

# Enhanced Bayesian Neural Networks for Macroeconomics and Finance

Niko Hauzenberger<sup>1</sup>, Florian Huber<sup>1</sup>,  
Karin Klieber<sup>2</sup>, Massimiliano Marcellino<sup>3</sup>

<sup>1</sup>University of Salzburg, <sup>2</sup>Oesterreichische Nationalbank, <sup>3</sup>Bocconi  
University



\*The content of these slides reflects the views of the authors and not necessarily those of the OeNB or the Eurosystem.



OESTERREICHISCHE NATIONALBANK  
EUROSYSTEM

**Karin Klieber**  
*Monetary Policy Section*  
*Oesterreichische Nationalbank*  
karin.klieber@oenb.at

# Motivation (1)

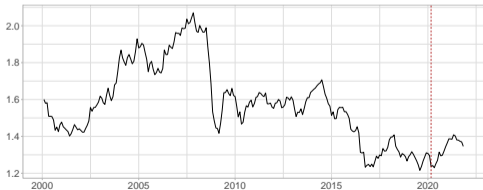
*US inflation*



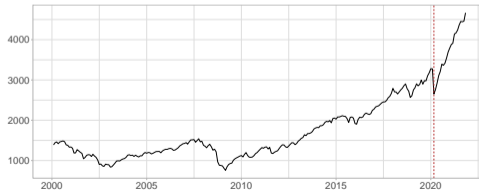
*US industrial production*



*US-UK exchange rate*



*S.P.500 index*



## Motivation (2)

---

- ▶ Practitioners and academics rely on large datasets to form forecasts, tailor policies or improve decisions, often particularly relevant in problematic times
- ▶ Regularization-based techniques as popular way to overcome the curse of dimensionality (Belmonte et al., 2014; Bhattacharya and Dunson, 2011; Carvalho et al., 2010; Griffin and Brown, 2013; Huber and Pfarrhofer, 2021)
- ▶ However, **common assumption of linearity** often remains
- ▶ Two major questions arise:
  - ▶ **How to model the relationship between a response and a large set of covariates?**
  - ▶ **How to safeguard against overfitting?**

# What we do (1)

---

## Methodological

- ▶ Apply neural networks as a device for learning any function under relatively few assumptions (Hornik et al., 1989), yet, performance of NNs in macro forecasting rather bad (Makridakis S., 2018)
- ▶ Use Bayesian methods to determine the adequate network structure
  - ▶ Allowing for either sparse or dense datasets (Giannone et al., 2021)
  - ▶ Combining alternative activation functions (Agostinelli et al., 2014; Karlik and Olgac, 2011)
  - ▶ Selecting the number of neurons via shrinkage (Bhattacharya and Dunson, 2011; Carvalho et al., 2010)
  - ▶ Introducing heteroscedasticity in the error terms (Kastner and Frühwirth-Schnatter, 2014)
- ▶ Design efficient MCMC algorithm for estimation of the resulting BNN even with large set of regressors

## What we do (2)

---

### Empirical

- ▶ Show that our approach works well in simulations
- ▶ Apply BNNs to a set of prominent macro and finance applications
- ▶ Conduct a thorough forecasting exercise
- ▶ Explore the degree of non-linearities

# Econometric framework

## A general non-linear regression

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

- ▶  $y_t$  denotes a scalar time series,
- ▶  $\mathbf{x}_t$  a set of  $K$  covariates,
- ▶  $\boldsymbol{\gamma}$  a vector of  $K$  (linear) coefficients,
- ▶  $f : \mathbb{R}^K \rightarrow \mathbb{R}$  a function of unknown (non-linear) form

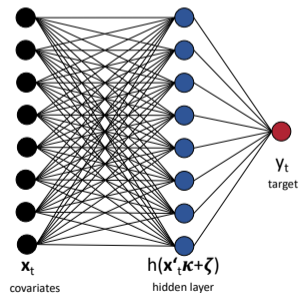
### ▶ How to specify $f$ ?

# A shallow Bayesian Neural Network

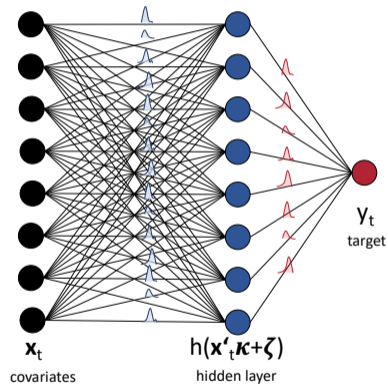
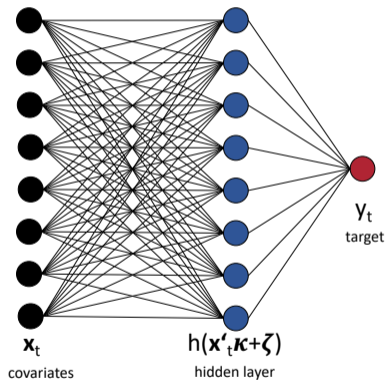
## A neural network with a single hidden layer

