Multiple Yield Curves Modelling via Deep Learning

Ronald Richman[1]   Salvatore Scognamiglio[2]

[1]Old Mutual Insure and University of the Witwatersrand, Johannesburg, South Africa,
[2]Department of Management and Quantitative Studies, University of Naples, Parthenope.

Insurance Data Science Conference 2024 - Stockholm University.

Financially suppported by:

IAA AAI IAALS Life & Annuities

- Yield curves are used in actuarial science and finance for deriving the present value of future cashflows;
- Several approaches have been proposed to model the uncertain future evolution of the yield curves;
- Globalisation has intensified the financial markets' connection, inducing a complex dependence structure among different yield curves;
- Deep learning has been successfully applied to several tasks of the actuarial domain.

AIM: To develop deep learning models for accurate modelling and forecasting multiple yield curves.

Yield Curves Modelling: A Static approach

Let $y(\tau)$ be the continuously-compounded zero-coupon nominal yield of a τ-month bond, Nelson and Siegel (1987) assume that:

$$y(\tau) = \beta_0 + \beta_1 \left(\frac{1 - e^{-\lambda\tau}}{\lambda\tau} \right) + \beta_2 \left(\frac{1 - e^{-\lambda\tau}}{\lambda\tau} - e^{-\lambda\tau} \right) + \epsilon_\tau$$

where $\beta_0, \beta_1, \beta_2, \lambda \in \mathbb{R}$ are model parameters.

Given a market data sample $\left(\dot{y}(\tau)\right)_{\tau \in \mathcal{M}}$, the parameters are estimated by fixing the decay factor τ, and by solving:

$$\arg\min_{\beta_0,\beta_1,\beta_2} \sum_{\tau \in \mathcal{M}} \left(\dot{y}(\tau) - \beta_0 - \beta_1 \left(\frac{1 - e^{-\lambda\tau}}{\lambda\tau} \right) - \beta_2 \left(\frac{1 - e^{-\lambda\tau}}{\lambda\tau} - e^{-\lambda\tau} \right) \right)^2.$$

Let $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$ be a set of dates. Diebold and Li (2006) introduces the dynamic version of the NS model:

$$y_t(\tau) = \beta_{0,t} + \beta_{1,t}\left(\frac{1 - e^{-\lambda\tau}}{\lambda\tau}\right) + \beta_{2,t}\left(\frac{1 - e^{-\lambda\tau}}{\lambda\tau} - e^{-\lambda\tau}\right) + \epsilon_\tau,$$

where the parameters $\beta_{0,t}, \beta_{1,t}, \beta_{2,t}$ change over time.

They are estimated at each date t, by solving the sequence of optimisation problems:

$$\underset{\beta_{0,t},\beta_{1,t},\beta_{2,t}}{\arg\min} \sum_{\tau\in\mathcal{M}} \left(\dot{y}(\tau) - \beta_{0,t} - \beta_{1,t}\left(\frac{1 - e^{-\lambda_t\tau}}{\lambda_t\tau}\right) - \beta_2\left(\frac{1 - e^{-\lambda_t\tau}}{\lambda_t\tau} - e^{-\lambda\tau}\right)\right)^2 \forall t \in \mathcal{T}.$$

Let $\mathcal{I} = \{\texttt{curve}_1, \texttt{curve}_2, \ldots, \texttt{curve}_l\}$ be a set of different yield curves. The Diebold and Li model in the multi-curve case reads:

$$y_t^{(i)}(\tau) = \beta_{0,t}^{(i)} + \beta_{1,t}^{(i)}\left(\frac{1-e^{-\lambda\tau}}{\lambda\tau}\right) + \beta_{2,t}^{(i)}\left(\frac{1-e^{-\lambda\tau}}{\lambda\tau} - e^{-\lambda\tau}\right) + \epsilon_{\tau,t}^{(i)},$$

where $\beta_{0,t}^{(i)}, \beta_{1,t}^{(i)}, \beta_{2,t}^{(i)}$ are curve-specific parameters.

They are estimated by optimising:

$$\underset{\beta_{0,t}^{(i)},\beta_{1,t}^{(i)},\beta_{2,t}^{(i)}}{\arg\min} \sum_{\tau\in\mathcal{M}} \left(\dot{y}(\tau) - \beta_{0,t}^{(i)} - \beta_{1,t}^{(i)}\left(\frac{1-e^{-\lambda\tau}}{\lambda\tau}\right) - \beta_{2,t}^{(i)}\left(\frac{1-e^{-\lambda\tau}}{\lambda\tau} - e^{-\lambda\tau}\right)\right)^2,$$

that have to be solved for each $t \in \mathcal{T}, i \in \mathcal{I}$.

