of the different activation functions across datasets over the respective hold-out periods. Dark blue shaded cells imply that the corresponding activation function (in the rows of the heatmap) receives substantial PIP whereas white cells signal low PIPs of a respective activation function.

At a very general level, we find two patterns. The first relates to activation functions in our standard monthly macro dataset. Modeling industrial production, inflation and employment is best done with a neural network that features the same activation function (softmax) for most of the hold-out period. Other activation functions receive a weight close to zero (i.e. are never included). However, there are distinct regimes (such as a



**Figure 3:** Choice of the activation function.



*Note:* This figure shows posterior inclusion probabilities (PIPs) over the hold-out period for the best performing models for each dataset. If a model with neuron-specific activation function is chosen the corresponding PIPs represent averages across neurons.

brief period during the GFC in the case of CPI inflation and industrial production) where the PIPs change so as to attribute most posterior probability to the sigmoid activation function. This shift happens later for employment which seems to lag the business cycle (i.e. employment reached its trough much later than industrial production growth during the GFC). Notice that softmax overweights extreme values of  $\mathbf{x}_t' \boldsymbol{\kappa}_q + \zeta_q$ , implying that if there is a switch in activation functions, we observe sharp changes in the dependent variable and this is captured through abrupt shifts in  $f(\mathbf{x}_t)$ . This provides empirical evidence in favor of recent theoretical macro models that allow for nonlinearities, see for example [Harding et al. \(2022\)](#) for the case of inflation explained with a nonlinear Phillips curve.

Second, when we focus on the other macro datasets we observe less time variation in PIPs over the hold-out period and, more importantly, that many more activation functions receive posterior weight and thus drive the actual results. For Macro B, three

activation functions (tanh, sigmoid and softmax) receive almost 90 percent of posterior mass with relu, softplus and leakyrelu sharing about ten percent. A similar but more attenuated picture arises for Macro C and Finance where sigmoid and softmax receive more posterior weight. Notice that for Finance the PIPs suggest that tanh receives much posterior weight until the late 1990s and then drops close to zero. From the late 1990s onward, tanh is replaced by softmax which receives substantial posterior weight. From Table 1, this implies that extreme values of variables such as dividend price ratios can have a much stronger effect on the stock returns than values close to their sample mean.

#### 4.4 Effective number of neurons $Q^*$

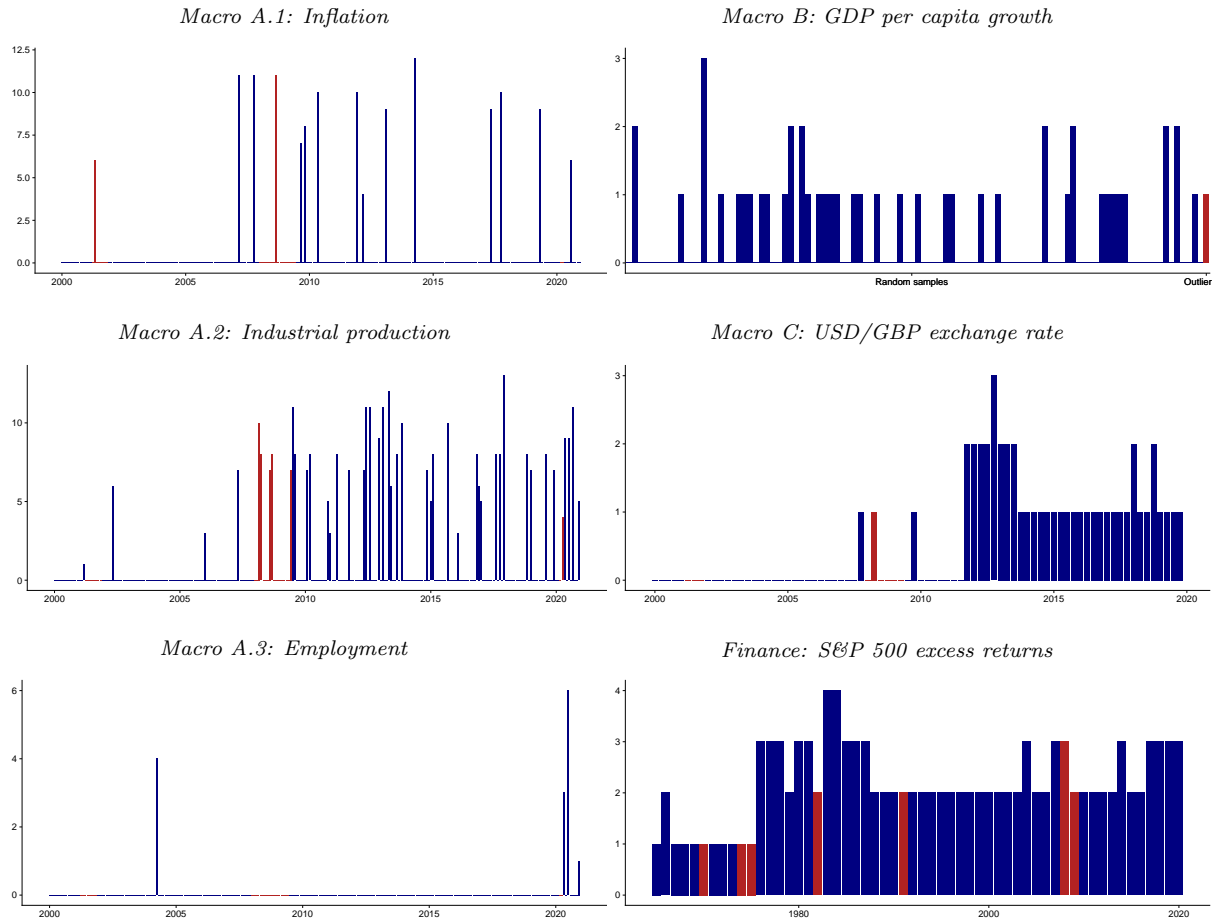
In the previous sub-section we focused on which activation function (and thus form of nonlinearity) gives rise to the forecast distributions. This analysis, however, does not tell us anything about the quantitative relevance of the nonlinear part of the model. In this sub-section we will focus on whether the forecast distributions are generated by models that feature a large number of neurons (i.e. models that are highly nonlinear) or by models that feature few active neurons (i.e. models closer to the linear specification).

Figure 4 shows the number of active neurons for the different applications and over the hold-out period. A simple consideration of the axis scale reveals that, if any neuron is included, the number of neurons is much larger for the monthly time series in the Macro A dataset (ranging from six to around twelve active neurons). For the remaining applications, this number is considerably lower (ranging from two up to four active neurons).

Focusing on the different applications we find that there is substantial heterogeneity with respect to the number of neurons over time. For Macro A, we find that in most periods, the forecasting model is actually a linear regression model. However, in (and around) recessions (indicated by the red bars), we find that the number of neurons increases appreciably. This pattern is much more pronounced for inflation (during the dotcom recession and the GFC) and industrial production (during the GFC and the early period of the pandemic).

For Macro B, nonlinearities are included for most random samples and the outlier sample. Notice, however, that only one to three neurons are included, indicating rather simple forms of nonlinearities. For Macro C we observe that predictions have been mostly

**Figure 4:** Active number of neurons.



*Note:* This figure shows the number of significant neurons for the best performing BNN of each application over the hold-out period. These are determined by computing the 5<sup>th</sup> / 95<sup>th</sup> posterior credible intervals for each  $\beta_q$  and if the credible intervals do not include zero the corresponding neuron is counted as being active a posteriori. In red we indicate the NBER recessions.

generated using a linear specification (with some exceptions, such as the third quarter of 2008) up to 2011. From 2011 onward, the model includes one up to three neurons.

Finally, S&P 500 excess return predictions are based on models that include one up to four neurons for all periods in the hold-out sample. Over time, there is some heterogeneity with respect to the number of neurons. In the late 1970s and early 1980s, three to four neurons are always entering the model. This number drops to two for most of the 1990s and early 2000s. More neurons are included during the downturn in stock markets observed in the GFC and in 2018-2019, a period in which the US Fed increased its policy rate for the first time since 2008.

## 4.5 The relationship between in-sample fit and out-of-sample predictability

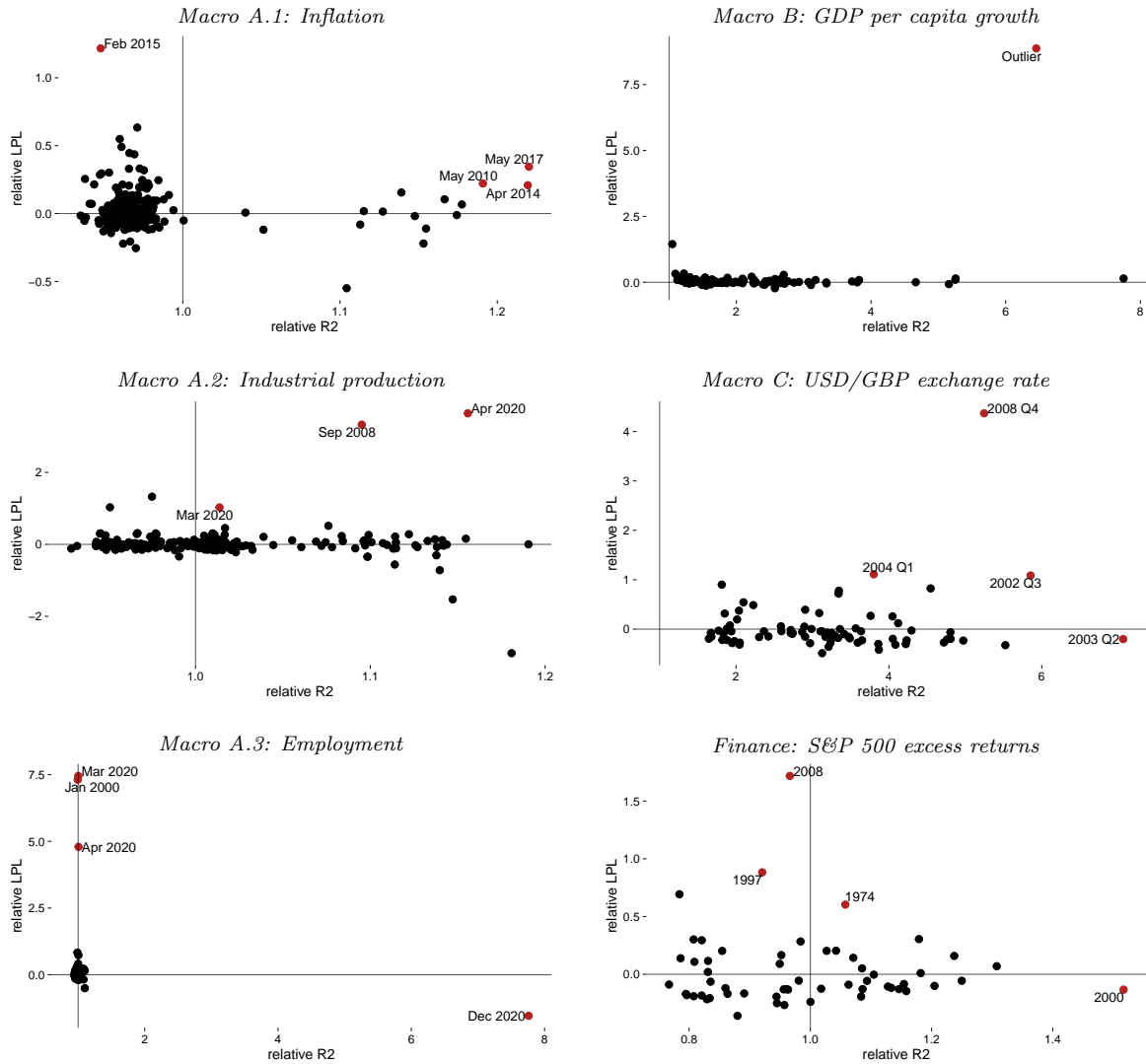
The results up to this point tell a story that flexible models help in the tails, are competitive in normal times and, depending on the dataset, the nature of nonlinearities might be subject to structural breaks or a combination of different types of nonlinearities induced by different activation functions. In this sub-section we ask whether neural networks extract information from  $\mathbf{x}_t$  that linear models can not exploit and how this impacts predictive accuracy (measured again through LPLs).