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

- ▶  $\boldsymbol{\beta} = (\beta_1, \dots, \beta_Q)'$  denotes a  $Q \times 1$  vector of factor loadings
- ▶  $Q$  the number of neurons
- ▶  $h(\bullet)$  a non-linear activation function
- ▶  $\boldsymbol{\kappa} = (\boldsymbol{\kappa}_1, \dots, \boldsymbol{\kappa}_Q)$  a  $K \times Q$  matrix of non-linear coefficients
- ▶  $\boldsymbol{\zeta} = (\zeta_1, \dots, \zeta_Q)'$  a  $Q \times 1$  vector of bias terms



# From Neural Networks to Bayesian Neural Networks





# Stochastic model selection

- ▶ Selecting the number of neurons ( $Q$ )
  - ▶ Multiplicative Gamma Process (MGP) prior (Bhattacharya and Dunson, 2011)

$$\beta_j \sim \mathcal{N}(0, \phi_j^{-1}), \quad \phi_j = \prod_{q=1}^Q \delta_q, \quad \delta_1 \sim \mathcal{G}(\mathbf{a}_1, 1), \quad \delta_l \sim \mathcal{G}(\mathbf{a}_2, 1), \quad \text{for } q > 1$$

- ▶ Shrinking the weighting coefficients ( $\kappa$ )
  - ▶ Column-wise horseshoe prior (Carvalho et al., 2010)

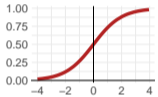
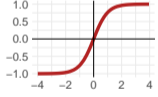
$$\kappa_{qj} \sim \mathcal{N}(0, \lambda_q^2 \phi_{qj}^2), \quad \lambda_q \sim \mathcal{C}^+(0, 1), \quad \phi_{qj} \sim \mathcal{C}^+(0, 1)$$

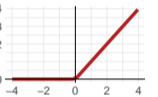
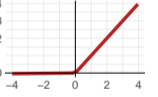
- ▶ Choosing between (four) activation functions ( $h$ )
  - ▶ Introduce latent discrete random variable  $\delta_q \in [1, 4]$

$$\text{Prob}(\delta_q = j) = \omega_{qj} = \frac{1}{4}$$

- ▶ Shrink  $\gamma$  with the horseshoe prior (Carvalho et al., 2010)

# Set of activation functions

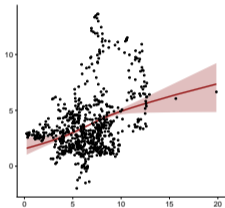
Function	$h(x) =$	Plot
sigmoid	$\frac{1}{1+\exp(-x)}$	
tanh	$\frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$	

Function	$h(x) =$	Plot
relu	$\max(0, x)$	
leaky relu	$\begin{cases} 0.01x & x < 0 \\ x & x \geq 0 \end{cases}$	

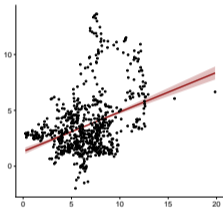
# Illustration of the activation functions

## Effect of money growth on inflation

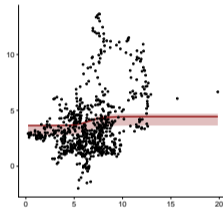
*convex combination*



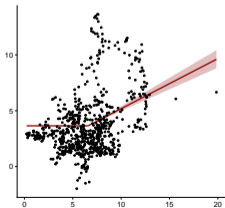
*linear*



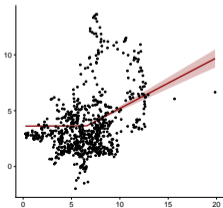
*sigmoid*



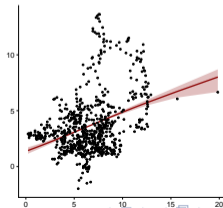
*relu*



*leakyrelu*



*tanh*



# Posterior simulation

- ▶ Coefficients  $\gamma$  and  $\beta$  are obtained jointly from a standard multivariate Gaussian posterior
- ▶ Hyperparameters for MGP prior are obtained through simple Gibbs updating steps (Bhattacharya and Dunson, 2011)
- ▶  $p(\kappa_q|\bullet)$ , for  $q = 1, \dots, Q$ :
  - ▶ If the corresponding scaling parameter of the MGP prior exceeds a threshold very close to zero, we sample  $\kappa_q$  using HMC step (Neal et al., 2011)
  - ▶ Otherwise,  $\kappa_q$  is obtained by drawing from the prior
- ▶  $h_q(x)$  is simulated from a multinomial distribution resulting in a mixture of activation functions
- ▶  $\{\log(\sigma_t^2|\bullet)\}_{t=1}^T$  is simulated using the algorithm proposed in Kastner and Frühwirth-Schnatter (2014)