Forecasts are obtained by specifying a dynamic model for the time-series $\left(\hat{\beta}^{(i)}_{j,t}\right)_{t\in\mathcal{T}}, j = 0, 1, 2, i \in \mathcal{I}$.

The two most popular choices are:

- Independent AR(1) models:

$$\beta^{(i)}_{j,t} = \psi_{0,j} + \psi^{(i)}_{1,j}\beta^{(i)}_{j,t-1} + \epsilon^{(i)}_{j,t},$$

  where $\psi^{(i)}_{0,j}, \psi^{(i)}_{1,j} \in \mathbb{R}, i \in \mathcal{I}, j = 0, 1, 2$ and $\epsilon^{(i)}_{j,t} \sim N(0, (\sigma^{(i)}_j)^2)$.

- A Multivariate VAR(1) models for $\boldsymbol{\beta}^{(i)}_t = (\beta^{(i)}_{0,t}, \beta^{(i)}_{1,t}, \beta^{(i)}_{2,t}) \in \mathbb{R}^3$:

$$\boldsymbol{\beta}^{(i)}_t = \boldsymbol{a}^{(i)}_0 + A^{(i)}\boldsymbol{\beta}^{(i)}_{t-1} + \boldsymbol{\eta}^{(i)}_t,$$

  with $\boldsymbol{a}^{(i)}_0 \in \mathbb{R}^3, A^{(i)} \in \mathbb{R}^{3\times 3}$, and $\boldsymbol{\eta}^{(i)}_t \sim N(0, E^{(i)})$ is the normal distributed error term with covariance matrix $E^{(i)} \in \mathbb{R}^{3\times 3}$.

Yield Curves Modelling: The Nelson-Siegel-Svensson Model

Svensoon (1994) also introduced a four-factor extension of the NS model that, in a dynamic framework, can be formalised as:

$$
\begin{aligned}
y_t^{(i)}(\tau) \quad = \quad & \beta_{0,t}^{(i)} + \beta_{1,t}^{(i)}\left(\frac{1-e^{-\lambda_1\tau}}{\lambda_1\tau}\right) + \beta_{2,t}^{(i)}\left(\frac{1-e^{-\lambda_1\tau}}{\lambda_1\tau} - e^{-\lambda_1\tau}\right) \\
& + \beta_{3,t}^{(i)}\left(\frac{1-e^{-\lambda_2\tau}}{\lambda_2\tau} - e^{-\lambda_2\tau}\right) + \epsilon_{\tau,t}^{(i)},
\end{aligned}
$$

where $\beta_{3,t}^{(i)}, \lambda_1, \lambda_2 \in \mathbb{R}$.

the parameters $\beta_{0,t}^{(i)}, \beta_{1,t}^{(i)}, \beta_{2,t}^{(i)}, \beta_{3,t}^{(i)}$ are estimated via OLS estimator for fixed values for λ_1, λ_2.

Neural Networks

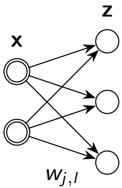
Let $\boldsymbol{x} \in \mathbb{R}^{q_0}$ be the vector of features, a fully connected (FC) layer of size $q_1 \in \mathbb{N}$ is a function

$$\boldsymbol{z} : \mathbb{R}^{q_0} \to \mathbb{R}^{q_1}, \qquad \boldsymbol{x} \mapsto \boldsymbol{z}(\boldsymbol{x}) = (z_1(\boldsymbol{x}), z_2(\boldsymbol{x}), \dots, z_{q_1}(\boldsymbol{x}))^\top .$$

Each component $z_j(\boldsymbol{x})$ is a non-linear function of $\boldsymbol{x}$

$$\boldsymbol{x} \;\mapsto\; z_j(\boldsymbol{x}) = \phi\left(w_{j,0} + \sum_{l=1}^{q_0} w_{j,l} x_l\right) = \phi\left(w_{j,0} + \langle \boldsymbol{w}_j, \boldsymbol{x} \rangle\right), \qquad j = 1, \dots, q_1,$$

where $\phi : \mathbb{R} \to \mathbb{R}$ is the activation function, $w_{j,l} \in \mathbb{R}$ represent the network parameters and $\langle \cdot, \cdot \rangle$ denotes the scalar product in $\mathbb{R}^{q_0}$.