To achieve this, we compute the amount of variation explained through the conditional mean piece (labeled R2) of the best performing neural network for each application and the linear regression model. These R2s are computed recursively and put in relation to the corresponding  $t$ -by- $t$  LPL using a simple scatter plot. The scatter plots are provided in Figure 5. The horizontal line at zero implies that if a point is below zero, the linear regression model produces superior density forecasts whereas in the opposite case, the BNN is forecasting better. Points to the left of the vertical line (which stands at one) imply less explanatory power of the BNN whereas points to the right indicate that the conditional mean part of the BNN explains more of the variation in the response as the linear model.

Two examples illustrate how the scatter plot can be interpreted. Points in the quadrant with  $R2 > 1$  and  $LPL > 0$  represent situations where the BNN is extracting information from  $\mathbf{x}_t$  and this information pays off for density forecasts. If  $R2 \approx 1$  but  $LPL > 0$  both models explain a similar amount of in-sample variation but density forecasting performance of the BNN is superior. In this case, these differences are likely driven by higher order moments of the predictive distribution.

Analyzing the scatter plots in Figure 5 provides several insights. We find that for Macro A.1, inflation, the vast majority of points is clustered to the left of the vertical line. This implies that the BNN provides a somewhat lower in-sample fit which often translates into slightly better out-of-sample LPLs, since the majority of points lie above the zero line. When we focus on extreme cases (i.e., large R2 ratios and positive/negative LPLs) no clear pattern emerges. There are points in which the BNN predicts better out-of-sample but also explains more in-sample (specifically May 2010, April 2014 and May 2017).

**Figure 5:** Relative R2 against relative LPL



*Note:* This figure shows relative R2 versus relative log predictive likelihood (LPL) of the best performing BNN against the linear model for each of our four applications. We color observations which feature high in-sample fit and high predictive accuracy relative to the benchmark.

This story carries over to output growth predictions (Macro A.2). Several points lie in the high R2, positive LPL quadrant but there is also a substantial amount of observations in the other quadrants. However, one of the consistent patterns emerging from our forecasting exercise was that BNNs work much better during economic downturns. When we focus on the points far out in the north-eastern corner we find periods which are characterized by high R2s but also substantially higher LPLs. The first is September 2008, the month of the bankruptcy of Lehman brothers that led to sharp declines in stock markets and contractions in economic activity and the second is April 2020, a month characterized by the US adopting a large battery of containment measures such as social distancing and lock-downs. For March 2020, a similar but less pronounced pattern arises.

In these three months, the BNN was explaining 10 (for September 2008) to 20 (for April 2020) percent more in-sample variation than the linear model. This translated into much better LPLs.

For employment (Macro A.3) we find a different picture. Most points are clustered around relative R2s close to unity and LPLs being positive. However, the first months of the pandemic are marked as outliers. Especially in March and April 2020 employment numbers exhibited historic declines. The fact that these months are located on the vertical line indicate that the in-sample fit was close to being the same as in the case of the linear model but LPLs are much higher. This indicates that the BNN does a better job in capturing these observations through higher order non-Gaussian features in the predictive distribution. Inspection of the relevant densities (not shown) reveals much heavier tails and thus a higher probability of capturing these large outliers.

For Macro B, the cross-section growth application, an interesting pattern emerges. All points are located to the right of the vertical line, indicating that the BNNs consistently explain more in-sample variation than their linear counterparts. This often leads to better LPLs (but not always). Notice, however, that the outlier sample (i.e. the one that includes only countries with extreme GDP per capita growth rates) are located far out in the north-east of the scatter plot. For this specific verification sample the neural network's R2 is more than six times larger than the one of the benchmark and the LPL is over 7.5. This indicates that the neural network is capable of extracting information from the covariates in the dataset that would, if a linear approach is adopted, be lost. And this information seems to pay off for density forecasts.

Interestingly, a similar story arises if we consider Macro C. For exchange rate forecasting, the neural network always features larger R2s than the linear model and this often translates into superior LPLs. In 2008Q4, a quarter where the US dollar appreciated sharply against the pound, the BNN's R2 was close to six times larger than the one of the linear regression model. This translated into a much better density prediction (with relative LPL above four). Finally, for the finance dataset we find no robust pattern. In 2008, the BNN did much better than the linear model but the R2 was slightly below one. This suggests a higher likelihood that out-of-sample gains do not stem from better point forecasts but are more likely to arise from higher order features of the predictive density.

Summing up this discussion, we find that in extreme cases (except for the finance

dataset), our BNNs perform better in terms of density forecasts and these improvements are often accompanied by a substantially larger in-sample fit. This is closely related to the benign overfitting phenomenon discussed in, e.g., [Bartlett et al. \(2020\)](#), which states that neural networks fit the data almost perfectly in-sample but also yield superior out-of-sample predictions.

## 5 Conclusion

In this paper we developed techniques to flexibly estimate shallow neural networks with many neurons and various types of activation functions in a Bayesian framework, for either sparse or dense datasets. Using shrinkage and selection priors allows us to determine the appropriate network structure. This includes not only selecting the relevant number of neurons but also different activation functions during a specifically designed and highly efficient MCMC sampling. As an additional technical improvement, we allow for heteroskedasticity in the shocks. The resulting BNNs are then applied to synthetic data. We show that they (as expected) improve upon linear models if the DGP is nonlinear but even if the DGP is linear, our techniques yield precise forecasts that are competitive to the ones of the linear model.

In our empirical application we apply the models to four different datasets commonly used in macroeconomics and finance, both cross-sectional and time series, and with different temporal frequency. We carry out forecasting experiments and show that different variants of the BNNs work well overall but in particular during extreme periods (or in the tails). A simple analysis based on in-sample  $R^2$ s and out-of-sample predictive likelihoods reveals that in these extreme cases, BNNs explain more in-sample variation and this is often accompanied by superior density forecasts. The possibility of using different activation functions and number of neurons over time / units in our enhanced BNNs also improves the forecasting performance.

These results are relevant not only for the forecasting literature but also for theoretical macro and finance, as they imply that nonlinearities and extreme events are pervasive and should be properly taken into account also in theoretical models.

## References

- AGOSTINELLI, F., M. HOFFMAN, P. SADOWSKI, AND P. BALDI (2014): “Learning activation functions to improve deep neural networks,” *arXiv preprint arXiv:1412.6830*.
- AI, C., AND E. C. NORTON (2003): “Interaction terms in logit and probit models,” *Economics Letters*, 80(1), 123–129.
- AMISANO, G., AND G. FAGAN (2013): “Money growth and inflation: A regime switching approach,” *Journal of International Money and Finance*, 33, 118–145.
- BARRO, R. J., AND J.-W. LEE (1994): “Sources of economic growth,” *Carnegie-Rochester Conference Series on Public Policy*, 40, 1–46.
- BARTLETT, P. L., P. M. LONG, G. LUGOSI, AND A. TSIGLER (2020): “Benign overfitting in linear regression,” *Proceedings of the National Academy of Sciences*, 117(48), 30063–30070.
- BECKMANN, J., G. KOOP, D. KOROBILIS, AND R. A. SCHÜSSLER (2020): “Exchange rate predictability and dynamic Bayesian learning,” *Journal of Applied Econometrics*, 35(4), 410–421.
- BELMONTE, M. A., G. KOOP, AND D. KOROBILIS (2014): “Hierarchical shrinkage in time-varying parameter models,” *Journal of Forecasting*, 33(1), 80–94.
- BHATTACHARYA, A., AND D. B. DUNSON (2011): “Sparse Bayesian infinite factor models,” *Biometrika*, 98(2), 291–306.
- BHATTACHARYA, A., D. PATI, N. S. PILLAI, AND D. B. DUNSON (2015): “Dirichlet–Laplace priors for optimal shrinkage,” *Journal of the American Statistical Association*, 110(512), 1479–1490.
- BLUNDELL, C., J. CORNEBISE, K. KAVUKCUOGLU, AND D. WIERSTRA (2015): “Weight uncertainty in neural network,” in *International Conference on Machine Learning*, pp. 1613–1622. PMLR.
- CAMPBELL, J. Y. (1987): “Stock returns and the term structure,” *Journal of Financial Economics*, 18(2), 373–399.
- CAMPBELL, J. Y., AND R. J. SHILLER (1988): “The dividend-price ratio and expectations of future dividends and discount factors,” *The Review of Financial Studies*, 1(3), 195–228.
- CAMPBELL, J. Y., AND T. VUOLTEENAHU (2004): “Inflation illusion and stock prices,” *American Economic Review*, 94(2), 19–23.
- CARRIERO, A., Y. BAI, T. CLARK, AND M. MARCELLINO (2022): “Macroeconomic Forecasting in a Multi-country Context,” *Journal of Applied Econometrics*, (forthcoming).
- CARVALHO, C. M., N. G. POLSON, AND J. G. SCOTT (2010): “The horseshoe estimator for sparse signals,” *Biometrika*, 97(2), 465–480.
- CHIPMAN, H. A., E. I. GEORGE, AND R. E. MCCULLOCH (2010): “BART: Bayesian additive regression trees,” *The Annals of Applied Statistics*, 4(1), 266–298.
- COULOMBE, P. G. (2020): “The macroeconomy as a random forest,” *arXiv preprint arXiv:2006.12724*.
- CRAWFORD, L., S. R. FLAXMAN, D. E. RUNCIE, AND M. WEST (2019): “Variable prioritization in nonlinear black box methods: A genetic association case study,” *The Annals of Applied Statistics*, 13(2), 958.
- CUI, T., A. HAVULINNA, P. MARTTINEN, AND S. KASKI (2021): “Informative Bayesian Neural Network Priors for Weak Signals,” *Bayesian Analysis*, 1(1), 1–31.
- DIEBOLD, F. X., AND R. S. MARIANO (1995): “Comparing Predictive Accuracy,” *Journal of Business & Economic Statistics*, 13(3), 253–263.
- DUSENBERRY, M., G. JERFEL, Y. WEN, Y. MA, J. SNOEK, K. HELLER, B. LAKSHMINARAYANAN, AND D. TRAN (2020): “Efficient and scalable bayesian neural nets with rank-1 factors,” in *International Conference on Machine Learning*, pp. 2782–2792. PMLR.
- ENGLE, R. F., AND J. G. RANGEL (2008): “The spline-GARCH model for low-frequency volatility and its global macroeconomic causes,” *The Review of Financial Studies*, 21(3), 1187–1222.
- ESCOBAR, M. D., AND M. WEST (1995): “Bayesian density estimation and inference using mixtures,” *Journal of the American Statistical Association*, 90(430), 577–588.
- FAMA, E. F., AND K. R. FRENCH (1988): “Dividend yields and expected stock returns,” *Journal of Financial Economics*, 22(1), 3–25.
- (1989): “Business conditions and expected returns on stocks and bonds,” *Journal of Financial Economics*, 25(1), 23–49.
- FAMA, E. F., AND G. W. SCHWERT (1977): “Asset returns and inflation,” *Journal of Financial Economics*, 5(2), 115–146.



