Accurate and Explainable Mortality Forecasting with the LocalGLMnet

Francesca Perla[1] Ronald Richman[2] Salvatore Scognamiglio[1] Mario V. Wüthrich[3]

[1] Department of Management and Quantitative Studies, University of Naples, Parthenope;

[2] Old Mutual Insure and University of the Witwatersrand, Johannesburg, South Africa;

[3] RiskLab, Department of Mathematics, ETH Zurich.

Insurance Data Science Conference 2023 - London (UK)

Mortality Forecasting

- Mortality is declining in most of developed countries;
- several mortality models have been proposed: Lee and Carter (1992), Ranshaw and Haberman (2006), Cairns, Blake and Dowd (2006), Plat (2009).

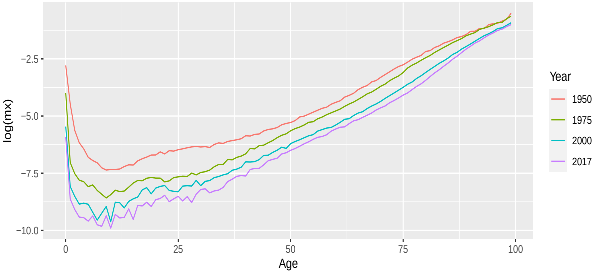


Figure: Italian mortality data (source: Human Mortality Database).

Neural Networks and Mortality Forecasting

- Deep Neural Networks (DNN) have been successfully applied to different tasks in actuarial science, including mortality modelling and forecasting. Several architectures have been investigated:
  - **Fully Connected Networks:** Hainaut (2019), Richman and Wüthrich (2021):
  - **Recurrent Networks:** Nigri et al. (2019), Perla et al. (2021), Lindholm and Palmborg (2022);
  - **1D Convolutional Networks:** Perla et al. (2021), Scognamiglio (2022);
  - **2D Convolutional Networks:** Wang et al. (2021), Schnurch and Korn (2022).
- Due to the complex structure of networks, it is difficult to determine the impact of inputs on the predictions.

Neural Networks and Mortality Forecasting

- Deep Neural Networks (DNN) have been successfully applied to different tasks in actuarial science, including mortality modelling and forecasting. Several architectures have been investigated:
  - **Fully Connected Networks:** Hainaut (2019), Richman and Wüthrich (2021):
  - **Recurrent Networks:** Nigri et al. (2019), Perla et al. (2021), Lindholm and Palmborg (2022);
  - **1D Convolutional Networks:** Perla et al. (2021), Scognamiglio (2022);
  - **2D Convolutional Networks:** Wang et al. (2021), Schnurch and Korn (2022).
- Due to the complex structure of networks, it is difficult to determine the impact of inputs on the predictions.

Can we benefit from the predictive accuracy of DNN while maintaining an explainable model structure?

Neural Networks

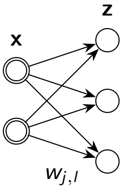
Let $\boldsymbol{x} \in \mathbb{R}^{q_0}$ be the vector of features, a fully connected (FC) layer of size $q_1 \in \mathbb{N}$ is a function

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

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 $\mathbb{R}^{q_0}$.



Neural Networks

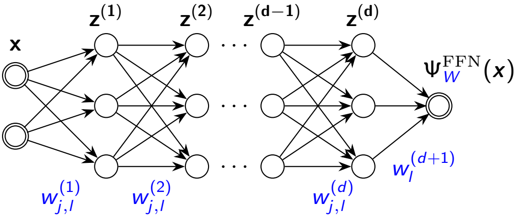
Deep neural networks compose multiple layers. For d layers of size $\boldsymbol{q} = \{q_k\}_{1 \le k \le d} \in \mathbb{N}^d$, the mapping reads:

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

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

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

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



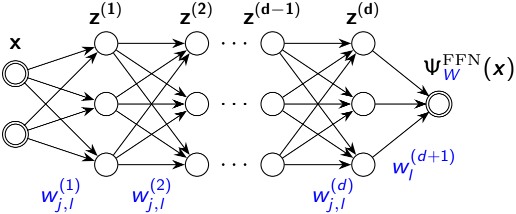
Neural Networks

The training of the NN induces the following optimisation:

$$\arg\min_{W} \mathcal{L}(y, \Psi_{W}^{\mathrm{FFN}}(\boldsymbol{x})),$$

where

- $\mathcal{L}(\cdot)$ is the chosen loss function;
- W is the vector of the neural network parameters.