Deep Neural Networks

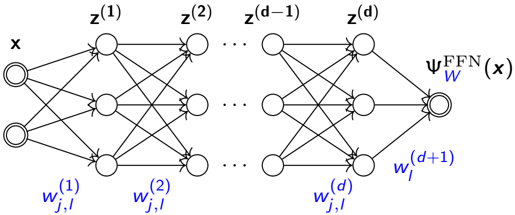
In the case of d layers of size $\boldsymbol{q} = \{q_k\}_{1\leq k\leq d} \in \mathbb{N}^d$, the mapping reads:

$$\boldsymbol{x} \;\mapsto\; \boldsymbol{z}^{(d:1)}(\boldsymbol{x}) \;\stackrel{\text{def}}{=}\; \left(\boldsymbol{z}^{(d)} \circ \cdots \circ \boldsymbol{z}^{(1)}\right)(\boldsymbol{x}) \;\in\; \mathbb{R}^{q_d},$$

where $\boldsymbol{z}^{(k)} : \mathbb{R}^{q_{k-1}} \to \mathbb{R}^{q_k}$. In the case of univariate response, the output of the network is:

$$\boldsymbol{x} \;\mapsto\; \mu_W(\boldsymbol{x}) \;\stackrel{\text{def}}{=}\; \Psi_W^{\mathrm{FFN}}(\boldsymbol{x}) \;\stackrel{\text{def}}{=}\; g^{-1}\left(w_0^{(d+1)} + \sum_{l=1}^{q_d} w_l^{(d+1)} z_l^{(d:1)}(\boldsymbol{x})\right),$$

$g^{-1}(\cdot)$ is an inverse link function.



Attention Layer

Let $Q \in \mathbb{R}^{q\times d}$ be a matrix of query vectors, $K \in \mathbb{R}^{q\times d}$ be a matrix of key vectors, $V \in \mathbb{R}^{q\times d}$ is a matrix of value vectors.
The scaled dot-product attention mechanism is a mapping:

$$A : \mathbb{R}^{(q\times d)\times(q\times d)\times(q\times d)} \to \mathbb{R}^{(q\times d)}, \qquad (Q, K, V) \mapsto A = \mathtt{attn}(\mathtt{Q}, \mathtt{K}, \mathtt{V}).$$

Formally, the attention mapping has the following structure:

$$A = \mathtt{softmax}(\mathtt{B})\mathtt{V} = \mathtt{softmax}\left(\frac{\mathtt{Q}\mathtt{K}^\top}{\sqrt{\mathtt{d}}}\right)\mathtt{V}$$

where $d \in [0, +\infty)$ is a scalar coefficient, and the matrix of the scores B^* is derived from the matrix B:

$$B^* = \mathtt{softmax}(\mathtt{B}) \quad \mathtt{where} \quad \mathtt{b}^*_{\mathtt{i},\mathtt{j}} = \frac{\exp(\mathtt{b}_{\mathtt{i},\mathtt{j}})}{\sum_{\mathtt{k}=1}^{\mathtt{q}} \exp(\mathtt{b}_{\mathtt{i},\mathtt{k}})} \in (0, 1).$$

A multi-output Neural Network Model

Let $\mathcal{M} = \{\tau_1, \tau_2, \dots, \tau_M\}$ be the set of maturities considered with $|\mathcal{M}| = M$.

We denote as:

- $\boldsymbol{y}_{t+1}^{(i)} \in \mathbb{R}^M$ the vector of the unknown yields related to the curve i at time $t+1$;
- $Y_{t-L,t}^{(i)} = \left(y_{t-l}^{(i)}(\tau)\right)_{0 \le l \le L, \tau \in \mathcal{M}} \in \mathbb{R}^{(L+1)\times M}$ the matrix of the yield rates for all maturities on the $L+1$ past dates.