- GAL, Y., AND Z. GHAHRAMANI (2016): “Dropout as a bayesian approximation: Representing model uncertainty in deep learning,” in *International Conference on Machine Learning*, pp. 1050–1059. PMLR.
- GALLEGATI, M. (2008): “Wavelet analysis of stock returns and aggregate economic activity,” *Computational Statistics & Data Analysis*, 52(6), 3061–3074.
- GEORGE, E. I., AND R. E. MCCULLOCH (1993): “Variable selection via Gibbs sampling,” *Journal of the American Statistical Association*, 88(423), 881–889.
- GEORGE, E. I., D. SUN, AND S. NI (2008): “Bayesian stochastic search for VAR model restrictions,” *Journal of Econometrics*, 142(1), 553–580.
- GHOSH, S., J. YAO, AND F. DOSHI-VELEZ (2019): “Model Selection in Bayesian Neural Networks via Horseshoe Priors,” *Journal of Machine Learning Research*, 20(182), 1–46.
- GIACOMINI, R., AND B. ROSSI (2010): “Forecast comparisons in unstable environments,” *Journal of Applied Econometrics*, 25(4), 595–620.
- GREENE, W. (2010): “Testing hypotheses about interaction terms in nonlinear models,” *Economics Letters*, 107(2), 291–296.
- GRIFFIN, J. E., AND P. J. BROWN (2013): “Some priors for sparse regression modelling,” *Bayesian Analysis*, 8(3), 691–702.
- HAMILTON, J. D. (1989): “A New Approach to the Economic Analysis of Nonstationary Time Series and the Business Cycle,” *Econometrica*, 57(2), 357–384.
- HARDING, M., J. LINDE, AND M. TRABANDT (2022): “Understanding Post-Covid Inflation Dynamics,” *IWH Working Paper*.
- HAUZENBERGER, N., F. HUBER, M. MARCELLINO, AND N. PETZ (2021): “Gaussian process vector autoregressions and macroeconomic uncertainty,” *arXiv preprint arXiv:2112.01995*.
- HODRICK, R. J. (1992): “Dividend yields and expected stock returns: Alternative procedures for inference and measurement,” *The Review of Financial Studies*, 5(3), 357–386.
- HORNIK, K. (1991): “Approximation capabilities of multilayer feedforward networks,” *Neural Networks*, 4(2), 251–257.
- HORNIK, K., M. STINCHCOMBE, AND H. WHITE (1989): “Multilayer feedforward networks are universal approximators,” *Neural Networks*, 2(5), 359–366.
- HUBER, F., G. KOOP, AND L. ONORANTE (2021): “Inducing Sparsity and Shrinkage in Time-Varying Parameter Models,” *Journal of Business & Economic Statistics*, 39(3), 669–683.
- HUBER, F., G. KOOP, L. ONORANTE, M. PFARRHOFER, AND J. SCHREINER (2020): “Nowcasting in a pandemic using non-parametric mixed frequency VARs,” *Journal of Econometrics*, (forthcoming).
- HUBER, F., AND M. PFARRHOFER (2021): “Dynamic shrinkage in time-varying parameter stochastic volatility in mean models,” *Journal of Applied Econometrics*, 36(2), 262–270.
- HUBER, F., AND T. O. ZÖRNER (2019): “Threshold cointegration in international exchange rates: A Bayesian approach,” *International Journal of Forecasting*, 35(2), 458–473.
- IMBENS, G. W., AND J. M. WOOLDRIDGE (2009): “Recent developments in the econometrics of program evaluation,” *Journal of Economic Literature*, 47(1), 5–86.
- JOHNDROW, J., P. ORENSTEIN, AND A. BHATTACHARYA (2020): “Scalable approximate MCMC algorithms for the horseshoe prior,” *Journal of Machine Learning Research*, 21(73), 1–61.
- KARLIK, B., AND A. V. OLGAC (2011): “Performance analysis of various activation functions in generalized MLP architectures of neural networks,” *International Journal of Artificial Intelligence and Expert Systems*, 1(4), 111–122.
- KASTNER, G., AND S. FRÜHWIRTH-SCHNATTER (2014): “Ancillarity-sufficiency interweaving strategy (ASIS) for boosting MCMC estimation of stochastic volatility models,” *Computational Statistics & Data Analysis*, 76, 408–423.
- KINGMA, D. P., AND M. WELLING (2013): “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*.
- KOWAL, D. R., D. S. MATTESON, AND D. RUPPERT (2019): “Dynamic shrinkage processes,” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 81(4), 781–804.
- LEE, D. S., AND T. LEMIEUX (2010): “Regression discontinuity designs in economics,” *Journal of Economic Literature*, 48(2), 281–355.
- LEWELLEN, J. (2004): “Predicting returns with financial ratios,” *Journal of Financial Economics*, 74(2), 209–235.
- MACKAY, D. J. (1992): “A practical Bayesian framework for backpropagation networks,” *Neural Computation*, 4(3), 448–472.
- MAKALIC, E., AND D. F. SCHMIDT (2015): “A simple sampler for the horseshoe estimator,”

- IEEE Signal Processing Letters*, 23(1), 179–182.
- MAKRIDAKIS S., SPILIOTIS E., A. V. (2018): “Statistical and machine learning forecasting methods: Concerns and ways forward,” *PLoS One*, 13.
- MARCELLINO, M., J. H. STOCK, AND M. W. WATSON (2006): “A comparison of direct and iterated multistep AR methods for forecasting macroeconomic time series,” *Journal of Econometrics*, 135(1-2), 499–526.
- MCCRACKEN, M. W., AND S. NG (2016): “FRED-MD: A monthly database for macroeconomic research,” *Journal of Business & Economic Statistics*, 34(4), 574–589.
- MCCRARY, J. (2008): “Manipulation of the running variable in the regression discontinuity design: A density test,” *Journal of Econometrics*, 142(2), 698–714.
- NEAL, R. M. (1996): “Priors for infinite networks,” in *Bayesian Learning for Neural Networks*, pp. 29–53. Springer.
- RAFTERY, A. E., AND S. LEWIS (1992): “How many iterations in the Gibbs sampler?,” in *Bayesian Statistics*, ed. by J. Bernardo, J. Berger, A. Dawid, and A. Smith, vol. 4, pp. 763–773, Oxford, UK. Oxford University Press.
- RAMSEY, J. B., AND C. LAMPART (1998): “The decomposition of economic relationships by time scale using wavelets: expenditure and income,” *Studies in Nonlinear Dynamics & Econometrics*, 3(1).
- REICHLIN, L., AND M. LENZA (2007): “On short-term and long-term causality of money to inflation: understanding the problem and clarifying some conceptual issues,” Discussion paper, Mimeo.
- ROBERTS, G. O., AND J. S. ROSENTHAL (2009): “Examples of adaptive MCMC,” *Journal of Computational and Graphical Statistics*, 18(2), 349–367.
- ROSSI, B. (2013): “Exchange rate predictability,” *Journal of Economic Literature*, 51(4), 1063–1119.
- SCARDAPANE, S., D. COMMINIELLO, A. HUSSAIN, AND A. UNCINI (2017): “Group sparse regularization for deep neural networks,” *Neurocomputing*, 241, 81–89.
- SEZER O., OZBAYOGLU M., D. E. (2020): “Financial time series forecasting with deep learning: A systematic literature review: 2005–2019,” *arXiv preprint arxiv:abs/1911.13288*.
- STOCK, J., AND M. WATSON (1999): “Forecasting inflation,” *Journal of Monetary Economics*, 44(2), 293–335.
- TERÄSVIRTA, T. (1994): “Specification, estimation, and evaluation of smooth transition autoregressive models,” *Journal of the American Statistical Association*, 89(425), 208–218.
- TONG, H. (1990): *Non-linear time series: a dynamical system approach*. Oxford University Press.
- VASICEK, O. A., AND H. G. FONG (1982): “Term structure modeling using exponential splines,” *The Journal of Finance*, 37(2), 339–348.
- WELCH, I., AND A. GOYAL (2008): “A comprehensive look at the empirical performance of equity premium prediction,” *The Review of Financial Studies*, 21(4), 1455–1508.
- WILLIAMS, C. K., AND C. E. RASMUSSEN (2006): *Gaussian processes for machine learning*, vol. 2. MIT Press Cambridge, MA.
- WRIGHT, J. H. (2008): “Bayesian model averaging and exchange rate forecasts,” *Journal of Econometrics*, 146(2), 329–341.

# Appendices

## A Technical appendix

### A.1 Full conditional posterior distributions

In the following, we provide details on the full conditional posterior distribution for the proposed Markov chain Monte Carlo (MCMC) algorithm outlined in Section 2.4.

