NOVA IMS Information Management School

# MEGI

Mestrado em Estatística e Gestão de Informação
Master Program in Statistics and Information Management

## A Deep Learning integrated mortality model for Longevity Swap pricing

Alberto Di Salvo

Thesis presented as partial requirement for obtaining the Master's degree in Statistics and Information Management

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

**NOVA Information Management School**
**Instituto Superior de Estatística e Gestão de Informação**
Universidade Nova de Lisboa

# A DEEP LEARNING INTEGRATED MORTALITY MODEL FOR LONGEVITY SWAP PRICING

by

Alberto Di Salvo

Theis presented as partial requirement for obtaining the Master's degree in Information Management/ Master's degree in Statistics and Information Management , with a specialization in Risk Management and Analysis

**Advisor / Co Advisor:** Prof. Dr. Jorge Miguel Ventura Bravo

June 2022

# ACKNOWLEDGEMENTS

This dissertation, which establishes the culmination of an important stage, could not be fully accomplished without the support of my friends, family, and adviser whose I dedicate this section.

First, I would like to express my appreciation to my advisor, Professor Jorge Bravo for his full availability and guidance throughout this newly research.

Then I would like to thank family and friends in Rome and Lisbon. Especially to my parents, brother and my grand mothers, for all the contribution to this journey that helped during my stay in Lisbon.

In the end would like to thank all the research staff of Human Mortality Database for the great work and data avilability they provide for the studyes in this field.

**Abstract**

This research empirically investigates the usage of Recurrent Neural Networks (RNN) to improve the accuracy of mortality rates forecasting within the context of Longevity linked securities pricing. The benchmark model in the mortality field is the classical Lee-Carter; the forecasting procedure of these model is often conducted with ARIMA models. I consider a fixed forecasting time horizon in order to compare the performance of Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) with different hyperparameter and data input choices against that produced by the best fitted ARIMA models. The results are then applied to Longevity Swap pricing in order to better estimates the premium of the derivatives contracts. The investigation is conducted for six countries, using mortality data from 1950 onwards, differentiating by gender. The research shows how RNN outperform the classical ARIMA models in the forecasting procedure. Although the advantages of RNN's techniques are strictly bounded to the set of hyperparameter used for the comparison; the outcomes of such approaches can vary greatly using different input choices. In the end the results shows that an RNN approach can bring significant changes to the price of Longevity Linked securities. The research is in the first place one of the few to test the forecasting accuracy of Deep Learning methods accounting for alternative methodological, hyperparameter and data input choices. Afterwards the investigation demonstrate the necessity of revisit the classical mortality models in order to better estimates prices of derivatives contracts that are very useful in the context of Longevity risk.

**Keywords:** Mortality, Deep Learning, Long short-term memory, Gated Recurrent Unit, Lee-Carter model, Longevity risk, Longevity swap, Longevity swap pricing.

# Contents



# List of Figures



# List of Tables

# 1 Introduction

Mortality modeling is essential in economy, demography and in life and social insurance, because mortality rates impact insurance liabilities, prices of insurance products and social benefit schemes. Insurance companies allow individuals to trade uncertainty for certainty by transferring their own risk to the insurer in exchange for a fixed premium. An insurer sets the price for a policy before it's actual cost is revealed. So because of this phenomenon, known as the reverse production cycle, it is of fundamental importance that an insurer properly assesses the risks in its portfolio.

A fundamental longevity improvement is occurring in the past decades, the causes are several. In this regard there is often referred to Longevity Risk: the risk of insured capital population living "longer" than expected. Evidence of longevity improvements can be found directly in the increase of life expectancy at all ages, showing a strong positive trend in the survival probabilities. Is clear how Longevity risk is nowadays an important topic and challenge for insurance companies and especially for pension funds. To hedge against this long-term risk, financial and insurance instruments have been proposed and placed on the market since 2004; the solution ranges from capital market instruments to simple insurance-based solutions. Focusing on the shield provided by the capital market, Longevity-linked securities assume a particular relevance. In fact Longevity-linked securities have some advantages over the classic insurance-based solution (e.g. reinsurance coverage); hence capital market based solution are prevalent in the hedging of this risk, leading to an intense use of such contracts. Despite the capacity of financial markets to absorb longevity risk, some instruments have many challenges on the calculation of their fair values [2]. But there are some issues and difficulties in the pricing of these derivatives contract: in the first place mortality is not a traded asset, so is impossible to value these contract through non-arbitrage rules. Then the problem is that since the cash flows in these contract are directly related to forecasted mortality rates, their prediction require an adequate modelling scheme. Therefore, since mostly all this instruments assume value in force of the mortality rate predictions, seems natural to revise some of the components of the well known mortality models. Among several stochastic mortality models, the Lee-Carter [13] can be considered the milestone and benchmark in this field. The main difficulties using this model are present in the forecasting procedures: in order to catch longevity improvements, the model need to consider both short both long term pattern in mortality. Hence this issue is mostly focused on the stochastic process, known as the time component. This component, representing the time series, is often modelled as the best fitted ARIMA calibration.

This research is focused on a different approach to time series modelling, a Deep Learning integrated model: the idea is to model the time parameter $\kappa_t$ with Recurrent Neural Networks, in order to catch both the long and short term trend in mortality. These kind of techniques are already widely used in other field because they have the ability to consider a large number of possible solution and inputs. The objective is therefore to consider this Artificial In-

telligence(AI) approach and to see the differences with the classical forecasting through the ARIMA arrangement. With a different pattern for the value of the stochastic process $\kappa_t$, new mortality rates $m_x$ will be produced. Therefore the difference will be assessed, with the aim of produce a different prediction in mortality rates. The applied context of the research will be the usage of a different forward mortality rate into the context of longevity risk management. Specifically the research will be focused on the pricing of Longevity Swap contracts, comparing the AI adjusted price with the benchmark estimation produced with a simple Lee-Carter mortality rates forecast. Hence the aim of the research is too evaluate and assess the impact of Deep Learning techniques on the pricing of Longevity Swap contracts, since the fair value of such instruments is very sensitive to the predicted underlying asset (mortality rate).

The thesis will be structured as follows:

1. In the first section I introduce the classical Lee-Carter mortality model.
2. Then present the Neural Network, Recurrent Neural Network and in the end two architecture that fit this specific forecast problem.
3. The next chapter will be focused on numerical applications, performing some forecasting experiments on different countries.
4. After the evaluation on the differences using Deep Learning techniques instead of ARIMA, I apply the results to Longevity-linked securities pricing.
5. In the end I show my results and conclusions.