We desire to learn the mapping

$$f : \mathbb{R}^{(L+1)\times M} \times \mathcal{I} \to \mathbb{R}^M \times \mathbb{R}^M \times \mathbb{R}^M$$

$$\left(Y_{t-\tau,t}^{(i)}, i\right) \mapsto \left(\widehat{\boldsymbol{y}}_{lb,t+1}^{(i)}, \widehat{\boldsymbol{y}}_{t+1}^{(i)}, \widehat{\boldsymbol{y}}_{ub,t+1}^{(i)}\right) = f\left(Y_{t-L,t}^{(i)}, i\right).$$

where, choosen a confidence level $\alpha \in [0, 1]$, we denote as

- $\widehat{\boldsymbol{y}}_{lb,t+1}^{(i)}$ the estimate of the lower quantile at level $\alpha/2$;
- $\widehat{\boldsymbol{y}}_{t+1}^{(i)}$ the estimate expected value or the median;
- $\widehat{\boldsymbol{y}}_{ub,t+1}^{(i)}$ the estimate of the upper quantile at level $1 - \alpha/2$.

Neural Network Model Architecture

We use a NN architecture that combines Embedding layers and some NN layers specifically designed for processing sequential data.

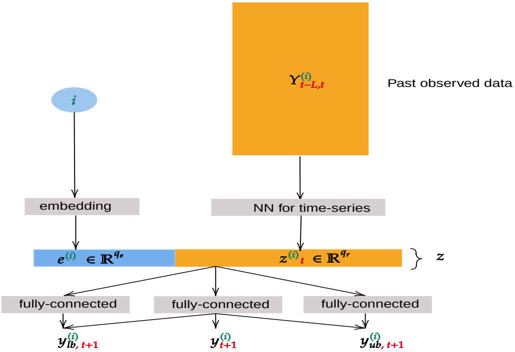


Figure: Graphical representation of the neural network architecture.

Predictions

The predictions are derived as:

$$
\begin{aligned}
\boldsymbol{y}_{t+1}^{(i)} &= g\left(\boldsymbol{b}_c + U_c \boldsymbol{e}^{(i)} + W_c \boldsymbol{z}_t^{(i)}\right) \\
\boldsymbol{y}_{lb,t+1}^{(i)} &= \boldsymbol{y}_{t+1}^{(i)} - \phi_+\left(\boldsymbol{b}_{lb} + U_{lb} \boldsymbol{e}^{(i)} + W_{lb} \boldsymbol{z}_t^{(i)}\right) \\
\boldsymbol{y}_{ub,t+1}^{(i)} &= \boldsymbol{y}_{t+1}^{(i)} + \phi_+\left(\boldsymbol{b}_{ub} + U_{ub} \boldsymbol{e}^{(i)} + W_{ub} \boldsymbol{z}_t^{(i)}\right)
\end{aligned}
$$

where $\phi_+ : \mathbb{R} \to (0, +\infty)$, and $\boldsymbol{b}_j, U_j, W_j, j \in \{c, lb, ub\}$ are network parameters.

This formulation ensures no-quantile crossing:

$$\boldsymbol{y}_{lb,t+1}^{(i)} < \boldsymbol{y}_{t+1}^{(i)} < \boldsymbol{y}_{ub,t+1}^{(i)}.$$

Some Remarks

(1) The model presents some connections with the affine models:

$$g^{(-1)}\left(\widehat{\boldsymbol{y}}_{t+1}^{(i)}(\tau)\right) = b_c + \left\langle \boldsymbol{u}_{c,\tau}, \boldsymbol{e}^{(i)} \right\rangle + \left\langle \boldsymbol{w}_{c,\tau}, \boldsymbol{z}_t^{(i)} \right\rangle .$$

Indeed, it has the constant-plus-linear structure and depends on the vector of variables $\boldsymbol{z}_t^{(i)}$ derived by the past observed data.

(2) We can also reformulate the equations of the quantile predictions:

$$\phi_+^{-1}\left(\widehat{y}_{t+1}^{(i)}(\tau) - \widehat{y}_{lb,t+1}^{(i)}(\tau)\right) = b_{lb} + \left\langle \boldsymbol{u}_{lb,\tau}, \boldsymbol{e}^{(i)} \right\rangle + \left\langle \boldsymbol{w}_{lb,\tau}, \boldsymbol{z}_t^{(i)} \right\rangle$$

emphasizing that we model, on the $\phi^{(-1)}$ scale, the difference between the central measure and lower quantile at a given maturity τ is an affine model.