# Simulation study

---

- ▶ We illustrate our approach through different DGPs and vary
  - ▶ the **functional form**: linear vs highly nonlinear (neural network model)
  - ▶ the **variance specification**: homoskedastic vs heteroskedastic
  - ▶ the **size of the model**: large (60) vs small (30) number of regressors
  - ▶ the **parameter space**: sparse vs dense
  
- ▶ We show that our approach works well for both DGPs
  - ▶ We clearly outperform the linear model for the highly nonlinear DGPs
  - ▶ We can capture the linear DGP without overfitting
  - ▶ Details are given in the appendix

Details

Results

# Forecasting setup (1)

	Dependent variable	Set of predictors	Sample	Range	-	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-ahead	2000M1 to 2020M12	<a href="#">McCracken and Ng (2016)</a>
<b>Macro B</b>	Average economic growth rate	60 country-specific characteristics	Cross-section	90 countries	randomly sampled (100 times)	45 countries	<a href="#">Barro and Lee (1994)</a>
<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	<a href="#">Wright (2008)</a> ; <a href="#">Rossi (2013)</a>
<b>Finance</b>	Equity premium	16 economic & financial variables	Annual data for the US	1948 to 2020	one-year-ahead	1965 to 2020	<a href="#">Welch and Goyal (2008)</a>

## Forecasting setup (2)

---

- ▶ Set of competing models:
  - ▶ Bayesian neural network (BNN)
  - ▶ BNN with neuron-specific activation functions (BNN-NS)
  - ▶ Bayesian linear regression model with the horseshoe shrinkage prior and stochastic volatility (benchmark)
  - ▶ Bayesian neural network by backpropagation, labeled BNN-BP (Blundell et al., 2015) [Details](#)
  - ▶ Bayesian additive regression trees, labeled BART (Chipman et al., 2010; Huber et al., 2020; Huber and Rossini, 2022) [Details](#)
  
- ▶ Evaluation metrics:
  - ▶ Root mean squared error (RMSE) for point forecasts
  - ▶ Log predictive likelihood (LPL) for density forecasts

# Summary of findings

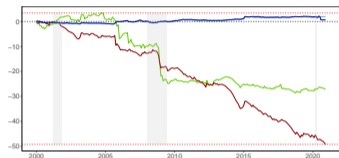
---

- ▶ Superior forecasting performance
  - ▶ Our BNN offers substantial improvements in density forecasting performance
  - ▶ We find competitive performance when interest centers on point forecasts
- ▶ Controlling for non-linear relations is of particular importance during recessionary periods of the business cycle
  - ▶ For Macro A, we see the highest gains during the Global Financial Crisis and the COVID-19 pandemic
  - ▶ For Macro C and Finance, the BNN yields superior density forecasting performance during the Global Financial Crisis
- ▶ In the cross section, the BNN handles extraordinarily low and high growth rates best
- ▶ Good performance in terms of density forecasts is often accompanied by larger in-sample fit for extreme observations
- ▶ Deep BNN yields comparable results [Details](#)