- Let  $\boldsymbol{\theta} = (\boldsymbol{\gamma}', \boldsymbol{\beta}')'$  denote a  $(K+Q)$ -dimensional vector of parameters and  $\tilde{\mathbf{x}} = (\tilde{\mathbf{x}}'_1, \dots, \tilde{\mathbf{x}}'_T)'$  a  $(K+Q) \times T$  matrix of neurons with element  $\tilde{\mathbf{x}}_t = (\mathbf{x}'_t, h_1(\mathbf{x}'_t \boldsymbol{\kappa}_1 + \zeta_1), \dots, h_Q(\mathbf{x}'_t \boldsymbol{\kappa}_Q + \zeta_Q))'$ . Moreover, we define  $\boldsymbol{\Sigma} = \text{diag}(\sigma_1^2, \dots, \sigma_T^2)$  as a  $T \times T$  matrix capturing the variances and  $\underline{\mathbf{V}}_{\boldsymbol{\theta}} = \text{diag}(\boldsymbol{\phi}_{\boldsymbol{\gamma}}^{-1}, \boldsymbol{\phi}_{\boldsymbol{\beta}}^{-1})$  where  $\boldsymbol{\phi}_{\boldsymbol{\gamma}}^{-1} = (\phi_{\gamma_1}^{-1}, \dots, \phi_{\gamma_K}^{-1})'$  denotes the  $K$  prior variances for the constant coefficients and we collect  $\boldsymbol{\phi}_{\boldsymbol{\beta}}^{-1} = (\phi_{\beta_1}^{-1}, \dots, \phi_{\beta_Q}^{-1})'$  for the nonlinear coefficients. The joint parameter vector  $\boldsymbol{\theta}$  is then obtained from a standard multivariate Gaussian posterior:

$$\boldsymbol{\theta} | \bullet \sim \mathcal{N}(\bar{\boldsymbol{\theta}}, \bar{\mathbf{V}}_{\boldsymbol{\theta}}), \quad (\text{A.1})$$

with

$$\begin{aligned} \bar{\mathbf{V}}_{\boldsymbol{\theta}} &= (\tilde{\mathbf{x}}' \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{x}} + \underline{\mathbf{V}}_{\boldsymbol{\theta}}^{-1})^{-1}, \\ \bar{\boldsymbol{\theta}} &= \bar{\mathbf{V}}_{\boldsymbol{\theta}} \tilde{\mathbf{x}}' \boldsymbol{\Sigma}^{-1} \mathbf{y}. \end{aligned}$$

- The prior on  $\boldsymbol{\gamma}$  is Normal of the form:

$$\gamma_j \sim \mathcal{N}(0, \phi_{\gamma_j}^{-1}), \quad \phi_{\gamma_j}^{-1} = \lambda_{\boldsymbol{\gamma}}^2 \varphi_{\gamma_j}^2, \quad \text{for } j = 1, \dots, K. \quad (\text{A.2})$$

We use a horseshoe prior and rely on the hierarchical representation of [Makalic and Schmidt \(2015\)](#). The global and local shrinkage parameters,  $\lambda_{\boldsymbol{\gamma}}^2$  and  $\varphi_{\gamma_j}^2$ , respectively, are obtained by introducing auxiliary random quantities which follow an inverse Gamma

distribution:

$$\varphi_{\gamma_j}^2 | \bullet \sim \mathcal{G}^{-1} \left( 1, c_{\gamma_j}^{-1} + \frac{\gamma_j^2}{2\lambda_{\gamma_j}^2} \right), \quad (\text{A.3})$$

$$\lambda_{\gamma}^2 | \bullet \sim \mathcal{G}^{-1} \left( \frac{K+1}{2}, d_{\gamma}^{-1} + \sum_{j=1}^K \frac{\gamma_j^2}{2\varphi_{\gamma_j}^2} \right), \quad (\text{A.4})$$

$$c_{\gamma_j} | \bullet \sim \mathcal{G}^{-1} \left( 1, 1 + \varphi_{\gamma_j}^{-2} \right), \quad (\text{A.5})$$

$$d_{\gamma} | \bullet \sim \mathcal{G}^{-1} \left( 1, 1 + \lambda_{\gamma}^{-2} \right). \quad (\text{A.6})$$

- We sample the hyperparameters associated with the MGP prior on  $\beta$  from inverse Gamma distributions:

$$\delta_1 \sim \mathcal{G}^{-1} \left( a_1 + \frac{Q}{2}, 1 + \frac{1}{2} \sum_{q=1}^Q (\phi_{\beta_q} \beta_q^2) \right), \quad (\text{A.7})$$

$$\delta_r \sim \mathcal{G}^{-1} \left( a_2 + \frac{Q-r-1}{2}, 1 + \frac{1}{2} \sum_{q=1}^Q (\phi_{\beta_q} \beta_q^2) \right). \quad (\text{A.8})$$

- To draw  $\kappa_q$  ( $q = 1, \dots, Q$ ), we use a random walk MH step with the following proposal distribution:

$$\log(\kappa_q^*) = \log(\kappa_q^{(a)}) + \sigma_{\kappa_q} \varphi, \quad \varphi \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad (\text{A.9})$$

where  $\sigma_{\kappa_q}$  is a scaling parameter and adjusted in a way such that the acceptance rate of the MH algorithm is between 20 and 40 percent (Roberts and Rosenthal, 2009). We evaluate the proposed and previously accepted values (defined as  $\kappa_q^*$  and  $\kappa_q^{(a)}$ , respectively) by combining the conditional data likelihood  $\mathcal{L}(\kappa_q | \bullet)$  with the corresponding MGP prior density  $p(\kappa_q)$  to obtain the posterior likelihood. Since the proposal in Eq. (A.9) is asymmetric, in addition, we need to account for a proposal correction term  $\mathcal{J}(\kappa_q)$ . The acceptance probability  $\eta_q$  is then defined by the inverse ratio of the posterior likelihood of  $\kappa_q^*$  and the posterior likelihood  $\kappa_q^{(a)}$  and the ratio of the corresponding correction terms. Formally, this is given by

$$\eta_q = \min \left( 1, \frac{\mathcal{L}(\kappa_q^* | \bullet) p(\kappa_q^*)}{\mathcal{L}(\kappa_q^{(a)} | \bullet) p(\kappa_q^{(a)})} \frac{\mathcal{J}(\kappa_q^{(a)})}{\mathcal{J}(\kappa_q^*)} \right), \quad (\text{A.10})$$

with

$$\begin{aligned}\mathcal{L}(\boldsymbol{\kappa}_q|\bullet) &\propto \exp \left\{ -\frac{1}{2} \left( (\hat{\mathbf{y}}_q - \boldsymbol{\mu}_q)' \boldsymbol{\Sigma}^{-1} (\hat{\mathbf{y}}_q - \boldsymbol{\mu}_q) \right) \right\}, \\ p(\boldsymbol{\kappa}_q) &\propto \exp \left\{ -\frac{1}{2} \left( \boldsymbol{\kappa}_q' \mathbf{V}_{\boldsymbol{\kappa}_q}^{-1} \boldsymbol{\kappa}_q \right) \right\},\end{aligned}$$

where  $\mathbf{V}_{\boldsymbol{\kappa}_q}^{-1} = \text{diag}(\phi_{\kappa_{1q}}, \dots, \phi_{\kappa_{Kq}})$ ,  $\boldsymbol{\mu}_q = (\mu_{1q}, \dots, \mu_{Tq})'$  with elements  $\mu_{tq} = \beta_q h_q(\mathbf{x}_t' \boldsymbol{\kappa}_q + \zeta_q)$  and  $\hat{\mathbf{y}}_q = \mathbf{y} - \mathbf{x}_t' \boldsymbol{\gamma} - \sum_{k \neq q} \beta_q h_q(\mathbf{x}_t' \boldsymbol{\kappa}_k + \zeta_k)$ .

- To achieve shrinkage in the neurons we apply a column-wise horseshoe prior on the elements of  $\boldsymbol{\kappa}_q$ . We follow [Makalic and Schmidt \(2015\)](#) and define auxiliary random quantities, which are used to obtain the global and local shrinkage parameters,  $\lambda_{\boldsymbol{\kappa}_q}^2$  and  $\varphi_{\kappa_{jq}}^2$ . The (hyper)parameters follow an inverse Gamma distribution:

$$\varphi_{\kappa_{jq}}^2 | \bullet \sim \mathcal{G}^{-1} \left( 1, c_{\kappa_{jq}}^{-1} + \frac{\kappa_{jq}^2}{2\lambda_{\boldsymbol{\kappa}_q}^2} \right), \quad (\text{A.11})$$

$$\lambda_{\boldsymbol{\kappa}_q}^2 | \bullet \sim \mathcal{G}^{-1} \left( \frac{K+1}{2}, d_{\boldsymbol{\kappa}_q}^{-1} + \sum_{j=1}^K \frac{\kappa_{jq}^2}{2\varphi_{\kappa_{jq}}^2} \right), \quad (\text{A.12})$$

$$c_{\kappa_{jq}} | \bullet \sim \mathcal{G}^{-1} \left( 1, 1 + \varphi_{\kappa_{jq}}^{-2} \right), \quad (\text{A.13})$$

$$d_{\boldsymbol{\kappa}_q} | \bullet \sim \mathcal{G}^{-1} \left( 1, 1 + \lambda_{\boldsymbol{\kappa}_q}^{-2} \right). \quad (\text{A.14})$$

- For choosing the activation function  $h_q$ , we draw the indicator  $\delta_q$  from a multinomial distribution of the following form:

$$\Pr(\delta_q = m | \bullet) \propto \omega_{qm} \times \exp \left\{ -\frac{1}{2} \left( (\hat{\mathbf{y}}_q - \boldsymbol{\mu}_q^{(m)})' \boldsymbol{\Sigma}^{-1} (\hat{\mathbf{y}}_q - \boldsymbol{\mu}_q^{(m)}) \right) \right\}, \quad \text{for } m = 1, \dots, 6, \quad (\text{A.15})$$

where  $\boldsymbol{\mu}_q^{(m)} = (\mu_{1q}^{(m)}, \dots, \mu_{Tq}^{(m)})'$  with elements  $\mu_{tq}^{(m)} = \beta_q h_q^{(m)}(\mathbf{x}_t' \boldsymbol{\kappa}_q + \zeta_q)$ .

## A.2 Bayesian neural network by backpropagation (BNN-BP)

A Bayesian neural network estimated by backpropagation was introduced by [Blundell et al. \(2015\)](#). It serves as a variational inference scheme for learning the posterior distribution on the weights of a neural network. To do so, the approach maximizes the log-likelihood of the model subject to a Kullback-Leibler complexity term on the parameters and makes use of the reparameterization trick ([Kingma and Welling, 2013](#)) to obtain the posterior distribution of the weights with stochastic gradient descent.

The prior on the weights is specified as a scale mixture of two Gaussian densities with zero mean but differing variances. The first mixture component features a large variance ( $\sigma_1^2 = 3$ ) providing a heavy-tailed distribution whereas the variance of the other component is set small ( $\sigma_2^2 = 0.0025$ ) concentrating the weights a priori around zero. This setup is similar to a spike and slab prior (see, [George and McCulloch, 1993](#)) but with the same prior parameters for all the weights to allow for the optimization by stochastic gradient descent.