The localGLMnet model of Richman and Wüthrich (2023)

Let Ψ_W be a neural network with output dimension equal to the input dimension q_0:

$$\Psi_W : \mathbb{R}^{q_0} \to \mathbb{R}^{q_0}, \qquad \boldsymbol{x} \mapsto \Psi_W(\boldsymbol{x}),$$

having network weights W. The *LocalGLMnet* regression function is defined by

$$\boldsymbol{x} \;\mapsto\; \mu_{W,\beta_0}(\boldsymbol{x}) \;\overset{\text{def}}{=}\; g^{-1}\left(\beta_0 + \boldsymbol{\beta}(\boldsymbol{x})^\top \boldsymbol{x}\right),$$

where $g : \mathbb{R} \to \mathbb{R}$ is the link function, $\beta_0 \in \mathbb{R}$, and $\boldsymbol{\beta}(\boldsymbol{x}) = \Psi_W(\boldsymbol{x})$.

(1) If $\beta_j(\boldsymbol{x}) \equiv \beta_j$ is not feature dependent.

(2) If $\beta_j(\boldsymbol{x}) \equiv 0$, term $\beta_j(\boldsymbol{x})x_j$ is dropped altogether.

(3) If $\beta_j(\boldsymbol{x}) = \beta_j(x_j)$, term $\beta_j(x_j)x_j$ does not interact with any other terms $x_{j'}$, $j' \neq j$.

(4) Interactions can be studied by considering the gradient of $\beta_j(\boldsymbol{x})$

$$\nabla \beta_j(\boldsymbol{x}) = \left(\partial_{x_1}\beta_j(\boldsymbol{x}), \ldots, \partial_{x_{q_0}}\beta_j(\boldsymbol{x})\right)^\top \;\in\; \mathbb{R}^{q_0}.$$

A multi-output extension of localGLMnet model

Let $X \in \mathbb{R}^{p\times q}$ be the matrix of input data, and Ψ_W be a neural network with output dimension equal to the input dimension $\mathbb{R}^{p\times q}$:

$$\Psi_W : \mathbb{R}^{p\times q} \to \mathbb{R}^{p\times q}, \qquad X \mapsto \Psi_W(X),$$

having network weights W. The *multi-output LocalGLMnet* regression function is defined by

$$X \ \mapsto \ \mu_{W,\boldsymbol{\beta}_0}(X) \ \overset{\text{def}}{=} \ g^{-1}\left(\boldsymbol{\beta}_0 + \mathbf{1}_p^\top \left[B(X) \odot X\right]\right) \ \in \ \mathbb{R}^q,$$

where $\odot$ is the Hadamard product, $\mathbf{1}_p = (1,\ldots,1)^\top \in \mathbb{R}^p$, $g^{-1} : \mathbb{R} \to \mathbb{R}$ is applied in an element-wise manner, $\boldsymbol{\beta}_0 \in \mathbb{R}^q$ is a vector of bias terms, and where we set regression attention matrix $B(X) = \Psi_W(X)$.

Let $\mathcal{X} = \{x \in \mathbb{N}_0 : 0 \leq x \leq \omega\}$ be the set of ages considered.

We denote:

- $\boldsymbol{m}_{t+1}^{(i)} \in \mathbb{R}^{\omega+1}$ the vector mortality rates of a population i in year $t+1$;
- $M_{t-\tau,t}^{(i)} \in \mathbb{R}^{(\tau+1)\times(\omega+1)}$ the matrix of the mortality rates for all ages in the $\tau+1$ past years.