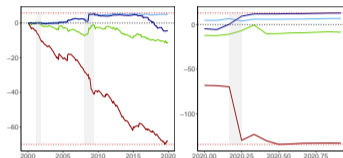


# Out-of-sample predictive accuracy - relative LPL

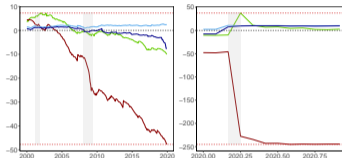
*Inflation*



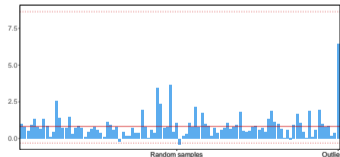
*Industrial production*



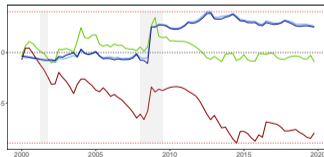
*Employment*



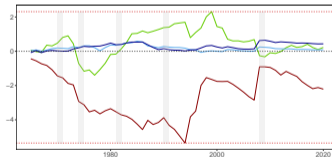
*GDP per capita growth*



*USD/GBP exchange rate*

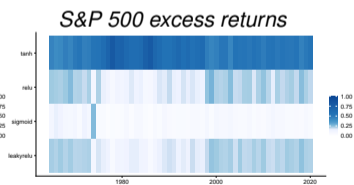
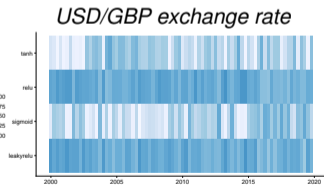
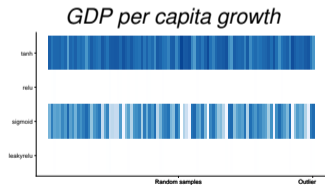
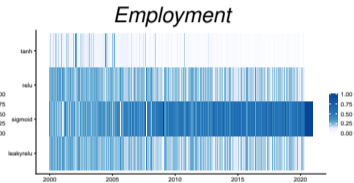
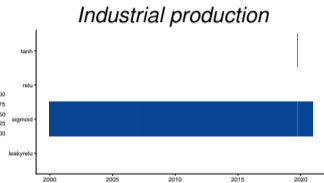
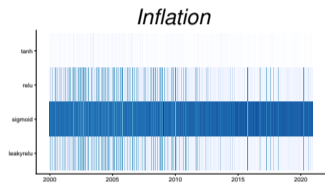


*S&P 500 excess returns*

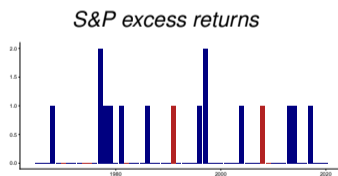
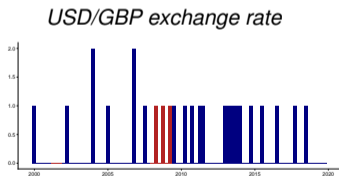
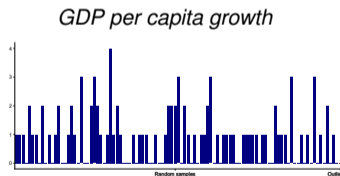
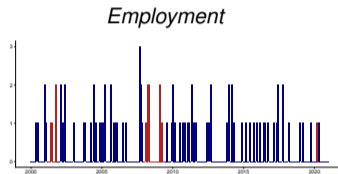
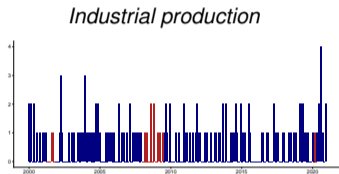
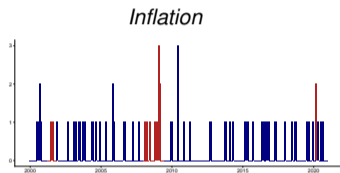


● BNN   ● BNN-NS   ● BNN-BP   ● BART

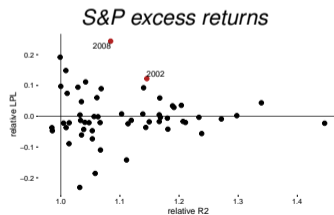
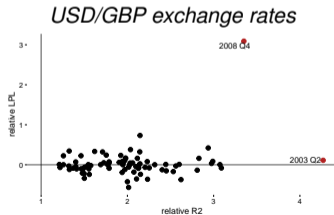
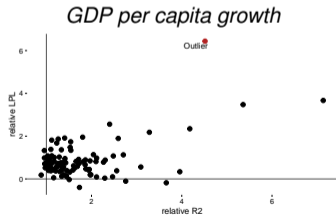
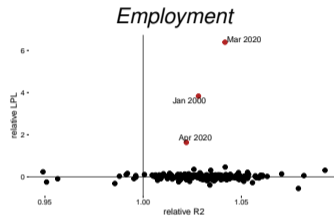
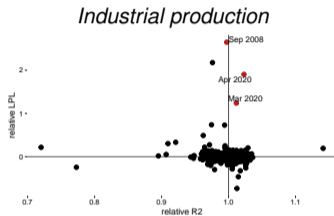
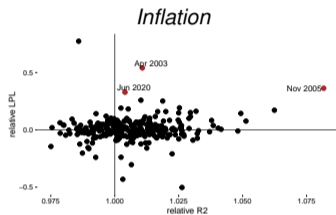
# Non-linearities in the sample



# Effective number of neurons $Q^*$



# Relationship in-sample fit and out-of-sample predictability



## Closing remarks

---

- ▶ We have developed a non-parametric regression model based on Bayesian Neural Networks
- ▶ Our framework allows to remain agnostic on the form of the network
- ▶ We use popular techniques from the Bayesian literature to determine the adequate network structure
- ▶ In a broad set of macro and finance applications we show the superior forecasting performance of our approach

# References I

---

- AGOSTINELLI, F., M. HOFFMAN, P. SADOWSKI, AND P. BALDI (2014): “Learning activation functions to improve deep neural networks,” *arXiv preprint arXiv:1412.6830*.
- BARRO, R. J., AND J.-W. LEE (1994): “Sources of economic growth,” in *Carnegie-Rochester conference series on public policy*, vol. 40, pp. 1–46. Elsevier.
- 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*, pp. 291–306.
- 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.
- 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.