The hyperparameters are chosen in a cross validation exercise. For the cross-sectional datasets (i.e., Macro B and synthetic) we randomly split the data into equally sized training and test sets. We evaluate each model specification in 20 replications and use those yielding the lowest average RMSE for the final model. For the time series applications (i.e., Macro A, Macro C and Finance) we use a cross validation based on an expanding window time series split. Specifically, we use all observations up to the last 24 months for Macro A, 12 quarters for Macro C and 10 years for Finance before the start of our hold-out to train the model and then, after obtaining the predictive densities, add the next observation and recompute the model. We repeat this until we end up at the beginning of our hold-out and choose the specification with the lowest average RMSE. We train all models in 1000 epochs and use the MSE loss function, the ADAM optimizer and a learning rate of 0.01.

### A.3 Bayesian additive regression trees (BART)

Bayesian additive regression trees ([Chipman et al., 2010](#)) are an alternative to BNNs to approximate the unknown function  $f$ . The idea is to consider the sum over a number of  $Z$  regression trees. Formally, this boils down to:

$$f(\mathbf{x}_t) \approx \sum_{z=1}^Z g_z(\mathbf{x}_t | \mathcal{T}_z, \boldsymbol{\rho}_z) \quad (\text{A.16})$$

A single regression tree  $g_z$  depends on two parameters, the tree structure given by  $\mathcal{T}_z$  and the terminal node parameter  $\boldsymbol{\rho}_z$ . Following [Chipman et al. \(2010\)](#), we set  $Z = 250$  and build the prior on the tree structure upon a tree-generating stochastic process. This involves determining the probability that a given node is nonterminal, the selection of variables used in a splitting rule (to spawn left and right children nodes) and the corresponding thresholds. For the terminal node parameter we specify a conjugate Gaussian prior distribution with data-based prior variance. In particular, the specification centers prior mass on the range of the data while ensuring a higher degree of shrinkage if the number of trees is large. Details can be found in [Chipman et al. \(2010\)](#).

## B Empirical appendix

### B.1 Details on the data

**Table B.1:** Full set of empirical applications.

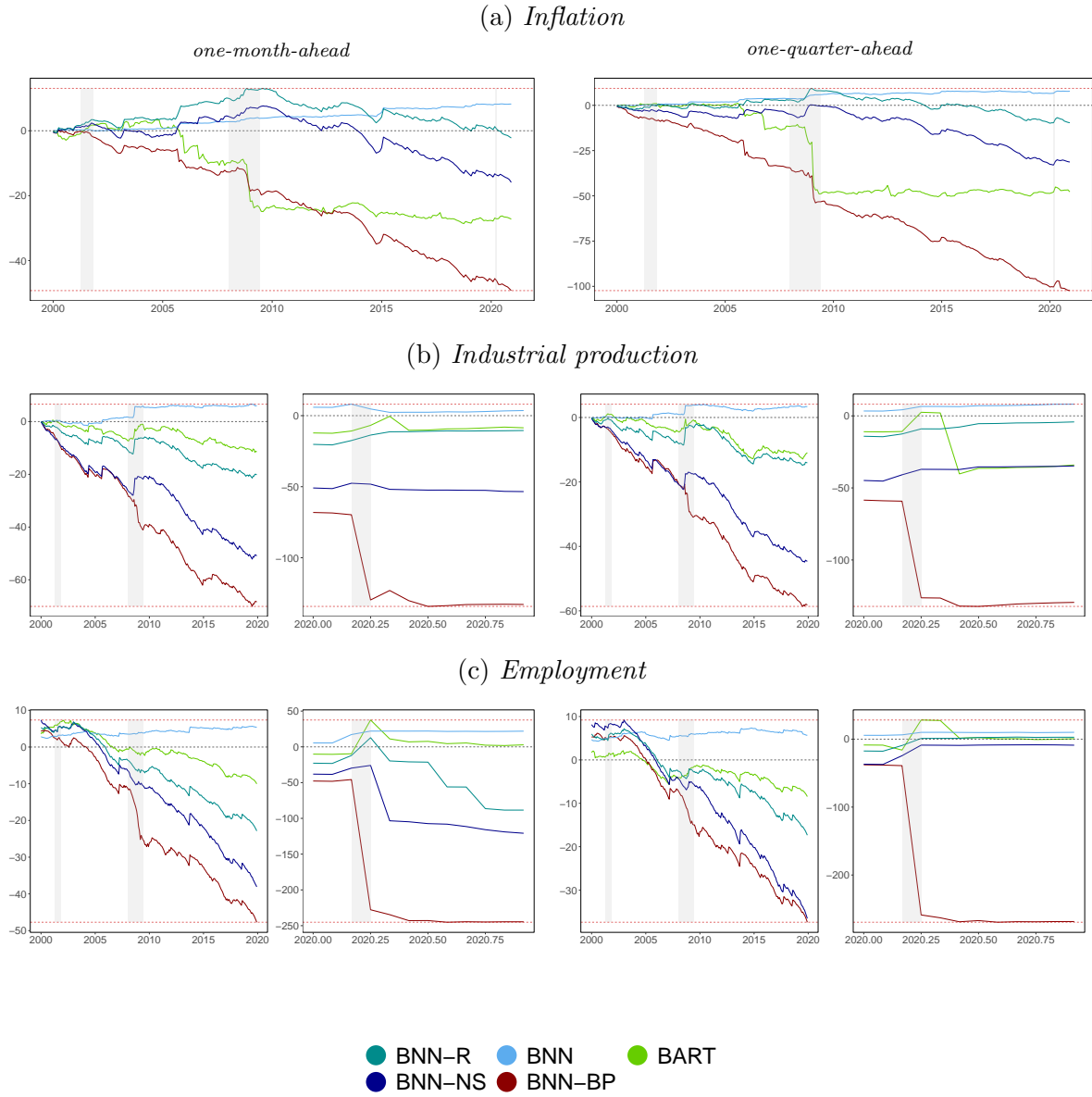
	Dependent variable	Set of predictors	Sample	Range	Horizon	Hold-out	Source
<b>Macro A</b>	A.1) Industrial production A.2) Inflation A.3) Employment	1) Medium (20 economic & financial variables) 2) Large (120 economic & financial variables) 3) PCA	Monthly data for the US	1960M1 to 2020M12	one-step- and three-steps-ahead	2000M1 to 2020M12	McCracken and Ng (2016)
<b>Macro B</b>	Average economic growth rate	60 country-specific characteristics	Cross-section	90 countries	100 random samples	45 countries	Barro and Lee (1994)
<b>Macro C</b>	USD/GBP exchange rate returns (qoq)	1) 20 exchange rate determinants 2) Interest rate differential 3) Inflation differential 4) Monetary fundamentals 5) Taylor rule differential 6) All fundamentals	Quarterly data for the US and UK	1990Q1 to 2019Q4	one-step- and four-steps-ahead	2000Q1 to 2019Q4	Wright (2008); Rossi (2013)
<b>Finance</b>	Equity premium (i.e., S&P 500 excess returns)	1) 16 economic & financial variables 2) Inflation 3) Term spread 4) Dividend yield 5) Dividend price ratio	Annual data for the US	1948 to 2020	one-year-ahead	1965 to 2020	Welch and Goyal (2008)

*Note:* The table gives an overview of the different empirical applications with which we test our proposed Bayesian neural network approach. We evaluate the performance of the model approach in each application through root mean squared errors (RMSEs) for point forecasts and log predictive likelihoods (LPLs) for density forecasts.

### B.2 Overall forecasting results

In this sub-section we provide additional results on our thorough forecasting exercise for the four empirical applications. We present detailed point and density forecasting performance of the different models compared to the linear model measured by relative root mean squared errors (RMSEs) and log predictive likelihoods (LPLs), respectively, for each application and forecasting horizon in Table B.2 to Table B.6. Moreover, we show cumulative LPLs over the hold-out for each forecasting horizon and application in Figure B.1 to Figure B.4. Note that for the cross-sectional example, Macro B in Figure B.2, we plot log predictive likelihoods against the linear model for the two best performing models, i.e., the BNN-NS and BART.

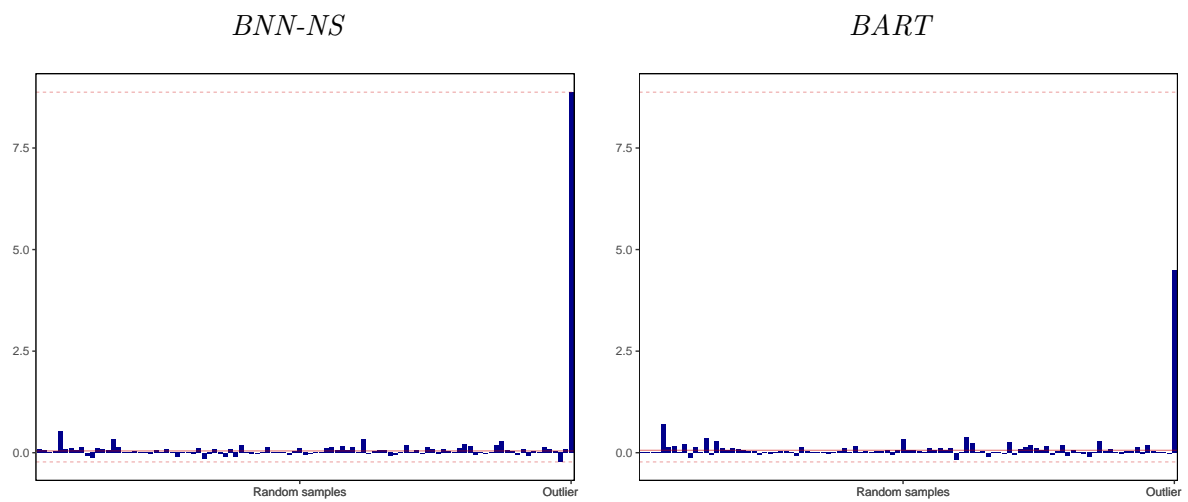
**Figure B.1: Macro A.** Evolution of cumulative LPLs against the benchmark (large covariate set).



*Note:* This figure shows cumulative log predictive likelihoods (LPLs) against the benchmark for each specification. Here, we choose the linear model of each specification as our benchmark to highlight the effect of controlling for nonlinearities. Note that this is in contrast to the tables where we choose a global benchmark for each application. The red dashed lines denote the max./min. LPLs at the end of the hold-out sample, while the gray shaded areas indicate the NBER recessions.



**Figure B.2: Macro B. LPLs against the benchmark.**



*Note:* This figures shows relative log predictive likelihoods (LPLs) against the linear model. The red dashed lines denote the max./min. LPLs and the red solid line indicates the mean.