We desire to learn the mapping

$$f : \mathbb{R}^{(\tau+1)\times(\omega+1)} \to \mathbb{R}^{\omega+1} \qquad M_{t-\tau,t}^{(i)} \mapsto \widehat{\boldsymbol{m}}_{t+1}^{(i)} = f\left(M_{t-\tau,t}^{(i)}\right).$$

Applying the multi-output localGLM regression function:

$$M^{(i)}_{t-\tau,t} \mapsto \mu_{W,\boldsymbol{\beta}_0}(M^{(i)}_{t-\tau,t}) \overset{\text{def}}{=} g^{-1}\left(\boldsymbol{\beta}_0 + \mathbf{1}_p^\top \left[B(M^{(i)}_{t-\tau,t}) \odot M^{(i)}_{t-\tau,t}\right]\right) \in \mathbb{R}^{\omega+1},$$

where $\boldsymbol{\beta}_0 \in \mathbb{R}^{\omega+1}$ and $B(M^{(i)}_{t-\tau,t}) \in \mathbb{R}^{(\tau+1)\times(\omega+1)}$. Rewriting the model for a single age:

$$\widehat{m}^{(i)}_{j,t+1} = \left(\mu_{W,\boldsymbol{\beta}_0}(M^{(i)}_{t-\tau,t})\right)_j = g^{-1}\left(\beta_{0,j} + \boldsymbol{\beta}_j(M^{(i)}_{t-\tau,t})^\top \boldsymbol{m}^{(i)}_{(t-\tau,t),j}\right).$$

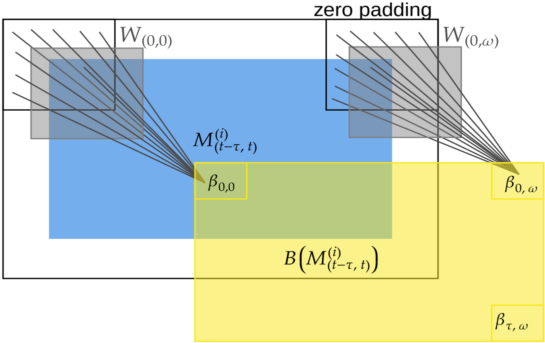
It can be rearranged as:

$$g\left(\widehat{m}^{(i)}_{j,t+1}\right) = \beta_{0,j} + \sum_{s=0}^{\tau} \beta_{s,j}(M^{(i)}_{t-\tau,t})\, m^{(i)}_{j,t-s},$$

that is AR($\tau + 1$), with varying coefficients derived from $M^{(i)}_{t-\tau,t}$.

How do we derive $B(M^{(i)}_{t-\tau,t})$?

We derive the *attention coefficients* $B(M^{(i)}_{t-\tau,t})$ by applying a **2D locally connected** layer to the matrix $M^{(i)}_{t-\tau,t}$.



where $W_{(s,x)} \in \mathbb{R}^{d_1 \times d_2}$ for $s = 0, 2, \dots, \tau, x = 0, 1, \dots, \omega$ are weight matrices.