# 2 Lee-Carter model and Artificial Intelligence algorithms

The existing literature about stochastic mortality refers in general to the well known Lee-Carter model (LC) [13], and to it's several extensions and modification. In order to forecast futures mortality rates, various stochastic models have been developed. I decided to adopt the first formulation of the Lee-Carter although several modifications have been proposed to improve the estimates of this model; the latter remains a benchmark in the forecasting procedures of future mortality rates.

The Lee-Carter model in it's first version (1992), returns the the central death rate $m_x(t)$ through the *log*-bilinear relation:

$$\log(m_{x,t}) = \alpha_x + \beta_x \kappa_t + \epsilon_{x,t} \tag{1}$$

where $\alpha_x$ describes the average age-specific trend of mortality, while $\beta_x$ denotes the deviation from the average mortality rate when $\kappa_t$ varies. The univariate index $\kappa_t$ represent changes in the levels of mortality over time, and is the keystone of the formula, explaining increases and decreases in mortality rates over time.

Finally $\epsilon_{x,t}$ is the homoschedastic error term, representing some of the aspects in mortality not captured by the model. In the literature, it is assumed that the parameters are subject to constraints that guarantee that identifiability of the model:

$$\sum_{t=t_1}^{t_n} \kappa_t = 0 \qquad \sum_{x=x_1}^{x_m} \beta_x = 1 \tag{2}$$

where $x_1$ and $x_m$ are respectively the minimum and maximum age considered for the fitting, and $[t_1, t_n]$ are the calendar years object of analysis. The model's parameters are originally estimated through a Singular Value Decomposition (SVD), with a two stage procedure. In the first place the SVD is applied to $\ln m_x(t) - \alpha_x$ to estimate $\beta_x$ and $\kappa_t$. Then $\kappa_t$ is refitted so that the observed number of deaths is equal to the observed one. After the model's fitting, in order to forecast the mortality rates, it will be necessary to predict the value of the stochastic process $\kappa_t$. There is not an unique method for the forecast procedure; among the most used and successful there are the ARIMA models [1]. Usually $\kappa_t$ is modeled as an ARIMA(0,1,0):

$$\kappa_t = \kappa_{t-1} + \delta + \epsilon_t \tag{3}$$

where $\delta$ is the drift parameter and $\epsilon_t$ are the heteroskedastic error terms, normally distributed with null mean and variance $\sigma_k^2$. Hence after the forecasting procedures, in order to obtain the central death rates $m_{x,t}$, it will just be necessary to solve the general equation with the SVD fitted parameters $\alpha_x, \beta_x$ and with the forecasted stochastic process $\kappa_t$.

The Lee-Carter model is the benchmark model in this field; many other extension have been proposed after the first version of the LC. Among the most relevant there is the Cairns-Blake-Dowd model [5], which enrich the estimation of the mortality rates with two stocastich process. Further research to incorporate the cohort effect, like the Age-Period-Cohort model [18] where Renshaw *et al.* incorporate the LC model with the introduction of a cohort dependent parameter. In the past years several modification of the LC have been proposed, in general to refer to these proposed research the literature refers the Generalised Age-Period-Cohort (GAPC) stochastic mortality models. An interesting application to GAPC models can be fund in the research of Bravo *et al.* [4]; here instead of a single "candidate" model, the mortality rates are obtained through a novel adaptive Bayesian Model Ensemble of heterogeneous parametric generalized age-period-cohort stochastic mortality models. Apart from the very interesting applications proposed here, for the purpose of this research I focus solely on the Lee-Carter. In the next section I introduce the Neural Network architecture, that will serve as a starting point to introduce the AI techniques object of the research.

## 2.1 Neural Networks

A Neural Network (NN) is a mathematical model that, as in the first version [15], aims to replicate the behaviour of human brain's neural networks. This architecture is composed by neurons, synaptic connections that link the neurons, and learning algorithms. Usually a NN is composed by three types of layers, called input layer, hidden layer and output layer and each one of these comprises several neurons. In the artificial network every unit obtains "proportional"(weighted) information via synaptic links from many other well connected units at the same time returning an output through the usage of an activation function that transforms these proportional totals of the input signals. So considering a single neuron $\mathbf{H}$, its output is defined by [16]:

$$out\mathbf{H} = \mathbf{\Phi}(\mathbf{w^T x} + b) \tag{4}$$

where $\mathbf{x} \in \mathbb{R}^d$ is the input, $\mathbf{w} \in \mathbb{R}^d$ is the related synaptic weight, $d \in \mathbb{N}$ is the number of input signals, $b \in \mathbb{R}$ represent the bias and $\mathbf{\Phi}$ is the activation function that must be differentiable. Then if $h = 1, 2, ..d$ is the number of

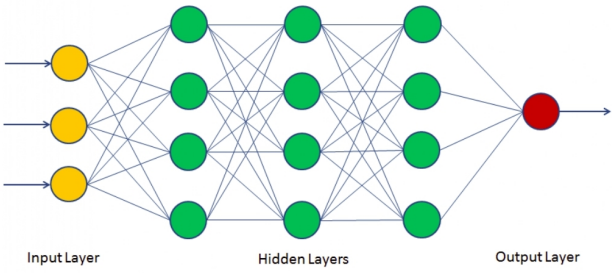


Figure 1: A neural Network scheme.

hidden layers in the network, the output $\mathbf{outH} \in \mathbb{R}^{n_h}$ of a generic hidden layer composed by $n_h$ neurons is defined by:

$$out\mathbf{H} = \mathbf{\Phi}(W^T\mathbf{x} + b) \tag{5}$$

where $W \in \mathbb{R}^{d \times n_h}$ is the weight matrix and $b \in \mathbb{R}^{n_h}$ is the biases vector. Hence in the context of a regression problem, where $d \in \mathbb{N}$ is the number of hidden layers, then output $\hat{y} \in \mathbb{R}$ is processed and obtained as:

$$out\mathbf{H_1} = \mathbf{\Phi_1}(W_1^T\mathbf{x} + b_1) \tag{6}$$

$$out\mathbf{H_2} = \mathbf{\Phi_2}(W_2^T out\mathbf{H_1} + b_2) \tag{7}$$

....

$$\hat{y} = out\mathbf{H_d} = \mathbf{\Phi_d}(W_d^T out\mathbf{H}_{d-1} + b_d) \tag{8}$$

where $W_1, W_2, ..., W_d$ are weight matrices, $b_1, b_2, ..., b_d$ are bias vectors, and $\Phi_1, \Phi_2, ..., \Phi_d$ are the activation functions. It is important to notice that the dimension of the weight matrices and the bias vectors depend on the number of units in the hidden layers. As matter of fact by increasing the number of hidden layers, the level of abstraction of the input data increases.