**Table B.2: Macro A.** Forecast performance across 252 hold-out observations (one-month-ahead).

Covariates	Model					
	BART	BNN	BNN-NS	BNN-R	BNN-BP	Linear model
Inflation						
AR(1) (sv, constant)		1.05 (0.03)	1.05 (0.02)	1.07** (0.03)		<b>1.05</b> (-0.03)
AR(1) (sv)	1.11*** (-0.22***)	1.04 (-0.01)	1.05 (-0.01)	1.03 (-0.01)	<b>1.03</b> (-0.19***)	
Large (sv, constant)		0.94*** (0.11***)	0.94*** (0.02)	<b>0.93*</b> (0.07**)		0.94*** (0.08***)
Large (sv)	0.97 (-0.03)	1.03 (0.01)	1.04** (-0.04)	<b>0.93*</b> (0.08**)	1.03 (-0.12***)	
Medium (sv, constant)		<b>0.97</b> (0.08***)	0.98 (0.03)	0.98 (0.06**)		0.98 (0.06***)
Medium (sv)	1.04 (-0.10*)	0.99 (0.02)	0.99 (0.01)	<b>0.97</b> (0.07**)	1.04** (-0.12***)	
PCA (sv, constant)		<b>1.01</b> (0.02**)	1.02* (0.01)	1.02 (0.04*)		1.16 (-1.45)
PCA (sv)	1.07** (-0.20***)	1.02 (0.00)	1.02 (0.02)	<b>1.02</b> (0.04*)	1.04* (-0.11***)	
Industrial production						
AR(1) (sv, constant)		<b>0.90</b> (0.11**)	0.91 (0.13**)	0.92 (0.14**)		0.91 (0.03)
AR(1) (sv)	<b>0.89</b> (-0.12)	0.89 (0.06)	0.90 (0.06)	0.93 (0.03)	0.92 (-0.81)	
Large (sv, constant)		1.06** (0.13***)	1.32 (-0.10**)	1.09 (0.07)		<b>0.98***</b> (0.12***)
Large (sv)	<b>0.88</b> (0.08)	0.95 (0.04)	1.33* (-0.08*)	0.98*** (0.07)	0.93 (-0.41*)	
Medium (sv, constant)		1.00** (0.11***)	1.05 (0.00)	1.38 (0.05)		<b>0.99**</b> (0.06***)
Medium (sv)	<b>0.88</b> (0.03)	0.92 (0.06)	1.14* (-0.05)	1.27 (0.07**)	0.93 (-0.41*)	
PCA (sv, constant)		<b>1.04</b> (0.05***)	1.26 (-0.02)	1.17 (0.03)		1.75 (-1.33)
PCA (sv)	0.89 (-0.01)	<b>0.88</b> (0.06)	1.11* (0.01)	0.91 (0.07)	0.94 (-0.46)	
Employment						
AR(1) (sv, constant)		1.02* (0.21)	1.13 (-0.19)	<b>1.01</b> (0.16*)		1.02 (-0.07*)
AR(1) (sv)	1.14** (-0.67)	1.12 (0.24)	<b>1.01</b> (0.23)	1.02 (0.21*)	1.01 (-0.96)	
Large (sv, constant)		<b>1.01</b> (0.18**)	1.36*** (-0.38)	1.45** (-0.25)		1.01 (0.10*)
Large (sv)	1.04 (0.11)	1.13 (-0.33)	1.37*** (-0.33)	1.61*** (-0.94**)	<b>1.01</b> (-0.87)	
Medium (sv, constant)		1.01 (0.12*)	1.18 (-0.27)	1.35* (-0.07)		<b>1.00</b> (0.04)
Medium (sv)	1.05 (-0.28)	1.01 (0.02)	1.30*** (-0.29)	1.55*** (-0.46)	<b>1.01</b> (-0.97)	
PCA (sv, constant)		1.01 (0.06)	1.33*** (0.05)	<b>0.99</b> (0.09)		3.50 (-1.88)
PCA (sv)	1.03 (-0.19*)	1.01 (0.03)	1.48*** (0.01)	<b>1.00</b> (0.09)	1.01 (-0.59)	

*Note:* The table shows root mean squared errors (RMSEs), and average log predictive likelihoods (LPLs) in parentheses, relative to the linear benchmark. In bold we mark the best performing model for each case. The grey shaded area gives the actual RMSE and LPL scores of our benchmark (linear model). The red shaded area marks the best performing model in terms of LPL which we analyze in more detail. Asterisks indicate statistical significance by means of the Diebold and Mariano (1995) test for each model relative to the benchmark at the 1% (\*\*\*), 5% (\*\*) and 10% (\*) significance levels. Results are averaged across the hold-out.

**Table B.3: Macro A.** Forecast performance across 252 hold-out observations (one-quarter-ahead).

Covariates	Model					
	BART	BNN	BNN-NS	BNN-R	BNN-BP	Linear model
Inflation						
AR(1) (sv, constant)		1.08*	<b>1.08*</b>	1.08*		1.08**
AR(1) (sv)	<b>1.08**</b> (-0.28***)	(-0.05) 1.09** (-0.09***)	(-0.05) 1.09* <b>(-0.09***)</b>	(-0.04) 1.10** (-0.09***)		(-0.10***)
Large (sv, constant)		0.90*** <b>(0.15***)</b>	0.96 (-0.01)	<b>0.90**</b> (0.08*)	1.13*** (-0.39***)	0.90*** (0.12***)
Large (sv)	<b>0.93***</b> (-0.07)	1.18*** (-0.15***)	1.13*** (-0.09**)	0.93 <b>(0.07)</b>	1.17*** (-0.29***)	
Medium (sv, constant)		0.98 <b>(0.06***)</b>	1.00 (0.01)	1.02 (0.06*)		<b>0.98</b> (0.04**)
Medium (sv)	1.03 (-0.16)	<b>1.03</b> (0.01)	1.06 (-0.05)	1.03 <b>(0.05)</b>	1.17*** (-0.30***)	
PCA (sv, constant)		1.02 (0.03***)	<b>1.01*</b> (0.02)	1.03** <b>(0.04)</b>		1.16 (-1.40)
PCA (sv)	1.11** (-0.31)	1.03 (-0.01)	1.07 (0.00)	<b>1.02</b> <b>(0.03)</b>	1.18*** (-0.31***)	
Industrial production						
AR(1) (sv, constant)		0.95 (0.14)	0.95 <b>(0.15)</b>	<b>0.95</b> (0.15)		0.96 (0.02)
AR(1) (sv)	0.96 (-0.14*)	0.93 (0.05)	0.93 (0.04)	<b>0.93</b> <b>(0.05)</b>	0.93 (-0.86)	
Large (sv, constant)		0.97 <b>(0.06**)</b>	0.98 (-0.11**)	<b>0.94</b> (0.01)		0.98 (0.02*)
Large (sv)	0.96 (-0.11)	0.94 <b>(0.04)</b>	0.93 (-0.07)	0.94 (0.03)	<b>0.93</b> (-0.49*)	
Medium (sv, constant)		<b>0.94*</b> <b>(0.09***)</b>	0.95 (-0.03)	0.94* (0.04)		0.96** (0.05***)
Medium (sv)	0.95 (-0.11***)	0.93 <b>(0.05)</b>	0.94 (-0.03)	0.96* (0.04)	<b>0.92</b> (-0.49)	
PCA (sv, constant)		1.00 (0.03*)	0.96** (0.02)	<b>0.94</b> <b>(0.04)</b>		1.76 (-1.32)
PCA (sv)	0.96 (-0.09***)	0.93 <b>(0.06**)</b>	<b>0.92**</b> (0.04)	0.94 (0.06)	0.93 (-0.40*)	
Employment						
AR(1) (sv, constant)		1.00 (0.12)	<b>1.00</b> (0.19)	1.00 <b>(0.39)</b>		1.00 (-0.06*)
AR(1) (sv)	1.00*** (-0.12***)	1.00 (0.02)	1.00 (0.02)	<b>1.00</b> <b>(0.02)</b>	1.00 (-0.40)	
Large (sv, constant)		1.00 <b>(0.05*)</b>	1.00*** (-0.03)	<b>0.99</b> (0.02)		1.00 (0.01)
Large (sv)	1.01 (0.01)	1.00 <b>(0.04)</b>	1.00*** (-0.04)	<b>0.99</b> (0.02)	1.00 (-1.06)	
Medium (sv, constant)		<b>1.00</b> (0.05*)	1.01** (0.02)	1.00 <b>(0.06)</b>		1.01 (-0.04)
Medium (sv)	<b>0.99</b> (-0.21)	1.00 (0.01)	1.00 (-0.01)	1.00 <b>(0.07)</b>	1.00 (-0.93)	
PCA (sv, constant)		1.00 (0.03)	1.00** <b>(0.10)</b>	<b>1.00</b> (0.05)		3.55 (-1.89)
PCA (sv)	<b>0.99</b> (-0.16*)	0.99 (0.04)	1.00*** (0.06)	0.99 <b>(0.09)</b>	1.00 (-1.63)	

*Note:* The table shows root mean squared errors (RMSEs), and average log predictive likelihoods (LPLs) in parentheses, relative to the linear benchmark. In bold we mark the best performing model for each case. The grey shaded area gives the actual RMSE and LPL scores of our benchmark (linear model). Asterisks indicate statistical significance by means of the Diebold and Mariano (1995) test for each model relative to the benchmark at the 1% (\*\*\*) , 5% (\*\*) and 10% (\*) significance levels. Results are averaged across the hold-out.

**Table B.4: Macro B.** Forecast performance across 45 hold-out countries and 100 replications.

Covariates	Model					
	BART	BNN	BNN-NS	BNN-R	BNN-BP	Linear model
Kitchen sink	<b>0.95***</b> <b>(0.06***)</b>	1.06*** (-0.20***)	1.01 (0.03**)	0.97*** (0.04***)	1.04*** (-0.18***)	
Kitchen sink (constant)		1.10*** (-0.15***)	<b>1.00</b> <b>(0.04***)</b>	<b>0.99**</b> (0.03***)		1.04 (-1.47)

*Note:* The table shows root mean squared errors (RMSEs), and average log predictive likelihoods (LPLs) in parentheses, relative to the linear benchmark. In bold we mark the best performing model for each case. The grey shaded area gives the actual RMSE and LPL scores of our benchmark (linear model). The red shaded area marks the best performing model in terms of LPL which we analyze in more detail. Asterisks indicate statistical significance by means of the Diebold and Mariano (1995) test for each model relative to the benchmark at the 1% (\*\*\*) , 5% (\*\*) and 10% (\*) significance levels. Results are averaged across the hold-out.