Model Calibration

The network training requires to minimize the loss:

$$\begin{aligned}\mathcal{L}_{\alpha,\gamma}(W) &= \mathcal{L}^{(1)}_{\alpha,\gamma}(W) + \mathcal{L}^{(2)}_{\alpha,\gamma}(W) + \mathcal{L}^{(3)}_{\alpha,\gamma}(W) \\ &= \sum_{i,t,\tau} \ell_{\alpha/2}(\dot{y}^{(i)}_t(\tau) - \hat{y}^{(i)}_{lb,t}(\tau)) + \sum_{i,t,\tau} h_\gamma(\dot{y}^{(i)}_t(\tau) - \hat{y}^{(i)}_t(\tau)) + \\ &\quad \sum_{i,t,\tau} \ell_{1-\alpha/2}(\dot{y}^{(i)}_t(\tau) - \hat{y}^{(i)}_{ub,t}(\tau))\end{aligned}$$

where $\ell_\alpha(u), \alpha \in (0,1)$ is the pinball function:

$$\ell_\alpha(u) = \begin{cases} (1-\alpha)|u| & u \leq 0 \\ \alpha|u| & u > 0, \end{cases}$$

and $h_\gamma(u), \gamma \in \{1,2\}$ is:

$$h_\gamma(u) = \begin{cases} |u| & \gamma = 1 \\ u^2 & \gamma = 2, \end{cases}$$

Numerical Experiments: EIOPA data

- European Insurance and Occupational Pensions Authority Data
  - Maturities: $\mathcal{M} = \{\tau \in \mathbb{N} : 1 \leq \tau \leq 150\}$.
  - Period: Dec 2015 - Dec 2021.
  - 34 curves related to the governament bonds.

  Data Partitioning
  - Learning sample: Dec 2015-Dec 2020;
  - Test sample:Jan 2021-Dec 2021.
- NN architectures based on:
  - Long Short-Term Memory (LSTM) networks (YC_LSTM);
  - 1D Convolutional Neural networks (YC_CONV);
  - Self-Attention based networks (YC_ATT);
  - Transformers models (YC_TRAS).
- Benchmark models:
  - Dynamic Nelson-Siegel (NS);
  - Dynamic Nelson-Siegel-Svensson (NSS).
- Interval predictions at confidence level $\alpha = 0.95$.

Performance Measures

We compare the models in terms of:

$$MSE = \frac{1}{n} \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}} \sum_{\tau \in \mathcal{M}} (y_t^{(i)}(\tau) - \hat{y}_t^{(i)}(\tau))^2,$$

$$MAE = \frac{1}{n} \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}} \sum_{\tau \in \mathcal{M}} |y_t^{(i)}(\tau) - \hat{y}_t^{(i)}(\tau)|,$$

$$PICP = \frac{1}{n} \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}} \sum_{\tau \in \mathcal{M}} \mathbb{1}_{\{y_t^{(i)}(\tau) \in [\hat{y}_{t,lb}^{(i)}(\tau), \hat{y}_{t,ub}^{(i)}(\tau)]\}}$$

$$MPIW = \frac{1}{n} \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}} \sum_{\tau \in \mathcal{M}} \left(\hat{y}_{t,ub}^{(i)}(\tau) - \hat{y}_{t,lb}^{(i)}(\tau)\right).$$

EIOPA data

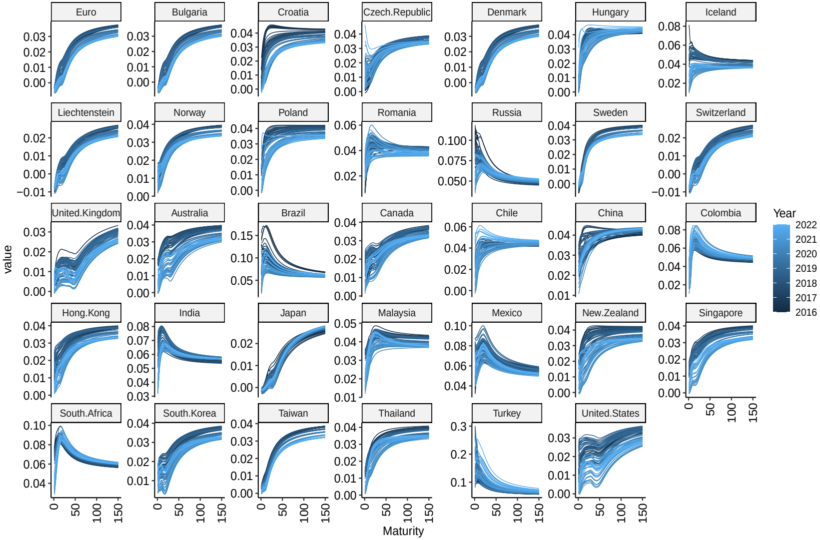


Figure: Yield Curve data provided by EIOPA.