Going further, the way NN training works has to be addressed to the scope of minimizing the error rather than better predict the response variable. For this reason is very important to decide which function will be designated to asses the loss in each cycle, also in order to understand how fast the NN works. So the training involves an unrestricted optimization where the goal is to minimize a loss function. The most used loss function is the Mean Squared Error:

$$E = \frac{\sum_{i=1}^{N}(y_i - \widehat{y}_i)^2}{N} \tag{9}$$

the quantity $E$ aims to measure the average difference between the observed values $y_i$ and the predicted ones $\widehat{y}_i$, where the lower value of $E$ represent the better functioning of the Network. The loss function depends on the matrices of weights $W_1, W_2, ..., W_d$ and on the respective biases $b_i$. The scope of the Loss function is to hint the NN with the best set of synaptic weights. Therefore the NN will try to find the synaptic weights $W_i$ that minimize the desired loss function. In the context of training algorithms, the back-propagation [20] is the most used one. This algorithm compares the predicted values of the response variable with the observed ones (benchmark); it assesses the difference (loss function wise) and then update the synaptic weights accordingly by back-propagating the gradient of the loss function $E$. So basically, in the forward step it computes the predictions $\widehat{y}_i$ setting the synaptic weights $W_i$, while in the back-forward step it change the weights in order to minimize the loss function $E$. A Neural Network performs this type of operations iteratively a lot of times till it gets the synaptic weights that produce a minimum for the loss function. Mathematically speaking, the back-propagation algorithm assesses and find the synaptic weights for the last layer with the usage of the Delta rule [21]:

$$\triangle W_d = -r\frac{\partial E}{\partial W_d} \tag{10}$$

where $r$ is the learning rate. While going backwards, to find and change the weight for the other layer, the algorithm repeat it self recursively through the chain derivation rule, so that:

$$\triangle W_{d-1} = -r\frac{\partial E}{\partial out\mathbf{H_{d-1}}}\frac{\partial out\mathbf{H_{d-1}}}{\partial W_{d-1}} \tag{11}$$

Figuratively speaking, the concept behind the Gradient Descent Algorithm is similar to climbing down a hill, where the downhill ends whenever a global or local minimum of the loss function is reached. In each epoch, the search of

new weight moves in the opposite direction of the gradient; the amplitude and direction of these "movements" are computed on the basis of the slope of the gradient and on the value of the learning rate $r$. The choice of the learning rate for some NN can be very important, since small values involves too many iterations while a large values could lead to the incapacity of convergence to a global minimum.

## 2.2 Recurrent Neural Networks

Although NN represents good algorithm for analysis problems, it result not efficient when comes to analyze time sequential data, as can be the time series $\kappa_t$. As a matter of fact, simple NN can not retrive all the inside pattern in the sequential data from past predictions, resulting in a constant loss of information. While the memory of Recurrent Neural Network (RNN) algorithms allows them to learn more about long-term dependencies in data and understand the whole context of the sequence while making the next prediction. For this reason, the usage of RNN's architecture is more adequate and efficient for the purpose of this work. A Recurrent Neural Network [20] is a class of artificial neural network that includes neurons connected together in a loop. Typically, the main innovation in this technique lies in the fact that the output of an hidden layer $d$ is used as input for the $d-i$ layer. This interconnection between layers either allows the use of one of the layers as a state memory, and also providing a temporal sequence of values as input, it allows to model a temporal dynamic behavior dependent on information received at previous time steps. The RNN's are a broad category of algorithms, all with the characteristic of a "connection" between the nodes, which is the key to analyze sequential data. But in general a simple RNN is not adequate to perform time series forecasting for mortality rates, since the time series are extensive and there are both long and short term pattern to catch. Besides this problem, in general the RNN's have the major problem of gradients vanishing: when the weights changes, but becoming smaller at each time-step till they have no effect on the response variable. So in recurrent neural networks, layers that get a small gradient update stops learning. Those are usually the earlier layers. So since these layers do not learn, RNN's can forget what it has seen in longer sequences, thus having a short-term memory. So the network gradually loses its ability to learn from the past, and become operationally inadequate for the analysis of long time series data.

## 2.3 Long short-term memory techniques

To overcome this problem we can make use of Long Short-Term Memory(LSTM) architectures [10]. An LSTM is a type of Recurrent Neural Network that allows the consideration of both long and short term memory. The upgrade within LSTM is made through internal mechanisms called gates that can regulate the flow of information at each step. These gates can learn which data in a sequence is important to keep or to drop away, in order to pass relevant information down to the chain of layers. Therefore the main innovation of LSTM's are

the cells state, and their gates. The cell state are basically a transport channel that bring relative information all the way down the sequence chain; these can be view as the "memory" of the network. In this way even information from the firsts time steps can arrive to the later time steps, filling the short-term memory problem of simple RNN. Thus as we move forward with the steps, there will be some new information, and also some parts of the existing cell state will be removed. These two operation on the cell state are performed through the usage of gates. Gates are neural networks that assesses whether to keep or drop information from the previous step. A plain vanilla LSTM unit (LSTM block), is composed of three gates, their interconnections and the resulting cell state (memory cell). The block runs as follows: it receives as initial information flow the current input $x_t \in \mathbb{R}$, the previous short-term output $h_{t-1} \in \mathbb{R}^{n_h}$ and the previous state of (long-term) memory cell $c_{t-1} \in \mathbb{R}^{n_h}$ . The information is then processed by the three gates , named, respectively, forget gate, input gate and output gate, and auxiliary NNs functions helpful in the regularization of information flow. First, we have the forget gate; this gate decides what information should be thrown away or kept. Information from the previous hidden state $h_{t-1}$ and information from the current input is passed through the sigmoid function $\sigma$. The sigmoid function has a range in $(0, 1)$. The closer to 0 means to forget the information, and the closer to 1 means to keep it. While in

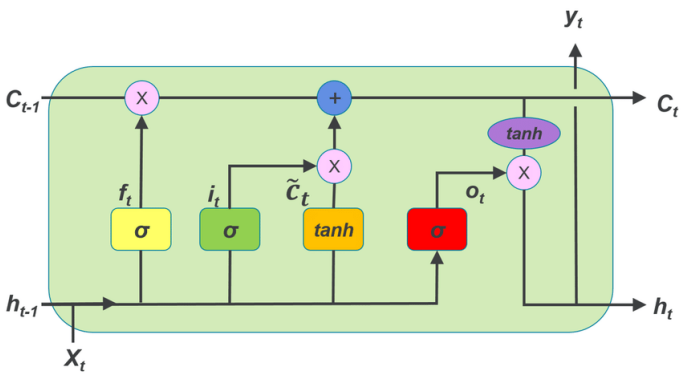


Figure 2: Plain vanilla LSTM block representation.