**Table B.5: Macro C. Forecast performance across 80 hold-out observations.**

Covariates	Multivariate models					
	BART	BNN	BNN-NS	BNN-R	BNN-BP	Linear model
one-quarter-ahead						
All fundamentals	1.03 (-0.02)	<b>1.01</b> ( <b>0.06</b> )	1.02 (0.05)	1.02 (0.04)	1.03 (-0.09*)	
All fundamentals (const)		1.02 (0.05)	1.03 ( <b>0.06</b> )	1.03 (0.06)		<b>1.01</b> (-0.01)
Kitchen sink	1.04 (-0.01)	1.01 ( <b>0.05</b> )	<b>0.99</b> (0.01)	1.00 (0.05)	1.02 (-0.10)	
Kitchen sink (const)		0.99 (0.04)	0.98 (0.02)	<b>0.99</b> ( <b>0.04</b> )		<b>0.95</b> (-1.29)
one-year-ahead						
All fundamentals	1.04 (0.00)	0.99 (0.05)	<b>0.99</b> ( <b>0.06</b> )	1.00 (0.06)	1.01 (-0.09**)	
All fundamentals (const)		<b>0.99</b> (0.05)	1.00 ( <b>0.07</b> )	1.00 (0.06)		1.00 (-0.01)
Kitchen sink	1.03 ( <b>0.06</b> )	0.99 (0.05)	1.02** (0.01)	1.02* (0.02)	<b>0.98**</b> (-0.09)	
Kitchen sink (const)		0.99 ( <b>0.04</b> )	1.04*** (0.00)	1.03** (0.02)		<b>0.96</b> (-1.29)
Univariate models						
	BART	BNN-NS		BNN-BP	Linear model	
one-quarter-ahead						
AR(1)	<b>0.97</b> (0.04)		0.98** ( <b>0.07</b> )	0.99 (-0.06)		
AR(1) (constant)			<b>0.96***</b> ( <b>0.11</b> )			0.96*** (0.05***)
Inflation differential	1.07 (-0.01)		1.02 ( <b>0.06</b> )	<b>1.01</b> (-0.07)		
Inflation differential (constant)			1.03 ( <b>0.05</b> )			<b>1.02</b> (0.00)
IR differential	<b>0.99</b> (0.00)		1.01 ( <b>0.07</b> )	1.02 (-0.08*)		
IR differential (constant)			1.02 ( <b>0.05</b> )			<b>1.00</b> (0.00)
Monetary fundamentals	1.06 (-0.04)		1.01 ( <b>0.07</b> )	<b>1.00</b> (-0.07)		
Monetary fundamentals (constant)			1.01 ( <b>0.06</b> )			<b>1.01</b> (0.01)
Taylor rule differential	1.09 (-0.02)		1.02 ( <b>0.06</b> )	<b>1.01</b> (-0.08*)		
Taylor rule differential (constant)			1.02 ( <b>0.07</b> )			<b>1.01</b> (0.00)
one-year-ahead						
AR(1)	1.02 (-0.01)		1.00 ( <b>0.06</b> )	<b>0.99</b> (-0.07**)		
AR(1) (constant)			1.00 ( <b>0.07</b> )			<b>0.99</b> (0.00)
Inflation differential	1.03 (0.00)		<b>0.99</b> ( <b>0.08</b> )	0.99 (-0.07**)		
Inflation differential (constant)			<b>0.99</b> ( <b>0.08</b> )			0.99 (0.00)
IR differential	1.04 (0.01)		<b>0.99</b> ( <b>0.08</b> )	1.01 (-0.08**)		
IR differential (constant)			<b>0.99</b> ( <b>0.06</b> )			0.99 (-0.01)
Monetary fundamentals	1.00 (0.01)		<b>0.99</b> ( <b>0.07</b> )	1.00 (-0.08**)		
Monetary fundamentals (constant)			<b>0.99*</b> ( <b>0.07</b> )			0.99 (0.00)
Taylor rule differential	1.06 (-0.02)		<b>0.99</b> ( <b>0.08</b> )	1.01 (-0.09***)		
Taylor rule differential (constant)			1.00 ( <b>0.08</b> )			<b>0.99</b> (0.00)

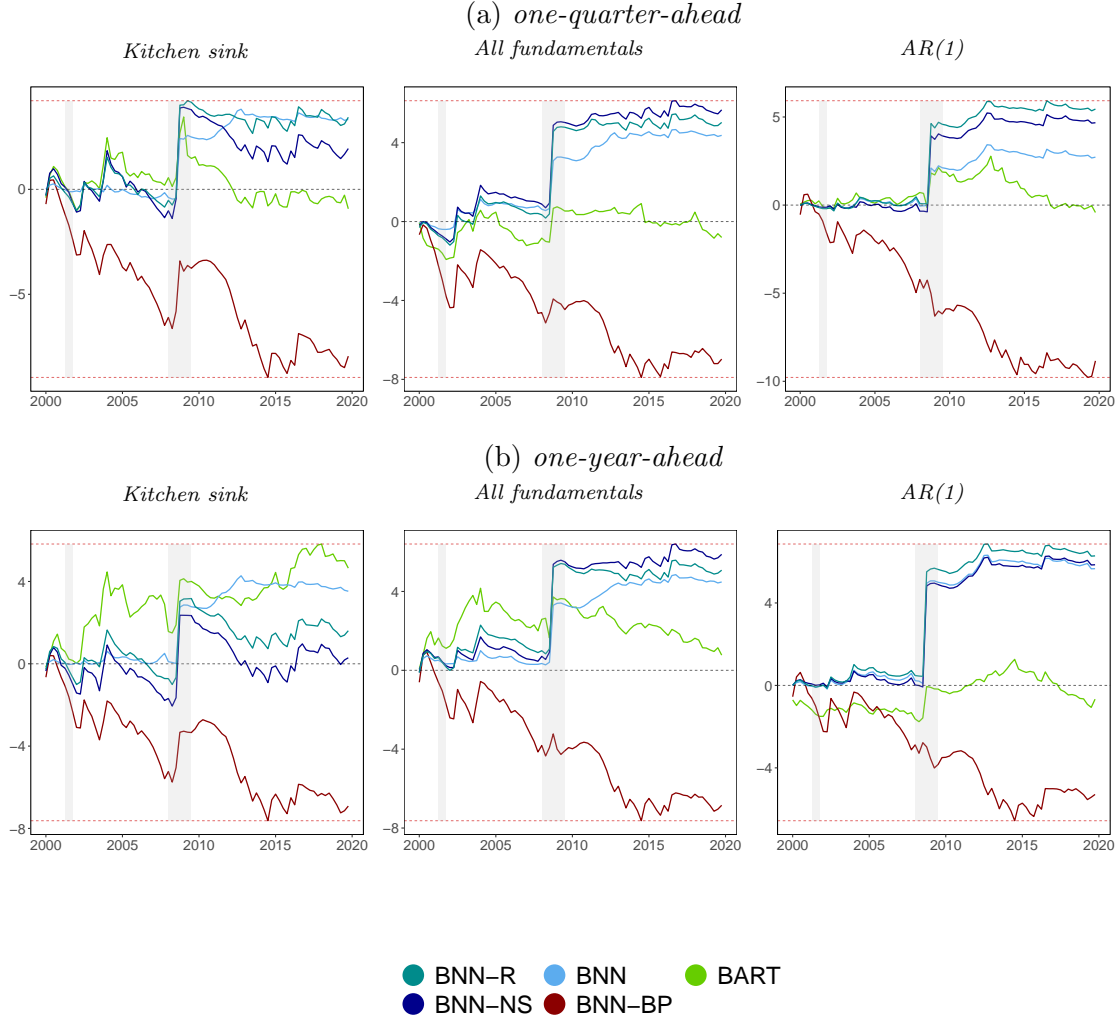
*Note:* The table shows root mean squared errors (RMSEs), and average log predictive likelihoods (LPLs) in parentheses, relative to the linear benchmark. In bold we mark the best performing model for each case. The grey shaded area gives the actual RMSE and LPL scores of our benchmark (linear model). Asterisks indicate statistical significance by means of the Diebold and Mariano (1995) test for each model relative to the benchmark at the 1% (\*\*\*), 5% (\*\*) and 10% (\*) significance levels. The red shaded area marks the best performing model in terms of LPL which we analyze in more detail. Results are averaged across the hold-out.

**Table B.6: Finance.** Forecast performance across 56 hold-out observations (one-year-ahead).

Covariates	Multivariate models					
	BART	BNN	BNN-NS	BNN-R	BNN-BP	Linear model
Kitchen sink	1.01 (0.00)	1.03 (-0.06)	1.02 (-0.01)	<b>0.98</b> ( <b>0.03</b> )	1.03 (-0.04)	
Kitchen sink (constant)		1.03 (-0.05*)	1.07 (-0.03)	1.01 ( <b>0.01</b> )		<b>1.01</b> (-0.03)
Kitchen sink (sv, constant)		1.00 (0.00)	1.02 (0.00)	<b>1.00</b> ( <b>0.00</b> )		1.07 (-1.46)
Kitchen sink (sv)	1.00* (0.01)	1.00 (0.01)	1.02 (-0.02)	<b>0.97</b> ( <b>0.03</b> )		
	Univariate models					
	BART	BNN-NS	BNN-NS (sv)	BNN-BP	Linear model	Linear model (sv)
Dividend price ratio	1.02 (-0.04)	0.98 (0.03)	<b>0.98</b> ( <b>0.04</b> )	0.99 (0.00)		
Dividend price ratio (constant)		0.99 ( <b>0.04</b> )	<b>0.98</b> (0.03)		0.99 (-0.03)	0.99 (-0.01)
Dividend yield	1.01 (-0.02)	1.00 (0.04)	<b>0.99</b> ( <b>0.04</b> )	1.01 (-0.03)		
Dividend yield (constant)		0.99 ( <b>0.04</b> )	<b>0.98</b> (0.03)		0.99 (-0.02)	0.99 (0.00)
Inflation	1.03 (-0.03)	1.00 ( <b>0.05</b> )	<b>0.99</b> (0.04)	1.00 (0.00)		
Inflation (constant)		1.01 ( <b>0.04</b> )	<b>1.00</b> (0.04)		1.01 (-0.02)	1.00 (0.00)
Term spread	1.00 (-0.01)	<b>0.99</b> ( <b>0.06</b> )	0.99 (0.05)	0.99 (0.00)		
Term spread (constant)		1.00 (0.04)	1.00 ( <b>0.04</b> )		1.00 (-0.03)	<b>0.99</b> (-0.01)