## References II

---

- GIANNONE, D., M. LENZA, AND G. E. PRIMICERI (2021): “Economic predictions with big data: The illusion of sparsity,” *Econometrica*, 89(5), 2409–2437.
- GRIFFIN, J. E., AND P. J. BROWN (2013): “Some priors for sparse regression modelling,” *Bayesian Analysis*, 8(3), 691–702.
- HOFFMAN, M. D., A. GELMAN, ET AL. (2014): “The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo,” *Journal of Machine Learning Research*, 15(1), 1593–1623.
- HORNIK, K., M. STINCHCOMBE, AND H. WHITE (1989): “Multilayer feedforward networks are universal approximators,” *Neural Networks*, 2(5), 359–366.
- 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 L. ROSSINI (2022): “Inference in Bayesian additive vector autoregressive tree models,” *The Annals of Applied Statistics*, 16(1), 104–123.

## References III

---

- 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.
- 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.
- MCCRACKEN, M. W., AND S. NG (2016): “FRED-MD: A monthly database for macroeconomic research,” *Journal of Business & Economic Statistics*, 34(4), 574–589.
- NEAL, R. M., ET AL. (2011): “MCMC using Hamiltonian dynamics,” *Handbook of Markov Chain Monte Carlo*, 2(11), 2.
- ROSSI, B. (2013): “Exchange rate predictability,” *Journal of Economic Literature*, 51(4), 1063–1119.



## References IV

---

- 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.
- WRIGHT, J. H. (2008): “Bayesian model averaging and exchange rate forecasts,” *Journal of Econometrics*, 146(2), 329–341.

# APPENDIX

## Bayesian neural learning of non-linearities in macroeconomics and finance

# Deep BNN [Go back](#)

A deep BNN with  $L$  hidden layers implies that:

$$f(\mathbf{x}_t) \approx \left( h^{(L)}(\boldsymbol{\kappa}^{(L)'} \hat{\mathbf{x}}_t^{(L-1)} + \zeta^{(L)}) \right) \beta, \quad \text{with}$$
$$\hat{\mathbf{x}}_t^{(l)} = h^{(l)}(\boldsymbol{\kappa}^{(l)'} \hat{\mathbf{x}}_t^{(l-1)} + \zeta^{(l)}), \quad \text{for } 1 \leq l \leq L - 1.$$

- ▶  $l = \{1, \dots, L\}$  denotes the number of layers
- ▶  $h^{(l)}(\bullet)$  refers to a layer-specific (nonlinear) activation function common to all neurons  $\boldsymbol{\kappa}^{(l)}$  and bias terms  $\zeta^{(l)}$  in the respective layer
- ▶  $\hat{\mathbf{x}}_t^{(0)} = \hat{\mathbf{x}}_t$  for the first layer (i.e.,  $l = 1$ )

# Results for the Deep BNN

Application		Deep BNN with 3 hidden layers
Macro A	Inflation	0.00
	Industrial production	-0.02
	Employment	0.03
Macro B		-0.38
Macro C		0.02
Finance		-0.02

*Note:* The table shows log predictive likelihoods (LPLs) relative to the best performing shallow BNN. Results are averaged across the hold-out.

# Simulation study [Go back](#)

$$y_t = f(\mathbf{x}_t' \boldsymbol{\kappa}_{true})' \boldsymbol{\beta}_{true} + v_t, \quad v_t \sim \mathcal{N}(0, \sigma_{t,true}^2), \quad \boldsymbol{\kappa}_{true} = \mathbf{I}_K,$$
$$x_{jt} \sim \mathcal{N}(0, 1), \text{ for } j = 1, \dots, K \text{ and } K \in [30, 60],$$

- ▶  $\beta_{j,true} \sim N(0, c^2)$ 
  - ▶ sparse DGP with 10 % active neurons:  $c = 0.5^2$
  - ▶ dense DGP with 90 % active neurons:  $c = 0.04$
- ▶ 2 different variance specifications:
  - ▶ homoskedastic:  $\sigma_{t,true}^2 = 0.1$  for all  $t$
  - ▶ heteroskedastic:  $\sigma_{t,true}^2 = 0.1 \exp(\eta_t)$ ,  $\eta_t \sim \mathcal{N}(0, 0.1^2)$
- ▶  $f : R^K \mapsto R^Q$  defined as
  - ▶ linear
  - ▶ a shallow neural network with a single hidden layer and a randomly selected activation function for each neuron

# Synthetic: Forecast performance for 100 hold-out observations (estimated with SV) [Go back](#)

K	Sparsity	Noise	Non-linear DGP			Linear DGP		
			BNN	BNN-NS	Linear model	BNN	BNN-NS	Linear model
30	Dense	hetero	1.00	<b>0.93</b>	0.51	1.01	<b>1.01</b>	0.43
		homo	0.99	<b>0.86</b>	0.41	1.02	<b>1.02</b>	0.32
	Sparse	hetero	0.99	<b>0.80</b>	0.98	0.99	<b>0.99</b>	0.49
		homo	0.98	<b>0.72</b>	1.00	1.01	<b>1.01</b>	0.35
60	Dense	hetero	1.00	<b>0.89</b>	0.61	1.00	<b>1.00</b>	0.42
		homo	1.01	<b>0.84</b>	0.54	1.01	<b>1.01</b>	0.33
	Sparse	hetero	0.99	<b>0.92</b>	1.48	<b>0.98</b>	0.99	0.51
		homo	1.00	<b>0.94</b>	1.37	1.01	<b>1.00</b>	0.38

*Note:* The table shows root mean squared errors (RMSEs) relative to the benchmark linear model. 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.