order to update the memory, the LSTM block uses the input gate. In first instance, we pass $h_{t-1}$ and the current input $x_t$ into the sigmoid function $\sigma$ that assess which values will be updated by transforming the values in the range $(0, 1)$. Then the information also pass $h_t - 1$ and $x_t$ into the hyperbolic tangent function (tanh)to squish values in range $(-1, 1)$ to help regulate the network. Then it multiply the tanh output with the sigmoid output. The sigmoid output will decide which information is important to keep from the tanh output. After these stages, there is enough information to calculate the memory cell. First, the cell state gets multiplied point-wise by the forget vector. This calculation has the possibility of dropping values in the cell state if it gets multiplied by

values near 0 and vice-versa. Then we take the output from the input gate and do a point-wise addition which updates the cell state to new values that the neural network finds relevant. That gives us our new memory. In the end there is an output gate, that decides what the next hidden state $h_t$ will be. Therefore, in the first place we pass the previous hidden state $h_{t-1}$ and the current input $x_t$ into the sigmoid function. Then we pass the newly modified cell state to the tanh function. We multiply the tanh output with the sigmoid output to decide what information the hidden state should carry. The output is the hidden state. The new cell state and the new hidden is then carried over to the next time step. Analytically, referring to an hidden layer composed of:

$$n_h \in \mathbb{N} \text{ neurons}, \quad W \in \mathbb{R}^{d \times n_h}, \quad U \in \mathbb{R}^{n_h \times n_h} \tag{12}$$

weight matrices, we refer to the following equations for the functioning of a general LSTM network:

$$\mathbf{f_t} = \sigma(W_f \mathbf{x_t} + U_f \mathbf{h_{t-1}} + b_f) \tag{13}$$

$$\mathbf{i_t} = \sigma(W_i \mathbf{x_t} + U_i \mathbf{h_{t-1}} + b_i) \tag{14}$$

$$\mathbf{o_t} = \sigma(W_o \mathbf{x_t} + U_o \mathbf{h_{t-1}} + b_o) \tag{15}$$

$$\mathbf{z_t} = \Phi(W_z \mathbf{x_t} + U_z \mathbf{h_{t-1}} + b_z) \tag{16}$$

The forget gate output $\mathbf{f_t}$, defined in equation 13, is such that the information from the previous cell state and the one coming from the current input are mixed in a nonlinear way by a sigmoid activation function. Therefore, the output can only assume value between 0 and 1, forgetting or keeping the state of the previous block. Afterwards, $\mathbf{f_t}$ is mixed by a point-wise product with the previous state of memory $c_{t-1}$. The input gate $\mathbf{i_t}$, defined in equation 14, also uses a sigmoid activation, allowing for deciding when information received should be updated. The output gate $\mathbf{o_t}$ , described in equation 15, has the role to prevent the transmission of non-significant memory content stored information to the other blocks. For this purpose, a sigmoid function is used in order to pass relevant memory information. In order to regularize the flow of processed data, the input gate it is combined with that obtained from the associated auxiliary NN $\mathbf{z_t}$ as in equation 16. Then we define the processing of the entire input block, which participates in formulation of the current state of memory cell, as follows:

$$\mathbf{c_t} = \mathbf{c_{t-1}} \circ \mathbf{f_t} + \mathbf{i_t} \circ \mathbf{z_t} \tag{17}$$

To obtain the current output, is necessary a combination between $\mathbf{s_t}$, a function of $\mathbf{c_t}$ expressed as $\mathbf{s_t} = \Phi(\mathbf{c_t})$, and the upshot of auxiliary NN associated to output gate $\mathbf{o_t}$:

$$\mathbf{outH} = \mathbf{s_t} \circ \mathbf{c_t} \tag{18}$$

And in the end the output $\mathbf{outH} \in \mathbb{R}^{n_h}$ is passed to the next layer and became the short memory $h_t = \mathbf{outH}$ for the next step.

## 2.4 Gated Recurrent Unit architecture

In the context of Recurrent Neural Networks , in the past few years took place another Memory-based network, the Gated recurrent Unit (GRU) [6] developed in 2014. The GRU was developed always to solve the Gradient vanishing problem, and as an alternative to LSTM, since it represents a less laborious version of the latter. In the GRU architecture, the memory is just composed by two gates, update and reset rate. The reset gate is bounded to the short-term memory of the recurrent neural network, represented analytically with:

$$r_t = \sigma_g(W_r x_t + U_r z_{t-1} + b_r) \tag{19}$$

where $z_{t-1}$ are the activations (outputs)at time-step $(t-1)$, while $W_r$ $U_r$ and $b_r$ are parameter matrices and vector. So basically the forget rate assess how much information needs to be forgotten by the model. The second gate is the update gate is responsible for the long-term memory and it is driven by the formula:

$$u_t = \sigma_g(W_u x_t + U_u z_{t-1} + b_u) \tag{20}$$

The update gate is needed from the model to determine how much of the past information (from previous time-steps) needs to be passed along to the future.

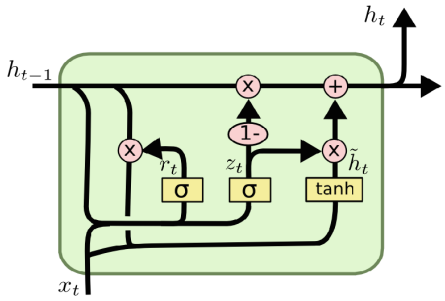


Figure 3: A representation of a Gated Recurrent Unit block structure and is internal information forward flow.

Once defined, the reset and update gate, these variables will determine the state of the activations at time $t$ given $z_{t-1}$:

$$z_t = r_t \circ z_{t-1} + (1 - r_t) \circ \phi(< W, x_T > + u_t \circ < U, z_{t-1} >) \tag{21}$$

So to summarize, the first difference between LSTM and GRU consist in the number of gates; then anyways GRU controls the flow of information like the LSTM unit, but without having to use a cell memory. While it can be noticed that GRU's train faster and perform better than LSTM's on less training data; this fact is directly related to the minor complexity of GRU's. The computational complexity may then be the main advantages of GRU's over LSTM,

since most of the time (not considering short sequences data) they produce both excellent results [23].