Applying the localGLMnet: Human Mortality Database

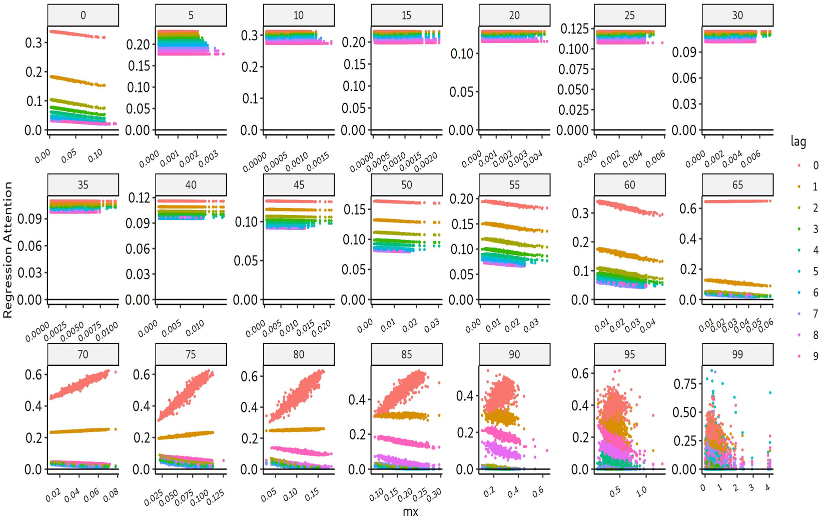
- Data Source: Human Mortality Database:
  - Ages $\mathcal{X} = \{x \in \mathbb{N}_0 : 0 \leq x < 100\}$;
  - Years $\mathcal{T} = \{t \in \mathbb{N} : 1950 \leq t \leq 2016\}$;
  - Populations $|\mathcal{I}| = 76$.
- Data Partitioning:
  - Learning data $\mathcal{T}_{learn} = \{t \in \mathbb{N} : 1950 \leq t \leq 1999\}$;
  - Test data $\mathcal{T}_{test} = \{t \in \mathbb{N} : 2000 \leq t \leq 2016\}$.

The networks are trained by minimising:

$$\underset{W}{\arg\min}\, \mathcal{L}(W) = \underset{W}{\arg\min} \sum_i \sum_x \sum_t \left(m_{x,t}^{(i)} - \widehat{m}_{x,t}^{(i)}\right)^2.$$

where W denote the vector of network parameters.

The attention coefficients $\beta_{s,x}(M^{(i)}_{t-\tau,t})$



Forecasting Accuracy

model	forecasting MSE	# parameters
LC	5.4659	18.698
LCCONV[1]	2.2936 (0.0282)	26.996
LocalGLMnet	2.1985 (0.0149)	26.000

Table: Average and standard deviation of the out-of-sample forecasting MSEs and number of trainable parameters of the LC, LCCONV, LocalGLMnet and models; the MSEs are multiplied by 10^4.

[1] Perla, F., Richman, R., Scognamiglio, S., & Wüthrich, M. V. (2021). Time-series forecasting of mortality rates using deep learning. Scandinavian Actuarial Journal, 2021(7), 572-598.

United States Mortality Database

- Data Source: United States Mortality Database:
  - Ages $\mathcal{X} = \{x \in \mathbb{N}_0 : 0 \leq x < 100\}$;
  - Years $\mathcal{T} = \{t \in \mathbb{N} : 1959 \leq t \leq 2017\}$;
  - Populations $|\mathcal{I}| = 102$.
- Data Partitioning:
  - Learning data $\mathcal{T}_{learn} = \{t \in \mathbb{N} : 1959 \leq t \leq 1999\}$;
  - Test data $\mathcal{T}_{test} = \{t \in \mathbb{N} : 2000 \leq t \leq 2017\}$.

We test three localGLMnet models:

- **LocalGLMnet_HMD** trained on the HMD data;
- **LocalGLMnet_transfer** trained on HMD data and the weights are further fine-tuned on the USMD data;
- **LocalGLMnet_USMD** directly trained on the USMD data.

Forecasting Accuracy on USMD data

model	forecasting MSE	# pssarameters
LC	1.1848	24.684
LCCONV	0.4938 (0.0268)	27.061
LocalGLMnet_HMD	0.4075 (0.0102)	26.000
LocalGLMnet_transfer	0.2986 (0.0039)	26.000
LocalGLMnet_USMD	0.3134 (0.0100)	26.000

Table: Average and standard deviation of the forecasting MSEs and number of trainable parameters of the LC, LCCONV, LocalGLMnet_HMD, LocalGLMnet_transfer and LocalGLMnet_USMD models; the MSEs are multiplied by 10^4.

Conclusions

- The LocalGLMnet model proposed by Richman and Wüthrich (2023) can be adapted to the time series forecasting task;
- accurate forecasts can be obtained without losing model explainability;
- transfer learning mechanisms can further improve forecasting accuracy.

For comments or suggestions:

salvatore.scognamiglio@uniparthenope.it