# Synthetic: Forecast performance for 100 hold-out observations (estimated without SV) [Go back](#)

K	Sparsity	Noise	Non-linear DGP			Linear DGP		
			BNN	BNN-NS	Linear model	BNN	BNN-NS	Linear model
30	Dense	hetero	1.01	<b>0.94</b>	0.51	1.02	<b>1.02</b>	0.43
		homo	0.98	<b>0.84</b>	0.41	1.02	<b>1.02</b>	0.32
	Sparse	hetero	1.01	<b>0.84</b>	0.98	1.02	<b>1.01</b>	0.49
		homo	1.02	<b>0.72</b>	1.01	1.00	<b>1.00</b>	0.35
60	Dense	hetero	1.01	<b>0.87</b>	0.61	1.02	<b>1.02</b>	0.43
		homo	1.01	<b>0.85</b>	0.53	1.01	<b>1.01</b>	0.33
	Sparse	hetero	1.00	<b>0.93</b>	1.48	1.01	<b>1.01</b>	0.51
		homo	1.01	<b>0.95</b>	1.38	<b>1.01</b>	1.01	0.38

*Note:* The table shows root mean squared errors (RMSEs) relative to the benchmark linear model. 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.

# Macro A: Forecast performance across 252 hold-out observations

[Go back](#)

Covariates	Model				
	BART	BNN	BNN-NS	BNN-BP	Linear model
	Inflation				
Large	0.97 (-0.03)	<b>0.94***</b> ( <b>0.09***</b> )	0.94*** (0.08***)	1.03 (-0.12***)	0.94*** (0.08***)
PCA	1.07** (-0.20***)	<b>1.00</b> ( <b>0.00</b> )	1.01 (-0.04***)	1.04* (-0.11***)	1.16 (-1.45)
	Industrial production				
Large	<b>0.88</b> (0.08)	0.97*** (0.14***)	1.02** ( <b>0.17**</b> )	0.93 (-0.41*)	0.98*** (0.12***)
PCA	<b>0.89</b> (-0.01)	1.00 (0.05**)	1.01 ( <b>0.07*</b> )	0.94 (-0.46)	1.75 (-1.33)
	Employment				
Large	1.04 (0.11)	<b>1.00</b> ( <b>0.14**</b> )	1.00 (0.14)	1.01 (-0.87)	1.01 (0.10*)
PCA	1.03 (-0.19*)	<b>0.99</b> ( <b>0.07</b> )	1.00 (-0.01)	1.01 (-0.59)	3.50 (-1.88)

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). Results are averaged across the hold-out.



# Macro B: Forecast performance across 45 hold-out countries and 100 replications

[Go back](#)

Covariates	Model				
	BART	BNN	BNN-NS	BNN-BP	Linear model
Kitchen sink	<b>0.95***</b> (0.06***)	1.01 (0.82***)	1.01 ( <b>0.95***</b> )	1.04*** (-0.18***)	5.24 (-4.40)

*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). Results are averaged across the hold-out.

# Macro C: Forecast performance across 80 hold-out observations

[Go back](#)

Covariates	Multivariate models				
	BART	BNN	BNN-NS	BNN-BP	Linear model
one-quarter ahead					
All fundamentals	1.03 (-0.02)	1.02 (0.01)	<b>1.02</b> <b>(0.02)</b>	1.03 (-0.09*)	1.01 (-0.01)
Kitchen sink	1.04 (-0.01)	<b>0.97</b> <b>(0.03)</b>	0.99 (0.03)	1.02 (-0.10)	0.95 (-1.29)
one-year-ahead					
All fundamentals	1.04 (0.00)	1.00 <b>(0.02)</b>	<b>1.00</b> (0.00)	1.01 (-0.09**)	1.00 (-0.01)
Kitchen sink	1.03 <b>(0.06)</b>	1.01 (0.01)	1.02 (0.01)	<b>0.98**</b> (-0.09)	0.96 (-1.29)

*Note:* The table shows RMSEs with average LPLs in parentheses. The grey shaded area gives the actual RMSE and LPL scores of our benchmark (linear model). Results are averaged across the hold-outs.