# 3 Numerical applications

In this chapter I will show and explain the empirical experiments I conducted in order to assess the advantages of Recurrent Neural Networks for the modelling stocastich process $\kappa_t$. Till now in the field of AI mortality modelling few solutions have been proposed. I remind in the first place the research of Deprez *et al.* [7], a comparison between benchmark models and the usage of Machine Learning techniques. Then I remind here Hainaut *et al.* [9] where a NN is proposed for mortality rates forecasting: this procedure is capable of consider and detect non-linearities presents in the evolution of *log*-forces of mortality. More over, a very important research about a deep learning integrated Lee-Carter model was proposed by Nigri *et al.* [17]; here $\kappa_t$, the stochastic process of the Lee-Carter, is modelled with an LSTM architecture. The results in this research are very hopeful for the introduction of AI in the forecasting phase of the stochastic parameter. Further research proposed by Bravo [3] in the field of RNN approach to mortality forecasting, shows how GRU and LSTM architectures increase the accuracy of predictions. While a recently proposed paper in the context of mortality rates was made by Richman and Wuthrich [19], where RNN's algorithm are applied to multiple-populations in order to better estimates Lee-Carter parameters.

The most suitable RNN for this purpose are LSTM and GRU, both very performing in time-series forecasting. While the benchmark to compare will be the classic ARIMA model. Therefore my purpose will be to first fit the Lee-Carter model and obtain the parameters, $\alpha_x, \beta_x, \kappa_t$; in order to produce a forecast comparison to the ARIMA, fit an RNN to the stochastic process $\kappa_t$ and compare the results obtained with the ARIMA method. In order to compare the different approaches, I will use the test-set to evaluate some metrics related to the observed values of the time series and the forecasted ones. Once I compared and found the best fitting, the real forecasting will begin, with a predicted feature time space of 10 years. So, once I have evaluate the Lee-Carter fit, the idea is to extract the fitted time series $\kappa_t$ that will become the new data frame to work on. Divided in train and test set, after a pre-processing data phase, is possible to perform one of the selected RNN technique. Once the AI model is fitted, is possible to predict both training and testing set. The same is done for the benchmark method; after selecting and fitting the best ARIMA model, will be possible to assess the difference between the two approaches. Is important to notice that the forecasting abilities of the 2 approaches (ARIMA and RNN) have to be proven on the testing set, since for the future forecasting there will be no possible actual value to compare with. In this phase I will produce several experiments on the RNN; this is due to the fact that there are a lot of different parametrization that can lead to better score in the fitting. After these experiments on the RNN, will be possible to assess the difference between the best ARIMA fitting and the best RNN for each country and gender. The difference will be valued on the "loss" score, to have an initial idea of the

magnitude of the accuracy. Once I have fixed the two approaches, it will be possible to forecast the real mortality rates in the context of the Lee-Carter model. And after some initial consideration, I will switch to longevity swap pricing.
For what concerns the ARIMA(p,d,q) calibration, according to to the Hyndman-Khandakar algorithm [12], the procedure to find the best set of parameters for the forecasting involves at first an assessment on the stationarity of the input time series, calibrated on the training set of $\kappa_t$; an appropriate unit root test to choose the differencing order $d$. Then, based on a specific information criteria( AIC or BIC information) the algorithm selects the best values of $p$ and $q$, respectively the auto-aggressive and the moving average orders.

## 3.1 Recurrent Neural Networks approach framework

For what concern the RNN, in order to perform a deep learning fit, the first step is always data pre-processing. Hence the input data, in order to produce a well functioning AI algorithm, needs to be scaled; this is related to the fact that when input data are not scaled, during the supervised training is possible to have tail values that can give to the RNN a misleading weight hint for some instance. Therefore mean and standard deviation of the input data set can be used as the scaling coefficients to scale both the training and testing data sets as well as the predicted values. This way we ensure that the scaling does not impact the model. So after scaling, the general procedure for time series forecasting involves a lagged split of the input data set: the algorithm will take as an input $i \in \mathbb{N}$ subsequent values to predict one. In this way the RNN will predict the value of $\kappa_t$ through a function $\phi$ linking the predicted value to its time lags:

$$\kappa_t = \phi(\kappa_{t-1}, \kappa_{t-2}, ..., \kappa_{t-i}) + \epsilon_t \tag{22}$$

where $\epsilon_t$ is the homoschedastic error term. Hence once the $k_t$ time series its modified with $i$ lags, the algorithm requires to split the time series into training and testing set. The training set, where $n$ is the number of years needed to train the model and time lag $i = 2$, will look as follows:

Table 1: Data set for training

| **INPUT** | | **OUTPUT** |
|---|---|---|
| $\kappa_t$ | $\kappa_{t+1}$ | $\kappa_{t+2}$ |
| $\kappa_{t+2}$ | $\kappa_{t+3}$ | $\kappa_{t+4}$ |
| ... | ... | ... |
| $\kappa_{t+n-2}$ | $\kappa_{t+n-1}$ | $\kappa_{t+n}$ |

So once the model is trained with the training set, and validated with the testing set, the RNN has learned the main features related to the input-output relationship. Then is now possible to perform the forecast of the time series. In order to produce values it will just be necessary to give as input the matrix of

dimension $[n \times i]$ to obtain the output vector $[n \times 1]$. While the values after the last year of observation, so at time $(n+1, n+2...)$, will be obtained recursively. It should be noted that for time $t = n+1, n+2, ...$ the $\widehat{\kappa}_t$ values are forecasted from those predicted previously as there is no data available for periods after $n$. For this reason it will be possible to compare the ARIMA and RNN models just on the testing set, since there are no available data for the comparison after the last year of $\kappa_t$ time series.

## 3.2 Empirical forecasting experiments

All the experiments and results were conducted with Rstudio 4.1.2 (R) while the data were taken from Human Mortality Database [12] HMD. For the Lee-Carter fit I used the package StMoMo [22], for the RNN fit the usual Keras package. While for the forecasting as well for the ARIMA fit Forecast package was used. In addiction package Demography [11] was needed to obtain the data. For the purpose of this research six countries were considered: USA, Italy, Japan, Belgium, United Kingdom, Portugal. While given the purpose of the research, I focused on a subset 3.2 of mortality data from 1950 to the last available year in the Human Mortality Database.

Table 2: Selected countries, LC fitted years and testing years.

| **Country** | **Fitted Years** | **Testing Set Years** |
|---|---|---|
| USA | 1950-2019 | 2006-2019 |
| Italy | 1950-2018 | 2005-2018 |
| Japan | 1950-2020 | 2007-2020 |
| Belgium | 1950-2020 | 2007-2020 |
| UK | 1950-2018 | 2005-2018 |
| Portugal | 1950-2020 | 2007-2020 |

For what concerns the analyzed age of the several countries, I take a range from 50-90 years old, since for the purpose of Longevity Swap pricing the reference population age is in this interval.