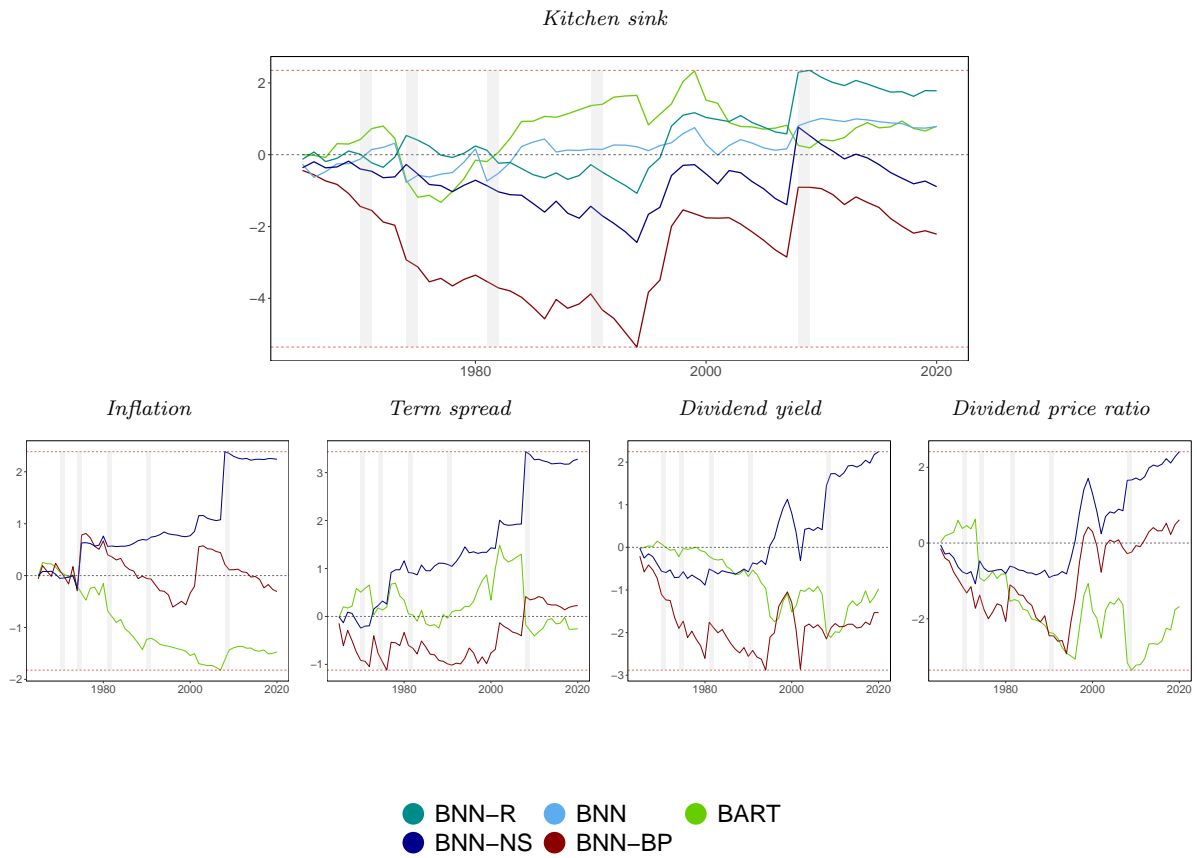
*Note:* The table shows root mean squared errors (RMSEs), and average log predictive likelihoods (LPLs) in parentheses, relative to the linear benchmark. In bold we mark the best performing model for each case. The grey shaded area gives the actual RMSE and LPL scores of our benchmark (linear model). The red shaded area marks the best performing model in terms of LPL which we analyze in more detail. Asterisks indicate statistical significance by means of the Diebold and Mariano (1995) test for each model relative to the benchmark at the 1% (\*\*), 5% (\*) and 10% (\*) significance levels. Results are averaged across the hold-out.

**Figure B.3: Macro C.** Evolution of cumulative LPLs against the benchmark.



*Note:* This figure shows cumulative log predictive likelihoods (LPLs) against the benchmark for each specification. Here, we choose the linear model of each specification as our benchmark to highlight the effect of controlling for nonlinearities. Note that this is in contrast to the tables where we choose a global benchmark for each application. The red dashed lines denote the max./min. LPLs at the end of the hold-out sample, while the gray shaded areas indicate the NBER recessions.

**Figure B.4: Finance.** Evolution of cumulative LPLs against the benchmark.

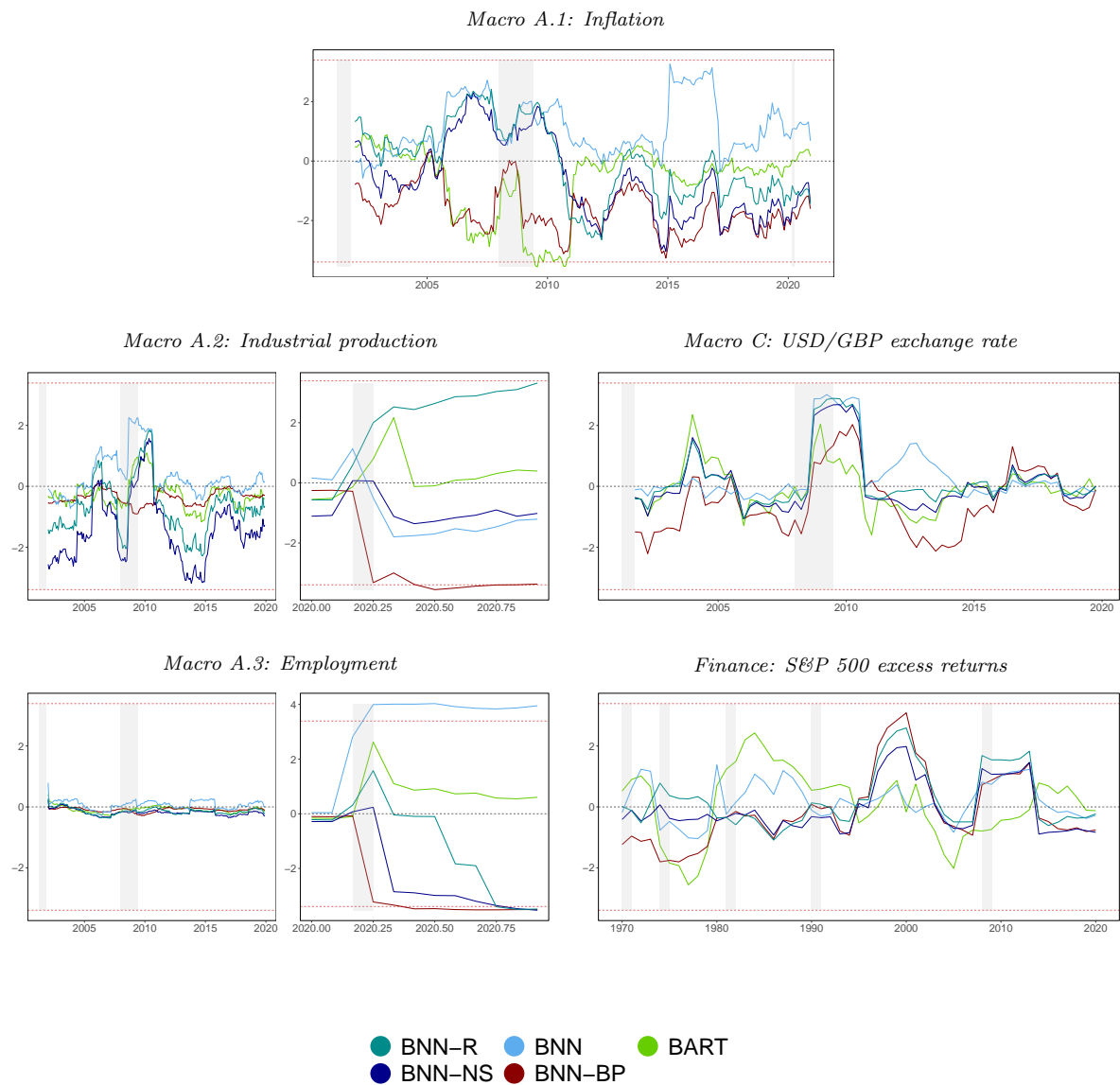


*Note:* This figure shows cumulative log predictive likelihoods (LPLs) against the benchmark for each specification. Here, we choose the linear model of each specification as our benchmark to highlight the effect of controlling for nonlinearities. Note that this is in contrast to the tables where we choose a global benchmark for each application. The red dashed lines denote the max./min. LPLs at the end of the hold-out sample, while the gray shaded areas indicate the NBER recessions.

### B.3 Fluctuation tests

To test whether the outperformance presented in Figure 2 is stable over time, we calculate the fluctuation test statistic as proposed by [Giacomini and Rossi \(2010\)](#). Again, the linear model with the horseshoe prior serves as the benchmark in each application. We choose the 5 percent level for statistical significance. Positive values of the fluctuation test statistic imply that the corresponding model outperforms the benchmark. If model performance is inferior the test statistic gives negative values. Results are presented in Figure B.5.

**Figure B.5:** Fluctuation test statistic for the one-step-ahead density forecast performance.



*Note:* This figure shows the fluctuation test statistic relative to the benchmark as proposed by [Giacomini and Rossi \(2010\)](#). Dashed lines indicate critical values for a 5% level of statistical significance. Positive values of the fluctuation test imply that the corresponding model outperforms the benchmark.



## B.4 MCMC diagnostics of posterior estimates

We analyze the convergence properties of our algorithm by means of inefficiency factors (IF) and the [Raftery and Lewis \(1992\)](#) diagnostics. The former is determined by the inverse of the relative effective sample size. The latter counts the number of draws taken from the algorithm which are necessary to reach a certain level of precision. Table B.7 presents for each dataset the IFs in the upper panel and the [Raftery and Lewis \(1992\)](#) statistic in the lower panel. According to both metrics our algorithm features satisfactory convergence properties.

**Table B.7:** Summary of MCMC diagnostics of posterior estimates.

Application	$\beta$			Summary Statistics $\kappa$			$\nu$		
	Median	10 <sup>th</sup> Perc.	90 <sup>th</sup> Perc.	Median	10 <sup>th</sup> Perc.	90 <sup>th</sup> Perc.	Median	10 <sup>th</sup> Perc.	90 <sup>th</sup> Perc.
Inefficiency factors (IF)									
Macro A.1	2.29	1.00	8.89	2.04	1.00	9.86	1.00	1.00	1.25
Macro A.2	1.00	1.00	1.66	1.84	0.97	8.19	1.00	1.00	1.24
Macro A.3	4.79	1.17	10.34	1.15	0.84	3.40	1.00	1.00	1.23
Macro B	1.60	1.18	5.73	2.74	1.19	20.37	2.68	2.68	2.68
Macro C	1.33	1.00	2.59	2.43	1.17	13.59	1.00	1.00	1.31
Finance	1.31	1.00	3.31	4.01	1.59	23.27	1.00	1.00	1.38
Raftery and Lewis (1992) diagnostics									
Macro A.1	171	145	827	423	240	2004	157	145	171
Macro A.2	157	145	172	423	221	1942	157	145	171
Macro A.3	171	145	221	203	157	570	157	145	171
Macro B	171	157	204	596	263	2989	145	145	145
Macro C	171	145	188	423	221	1343	157	145	186
Finance	157	145	186	644	287	1717	157	145	186

**Notes:** The table shows the inefficiency factors, specified as the inverse of the relative effective sample size, and the [Raftery and Lewis \(1992\)](#) diagnostics of the number of runs to obtain the 2.5<sup>th</sup> percentile with 95% probability and 2.5% accuracy.