# Finance: Forecast performance across 56 hold-out observations

[Go back](#)

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

Note: The table shows RMSEs with average LPLs in parentheses. The grey shaded area gives the actual RMSE and LPL scores of our benchmark (linear model). Results are averaged across the hold-outs

# Full conditional posterior distribution (1)

Go back

- ▶ Sample  $\theta = (\gamma', \beta')'$

$$\theta | \bullet \sim \mathcal{N}(\bar{\theta}, \underline{\mathbf{V}}_{\theta}),$$

with

$$\underline{\mathbf{V}}_{\theta} = \left( \tilde{\mathbf{x}}' \Sigma^{-1} \tilde{\mathbf{x}} + \underline{\mathbf{V}}_{\theta}^{-1} \right)^{-1},$$
$$\bar{\theta} = \underline{\mathbf{V}}_{\theta} \tilde{\mathbf{x}}' \Sigma^{-1} \mathbf{y}.$$

- ▶  $\tilde{\mathbf{x}} = (\tilde{\mathbf{x}}'_1, \dots, \tilde{\mathbf{x}}'_T)'$  as a  $(K + Q) \times T$  matrix of neurons with element  $\tilde{\mathbf{x}}'_t = (\mathbf{x}'_t, h_1(\mathbf{x}'_t \kappa_1 + \zeta_1), \dots, h_Q(\mathbf{x}'_t \kappa_Q + \zeta_Q))'$
- ▶  $\Sigma = \text{diag}(\sigma_1^2, \dots, \sigma_T^2)$  as a  $T \times T$  matrix capturing the variances
- ▶  $\underline{\mathbf{V}}_{\theta} = \text{diag}(\phi_{\gamma}^{-1}, \phi_{\beta}^{-1})$  where  $\phi_{\gamma}^{-1} = (\phi_{\gamma_1}^{-1}, \dots, \phi_{\gamma_K}^{-1})'$  as the  $K$  prior variances for the constant coefficients and  $\phi_{\beta}^{-1} = (\phi_{\beta_1}^{-1}, \dots, \phi_{\beta_Q}^{-1})'$  for the non-linear coefficients

## Full conditional posterior distribution (2)

Go back

- ▶ The prior on  $\gamma$  is Normal of the form:

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

with the global and local shrinkage parameters,  $\lambda_\gamma^2$  and  $\varphi_{\gamma_j}^2$  (HS prior in hierarchical representation of [Makalic and Schmidt \(2015\)](#)):

$$\varphi_{\gamma_j}^2 | \bullet \sim \mathcal{G}^{-1} \left( 1, \mathbf{c}_{\gamma_j}^{-1} + \frac{\gamma_j^2}{2\lambda_\gamma^2} \right),$$

$$\lambda_\gamma^2 | \bullet \sim \mathcal{G}^{-1} \left( \frac{K+1}{2}, \mathbf{d}_\gamma^{-1} + \sum_{j=1}^K \frac{\gamma_j^2}{2\varphi_{\gamma_j}^2} \right),$$

$$\mathbf{c}_{\gamma_j} | \bullet \sim \mathcal{G}^{-1} \left( 1, 1 + \varphi_{\gamma_j}^{-2} \right),$$

$$\mathbf{d}_\gamma | \bullet \sim \mathcal{G}^{-1} \left( 1, 1 + \lambda_\gamma^{-2} \right).$$

## Full conditional posterior distribution (3)

Go back

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

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

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

# Full conditional posterior distribution (4)

Go back

- ▶ Hamiltonian Monte Carlo (HMC) within Gibbs step to draw  $\kappa_q$  ( $q = 1, \dots, Q$ ): Let  $\mathbf{r}_q$  denote an auxiliary moment variable, where  $\mathbf{r}_q \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  and  $\mathcal{L}(\kappa_q) = \log p(\kappa_q | \bullet)$  the log conditional posterior density of  $\kappa_q$ . The fictitious Hamiltonian system of the conditional posterior density of  $\kappa_q$  is then given by

$$\mathcal{H}(\kappa_q, \mathbf{r}_q) = -\mathcal{L}(\kappa_q) + \frac{1}{2} \mathbf{r}_q' \mathbf{r}_q.$$

We simulate the Hamiltonian dynamics via the leapfrog integrator with proposals:

$$\mathbf{r}_q^{**} = \mathbf{r}_q^{(a)} + \frac{\epsilon}{2} \nabla_{\kappa_q} \mathcal{L}(\kappa_q^{(a)}),$$

$$\kappa_q^* = \kappa_q^{(a)} + \epsilon \mathbf{r}_q^{**},$$

$$\mathbf{r}_q^* = \mathbf{r}_q^{**} + \frac{\epsilon}{2} \nabla_{\kappa_q} \mathcal{L}(\kappa_q^*),$$

where  $\kappa_q^{(a)}$  and  $\mathbf{r}_q^{(a)}$  denote the previously accepted values,  $\nabla_{\kappa_q} \mathcal{L}(\kappa_q)$  is the gradient of the log conditional posterior,  $\epsilon$  is the discrete step size to generate a full-step proposal for  $\kappa_q$  (i.e.,  $\kappa_q^*$ ) and half-step updates for the momentum  $\mathbf{r}_q$  ( $\mathbf{r}_q^{**}$  for the first half-step update and  $\mathbf{r}_q^*$  for the final proposal).