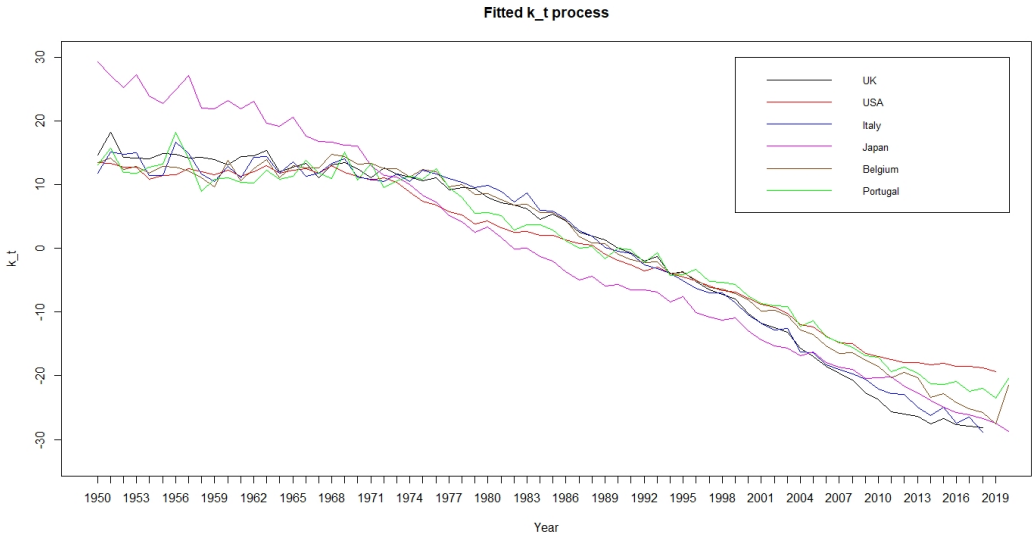


Figure 4: A representation of the fitted $\kappa_t$ process for all the countries.

In order to produce a different estimation for mortality rates, a first round of parameters fine tuning was carried out for all the countries distinguishing by gender. Once the assessment on the best parametrization for each country and gender was made, it was possible to compare the results with the best ARIMA fitting. The comparison is based on the RMSE between the actual and forecasted test set. The measure for the comparison, the Root Mean Squared Error is implemented as follows:

$$\mathbf{RMSE} = \sqrt{\frac{\sum_{t=\tau}^{T}(\kappa_\tau - \widehat{\kappa_\tau})^2}{(T-\tau+1)}} \tag{23}$$

where $\widehat{\kappa_t}$ is the forecasted process, $(\tau, T)$ are respectively the first and last year forecasted. For the purpose of these experiments, I decided to just adopt this loss measure to compare the performance of Recurrent Neural Networks over ARIMA, in the forecasting context. All the experiments were produced with differentiation by country and gender. I report in tables some of the main AI parametrization, showing the RNN techniques and their RMSE on the observed test set. Has to be noticed that the all the reported models outperform in term of RMSE the best fitted ARIMA for each country and gender. In the tables [3, 6, 12] I summarize the main RNN architectures for each country. The results obtained are very optimistic regard the usage of RNN's over ARIMA in the forecasting procedures. As matter of fact all the reported experiments of the several different parametrization and models outperform in term of RMSE the ARIMA forecasted ones. For example, in figure 3.2 is possible to see how the RNN forecasting better estimate the real stochastic process $\kappa_t$. In this case the

RNN estimation was produced by a GRU with one layer and 8 hidden units; the comparison (blue) was instead calibrated with ARIMA(0,2,2).

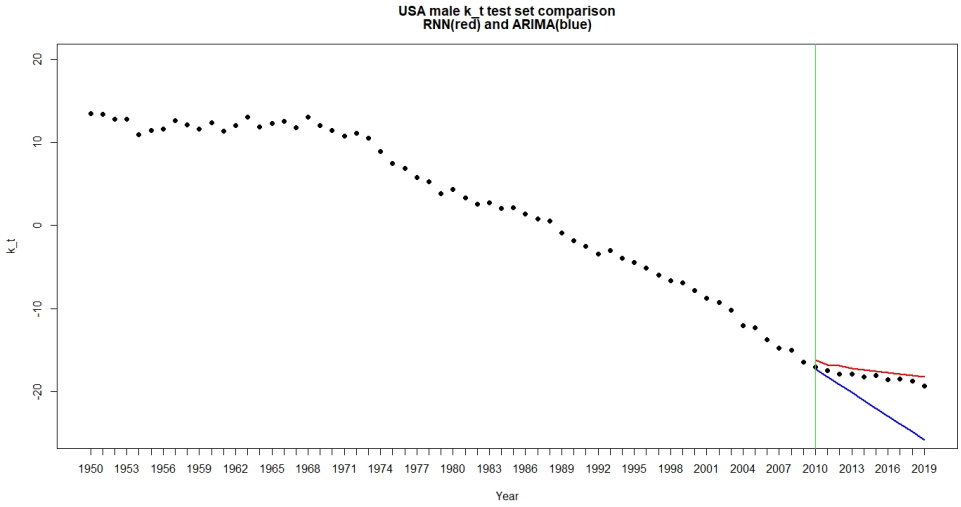


Figure 5: Forecasting $\kappa_t$ for USA male on the test set. RNN (red) and ARIMA (blue).

Is possible to see how the GRU architecture manages to catch both a short term both a long term pattern that leads to a better fitting to the observed data. Here in table 3 is possible to see the several Deep Learning forecast model for the $\kappa_t$ process; the obtained RMSE on the test set really shows how this techniques can improve the accuracy of the forecasted time series. The rest of the parametrization I found for the other four countries are showed in the appendix [6, 12]. I pick the best model (between RNN's) for each country and gender on the results obtained in the forecasted test set: the lowest RMSE will be the candidate for the longevity swap price procedures. In general both models produce a very accurate forecast, but for some cases GRU slightly exceeds the accuracy of LSTM. This advantage may be due to the fact that GRU's architecture requires less data for the training and in general they have less computational complexity as explained before. At most, for Portuguese female population in table 3 I was not able to find any LSTM parametrization that would exceed the accuracy of ARIMA forecast. For the purpose of estimating the stochastic process $\kappa_t$, the rectified linear unit (relu) has in most of the case overcome the performance of the Hyperbolic tangent (tanh). While for what regards the number of units and layers for each model, I discovered that for this time series, an excessive number of parameters do not lead to a better accuracy. Regarding epochs and batch size, it is difficult to not fall into over fitting situation. For this reason I

Table 3: Portugal and Belgium Recurrent Neural Network parametrization.