Dynamic Nelson-Siegel Model

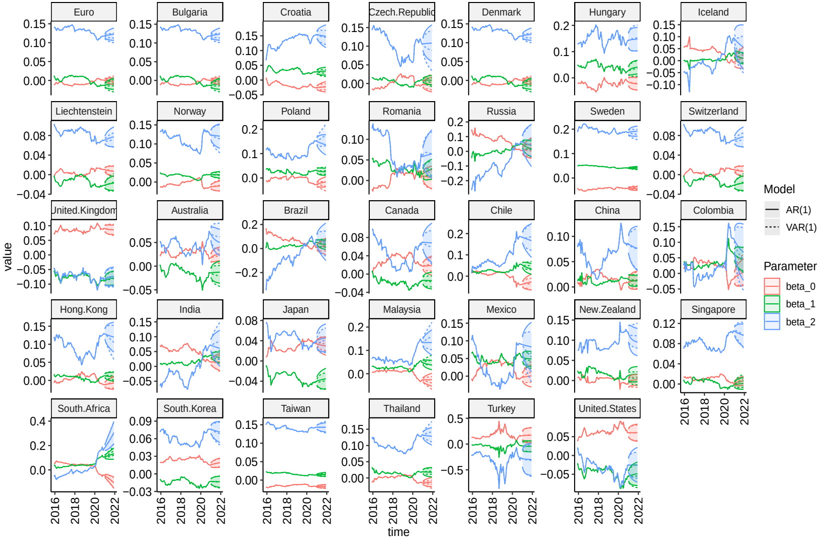


Figure: Dynamic Nelson-Siegel Model.

Forecasting Results

Model	MSE		MAE		PICP		MPIW	
	average	ensemble	average	ensemble	average	ensemble	average	ensemble
YC_ATT$_{\gamma=1}$	0.2947	**0.2887**	0.2667	**0.2616**	0.9154	0.9191	0.0106	0.0106
YC_ATT$_{\gamma=2}$	0.3663	0.3638	0.3463	0.3451	0.8528	0.8573	0.0105	0.0105
YC_CONV$_{\gamma=1}$	0.3778	0.3642	0.2975	0.2850	0.9035	**0.9235**	0.0115	0.115
YC_CONV$_{\gamma=2}$	0.4258	0.4244	0.3890	0.3884	0.8509	0.8530	0.0110	0.0110
YC_LSTM$_{\gamma=1}$	0.4272	0.4111	0.3164	0.2970	0.7757	0.8147	0.0093	0.0093
YC_LSTM$_{\gamma=2}$	0.3898	0.3697	0.3352	0.3198	0.6911	0.7081	**0.0084**	**0.0084**
YC_TRAN$_{\gamma=1}$	0.4308	0.4167	0.3313	0.3168	0.8371	0.8645	0.0113	0.0113
YC_TRANS$_{\gamma=2}$	0.4232	0.4124	0.4042	0.3987	0.5771	0.5760	0.0091	0.0091
NS_AR	0.7433		0.4496		0.9984		0.0540	
NS_VAR	0.4977		0.3492		0.7288		0.0080	
NSS_AR	0.5379		0.3709		0.9987		0.4253	
NSS_VAR	0.4626		0.3226		0.7462		0.0307	

Table: Out-of-sample performance of the different deep learning models in terms of MSE, MAE, PICP and MPIW; the MSE values are scaled by a factor of 10^5, while the MAE values are scaled by a factor of 10^2. Bold indicates the smallest value, or, for the PICP, the value closest to $\alpha = 0.95$.