# Full conditional posterior distribution (5)

Go back

- ▶ We repeat the leapfrog method in  $n = 1, \dots, N$  steps. To tune  $N$  and  $\epsilon$  we use the No U-Turn Sampler (NUTS) as in [Hoffman et al. \(2014\)](#).

Finally, we evaluate the proposed and previously accepted values by means of a Metropolis accept/reject step and determine the acceptance probability  $\eta_q$  for proposed  $\kappa_q^*$ :

$$\eta_q = \min \left( 1, \frac{\exp(\mathcal{L}(\kappa_q^*) - \frac{1}{2} \mathbf{r}_q'^* \mathbf{r}_q^*)}{\exp(\mathcal{L}(\kappa_q^{(a)}) - \frac{1}{2} \mathbf{r}_q'^{(a)} \mathbf{r}_q^{(a)})} \right).$$

This step remains conceptually similar for the deep BNN. In this case, we only need to iterate through all layers and also take into account the fact that the composite function of  $L$  layers has different implications on the gradients of the layer-specific neurons.



## Full conditional posterior distribution (6)

Go back

- ▶ Column-wise horseshoe prior on the elements of  $\kappa_q$ :

$$\varphi_{\kappa_{jq}}^2 | \bullet \sim \mathcal{G}^{-1} \left( 1, c_{\kappa_{jq}}^{-1} + \frac{\kappa_{jq}^2}{2\lambda_{\kappa_q}^2} \right),$$

$$\lambda_{\kappa_q}^2 | \bullet \sim \mathcal{G}^{-1} \left( \frac{K+1}{2}, d_{\kappa_q}^{-1} + \sum_{j=1}^K \frac{\kappa_{jq}^2}{2\varphi_{\kappa_{jq}}^2} \right),$$

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

$$d_{\kappa_q} | \bullet \sim \mathcal{G}^{-1} \left( 1, 1 + \lambda_{\kappa_q}^{-2} \right).$$

## Full conditional posterior distribution (7) [Go back](#)

- ▶ Choose activation function  $h_q$  by drawing  $\delta_q$  from a multinomial distribution:

$$\Pr(\delta_q = m | \bullet) \propto \omega_{qm} \times \exp \left\{ -\frac{1}{2} \left( (\hat{\mathbf{y}}_q - \boldsymbol{\mu}_q^{(m)})' \boldsymbol{\Sigma}^{-1} (\hat{\mathbf{y}}_q - \boldsymbol{\mu}_q^{(m)}) \right) \right\}, \quad \text{for } m = 1, \dots, 4,$$

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

# Bayesian Neural Network by backpropagation Go back

- ▶ Variational inference scheme for learning the posterior distribution on the weights of a neural network (Blundell et al., 2015)
- ▶ Maximize the log-likelihood subject to a Kullback-Leibler complexity term on the parameters (with reparameterization trick and stochastic gradient descent)
- ▶ Prior on the weights as a scale mixture of two Gaussian densities with zero mean but differing variances ( $\sigma_1^2 = 3$  and  $\sigma_2^2 = 0.0025$ )
- ▶ Cross validation exercise:
  - ▶ Macro A: time series split with 24 months
  - ▶ Macro B: random split in 20 replications
  - ▶ Macro C: time series split with 12 quarters
  - ▶ Finance: time series split with 10 years
  - ▶ 1000 epochs
  - ▶ MSE loss function
  - ▶ ADAM optimizer
  - ▶ learning rate 0.014

# Bayesian additive regression trees Go back

- ▶ Approximate  $f$  using Bayesian additive regression trees (Chipman et al., 2010)
- ▶ Sum over a number of  $Z$  regression trees ( $g_z$ ):

$$f(\mathbf{x}_t) \approx \sum_{z=1}^Z g_z(\mathbf{x}_t | \mathcal{T}_z, \rho_z)$$

- ▶ Tree structure  $\mathcal{T}_z$
- ▶ Terminal node parameter  $\rho_z$
- ▶  $Z = 250$
- ▶ Prior on the tree structure built upon a tree-generating stochastic process
  - ▶ Determining the probability that a given node is nonterminal
  - ▶ Selection of variables used in a splitting rule (to spawn left and right children nodes)
  - ▶ Terminal node parameter: conjugate Gaussian prior distribution with data-based prior variance

# BART illustration

[Go back](#)