| **Portugal** | **Model** | **Layer 1** | **Layer 2** | **Batch size** | **Epochs** | **Activation** | **RMSE** |
|---|---|---|---|---|---|---|---|
| Female | GRU_1 | 8 | 0 | 6 | 50 | relu | 1.313626 |
| Female | GRU_1 | 10 | 0 | 8 | 60 | tanh | 1.463815 |
| Female | GRU_1 | 8 | 0 | 8 | 80 | relu | 1.594573 |
| Female | GRU_2 | 4 | 2 | 12 | 150 | tanh | 1.603688 |
| Female | GRU_1 | 10 | 0 | 8 | 55 | relu | 1.170603 |
| Female | GRU_2 | 8 | 6 | 8 | 45 | tanh | 1.151239 |
| Male | GRU_1 | 10 | 0 | 10 | 130 | tanh | 1.588663 |
| Male | GRU_2 | 6 | 4 | 8 | 120 | relu | 1.593634 |
| Male | LSTM_2 | 7 | 6 | 8 | 80 | relu | 1.47787 |
| Male | GRU_2 | 12 | 6 | 8 | 75 | tanh | 1.161382 |
| Male | LSTM_1 | 12 | 0 | 10 | 200 | tanh | 1.617838 |
| Male | GRU_1 | 8 | 0 | 8 | 80 | tanh | 1.180199 |
| **Belgium** | **Model** | **Layer 1** | **Layer 2** | **Batch size** | **Epochs** | **Activation** | **RMSE** |
| Female | GRU_2 | 12 | 6 | 10 | 105 | relu | 0.964462 |
| Female | LSTM_2 | 8 | 6 | 10 | 100 | tanh | 0.868235 |
| Female | GRU_1 | 5 | 0 | 8 | 25 | tanh | 1.312584 |
| Female | LSTM_1 | 10 | 0 | 10 | 60 | relu | 1.26508 |
| Female | LSTM_2 | 10 | 5 | 10 | 120 | relu | 1.187499 |
| Female | LSTM_2 | 10 | 4 | 12 | 40 | relu | 0.960467 |
| Male | GRU_2 | 10 | 4 | 12 | 80 | relu | 0.874757 |
| Male | LSTM_1 | 8 | 0 | 12 | 120 | relu | 1.223899 |
| Male | LSTM_1 | 15 | 0 | 10 | 80 | relu | 1.34986 |
| Male | LSTM_1 | 12 | 0 | 10 | 70 | relu | 1.467062 |
| Male | GRU_2 | 10 | 8 | 8 | 110 | relu | 1.411856 |
| Male | GRU_2 | 12 | 6 | 10 | 75 | relu | 1.001406 |

Table 5: Parametrization for the model I use in the pricing section, with comparison to ARIMA RMSE on the test set.

| Country | Gender | Model | Layer 1 | Layer 2 | Batch size | Epochs | Activation | RMSE RNN | RMSE ARIMA |
|---|---|---|---|---|---|---|---|---|---|
| **USA** | Male | GRU_1 | 8 | 0 | 6 | 145 | relu | 0.3570894 | 3.671365 |
| | Female | LSTM_1 | 10 | 0 | 6 | 220 | relu | 0.4389882 | 1.756939 |
| **Italy** | Male | LSTM_2 | 10 | 5 | 6 | 70 | relu | 0.8941078 | 2.066384 |
| | Female | LSTM_2 | 10 | 6 | 6 | 40 | relu | 0.9486407 | 1.8091 |
| **Portugal** | Male | GRU_2 | 12 | 6 | 8 | 75 | tanh | 1.161382 | 1.699426 |
| | Female | GRU_2 | 8 | 6 | 8 | 45 | tanh | 1.151239 | 1.634175 |
| **Belgium** | Male | LSTM_1 | 12 | 0 | 10 | 70 | relu | 1.467062 | 2.257479 |
| | Female | LSTM_1 | 10 | 0 | 10 | 60 | relu | 1.26508 | 2.187526 |
| **UK** | Male | GRU_1 | 8 | 0 | 8 | 100 | relu | 1.143459 | 3.472622 |
| | Female | GRU_1 | 4 | 0 | 10 | 190 | tanh | 1.181998 | 2.642061 |
| **Japan** | Male | GRU_1 | 3 | 0 | 10 | 200 | relu | 0.840978 | 0.9126 |
| | Female | GRU_1 | 4 | 0 | 10 | 190 | tanh | 1.181998 | 2.642061 |

decided to stop all the training phase of the algorithm whenever the validation loss was not decreasing within 10 epochs: over fitting, although is not the main difficulty with time series data, can easily affect the forecasted values.

Table 4: Test set forecasted RMSE, RNN vs ARIMA.

| Country | Gender | RNN | ARIMA |
|---|---|---|---|
| **USA** | male | 0.3571 | 3.6714 |
| | female | 0.4390 | 0.6107 |
| **Italy** | male | 0.8941 | 2.0664 |
| | female | 0.9486 | 1.8091 |
| **Portugal** | male | 1.1614 | 1.6994 |
| | female | 1.1512 | 1.6342 |
| **UK** | male | 1.1435 | 3.4726 |
| | female | 1.1820 | 2.6421 |
| **Japan** | male | 0.8410 | 0.9126 |
| | female | 0.8714 | 2.4617 |
| **Belgium** | male | 0.8748 | 2.2575 |
| | female | 0.8682 | 2.1875 |

Hence in the context of mortality rate forecasting, the experiments I perform gives empirical evidence of a substantial advantage of RNN over the canonical ARIMA approach. As is possible to see in table 4, the RNN's techniques always exceed in accuracy the ARIMA forecasting in the test set. The category "RNN" refers to the best trained approach (RMSE-wise) as in table 5, while "ARIMA" refers to the best combinations (p,d,q) that are showed in 10.

Although the usage of AI shows an important development in the forecasting of mortality rates, I notice several problems in the fine-tuning phase. I think that the main problem focuses on the fact that these algorithms even with the same premises (parametrization) do not always return the same outputs. While an ARIMA process depends solely on the mathematical background ( fitted model) and the time series itself. For these reasons I think the usage of

Deep Learning techniques is not entirely obvious, although it markedly improves future estimates of mortality rates. The only country that perform well under ARIMA assumption is Japan; this fact may be related to the strong increasing longevity trend that the $\kappa_t$ process shows 4. In the next chapter I first introduce Longevity swaps contract to then explain the main pricing techniques; after the explanation of the chosen pricing methodology I show the results of the experiments conducted within the usage of RNN for the forecasting of mortality rates.