Forecasting Results: Uncertainty

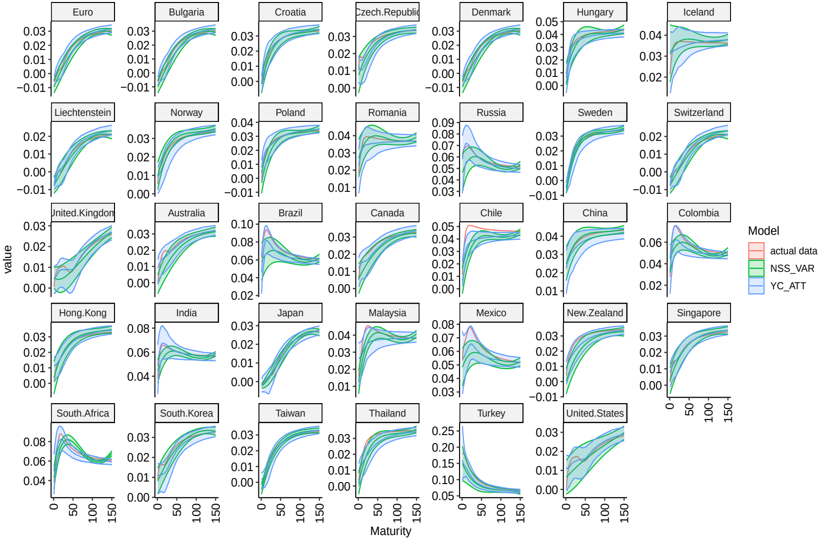


Figure: Interval predictions for $\alpha = 0.95$ related to the different yield curves.

Forecasting Results

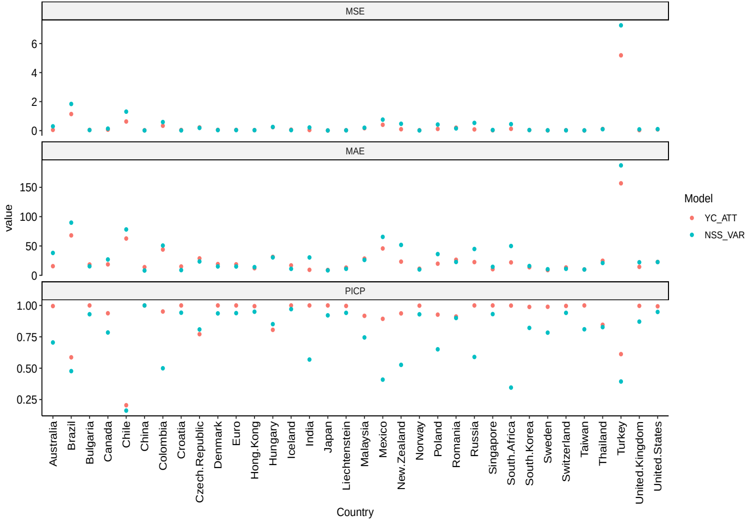


Figure: MSE, MAE, and PICP obtained by the YC_ATT and NSS_VAR models in the different countries.

Correlation between the NSS factors and the 4 PCs extracted from $(\boldsymbol{e}^{(i)}, \boldsymbol{z}_t^{(i)})$

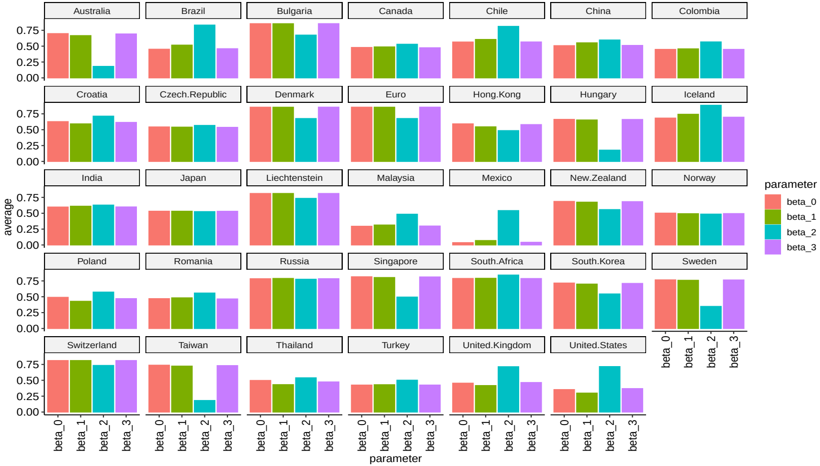


Figure: Linear correlation coefficients (in absolute value) of the four PCs derived from the learned features, represented as $(\boldsymbol{e}^{(i)}, \boldsymbol{z}_t^{(i)})$, with respect to the $\beta_t^{(i)}$ factors of the NSS model for the different yield curve families.

THANK YOU FOR YOUR ATTENTION !