# 4 Longevity Swap Pricing

The constant scientific and biological improvements in the past century leaded the humans to an ongoing improvements of the life expectancy at all ages. Longevity improvements have been constant over the past decades, and will probably be for the next ones. For this reason Longevity risk has become an important topic for insurance companies and pension funds. Longevity risk is the potential risk attached to the increasing life expectancy of policyholders, which can result in higher than expected payouts for insurance companies. To hedge this risk, there are two broad category of solutions:

1. Insurance-based solutions.
2. Capital markets-based solutions.

For the purpose of this research, I focus on the pricing of some of the Capital Markets solutions. Among the most used ones, I analyse Longevity Swap differentiating the pricing calculations among the techniques object of research. The hedging for these category of instruments is represented by the usage of Derivatives contracts, which are those whose value is dependent on an underlying asset, group of assets, or benchmark. A Longevity Swap is an agreement between two parties to exchange periodically at future dates $t = 1, 2, ..., T$ a series of fixed payments for a series of random longevity-dependent payments. The latter are based on realized survival rates of an index or reference population (index swap) or may be based on actual pension's plan experience (indemnity swap). Focusing on the index swap, the two counterparties will periodically exchange payments based on the notional amount $N$ agreed at inception and based on the survival rates of a reference population or index. Then a Longevity Swap is a capital market instrument for transferring to another counterpart longevity or mortality risk. So basically the fixed leg (i.e. pension fund)enter this type of contract with the aim to hedge its business in force against longevity risk.

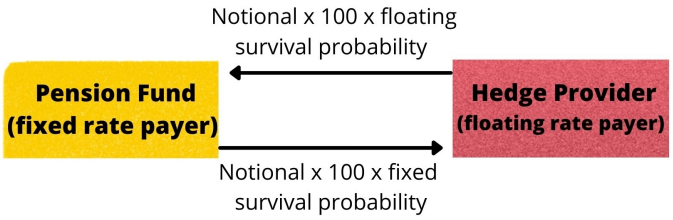


Figure 6: Plain vanila longevity swap scheme.

Net Payments Amount for the fixed leg at each epoch $t$, are given by:

$$NPA_t = Notional \times 100 \times [{}_tp_x^{fixed} - {}_tp_x^{obs.}] \tag{24}$$

where ${}_tp_x^{fixed}$ refers to the pre-agreed survival rate for the reference population aged $x$ at time $t$ and ${}_tp_x^{obs.}$ represent the observed at that time. The Net Payoff

Amount (NPA) can assume both positive and negative values; in order to hedge longevity risk, for example a pension fund should enter a Longevity Swap as a fixed-rate payer so that it would receive a positive net payment from the counterpart whenever observed mortality rates turns out to be lighter than expected (longevity risk).

To have a glimpse of the absorbing market capacity for Longevity Swap contracts, in the table below is possible to see the main deals for Longevity swap and longevity Reinsurance 6(extracted from and available at the link)

Table 6: Main Longevity Swap and Reinsurance deals since 2018.

| Buyer | Provider(s) | Solution | Size | Date |
|---|---|---|---|---|
| Phoenix Group | Metlife | Longevity reinsurance | $2.4 billion | Dec 2021 |
| Unnamed UK pension | Zurich / Metlife | Longevity swap and rein. | $3.5 billion | Dec 2021 |
| Aegon | Reinsurance Group | Longevity reinsurance | EUR 7 billion | Dec 2021 |
| Athora Netherlands | Reinsurance Group | Longevity reinsurance | EUR 3.3 billion | Sep 2021 |
| ICL Group Pension | Swiss Re | Longevity swap and rein. | £3.7 billion | May 2021 |
| Unknown UK pension | Inc. and Zurich | Longevity swap and rein. | £6 billion | Mar 2021 |
| Athora Netherlands | Canada Life Re | Longevity reinsurance | EUR 4.7 billion | Mar 2021 |
| AXA UK Pension Scheme | Hannover Re | Longevity swap | £3 billion | Mar 2021 |
| Legal General | MetLife, Inc. | Longevity reinsurance | $2 billion | Dec 2020 |
| BBC Pension Scheme | Zurich / Canada Life Re | Longevity swap reinsurance | £3 billion | Dec 2020 |
| Barclays Bank UK Ret. Fund | Reinsurance Group of America | Longevity swap | £5 billion | Dec 2020 |
| Prudential Staff Pension Scheme | Pacific Life Re | Longevity swap | £3.7 billion | Nov 2020 |
| Rothesay Life | MetLife, Inc. | Longevity reinsurance | $320m | Oct 2020 |
| UBS (UK) Pension | Zurich / Canada Life Re | Longevity swap reinsurance | £1.4 billion | Jul 2020 |
| Willis Pension Scheme | Munich Re | Longevity swap | £1 billion | Jun 2020 |
| Pension Insurance Corporation | Metlife, Inc. | Longevity reinsurance | £280m | Jun 2020 |
| NN Group | Canada Life, Munich Re, Swiss Re | Longevity reinsurance | EUR 13.5 billion | May 2020 |
| Lloyd's Banking Group | Pacific Life Re / Scottish Widows | Longevity swap reinsurance | £10 billion | Jan 2020 |
| Aegon | Canada Life Reinsurance | Longevity reinsurance | €12 billion | Dec 2019 |
| Unknown UK FTSE 100 company | Zurich, Hannover Re | Longevity swap reinsurance | £800m | Dec 2019 |
| HSBC UK Pension Scheme | Insurance Company of America | Longevity swap reinsurance | £7 billion | Aug 2019 |
| Phoenix Group | Insurance Company of America | Longevity reinsurance | ? | Aug 2019 |
| Manulife | PartnerRe | Longevity reinsurance | ? | May 2019 |
| Manulife | PartnerRe | Longevity reinsurance | ? | Mar 2019 |
| VIVAT | Canada Life Reinsurance | Longevity reinsurance | €5.5 billion | Mar 2019 |
| Manulife | RGA Life Reinsurance | Longevity reinsurance | ? | Feb 2019 |
| Pension Insurance Corporation | SCOR | Longevity reinsurance | £1.2 billion | Dec 2018 |
| Lafarge UK Pension Plan | Munich Re | Longevity swap | ? | Aug 2018 |
| Unnamed UK pension | Legal General | Longevity swap reinsurance | £300 million | Aug 2018 |
| Aviva | Insurance Company of America | Longevity reinsurance | $1.4 billion | Aug 2018 |
| National Grid | Zurich | Longevity swap | £2 billion | May 2018 |
| Pension Insurance Corp. | Insurance Company of America | Longevity reinsurance | $1.2 billion | May 2018 |
| Scottish Widows | Prudential | Longevity reinsurance | $1.8 billion | Feb 2018 |

As is possible to see, these type of hedging contract are nowadays widely used in order to prevent unexpected payout due to longevity improvements in mortality. Since Longevity improvements in the mortality are constantly increasing over time, and also the market is composed mainly of hedge providers (short position) more than "long" investors, the entities willing to hedge longevity risk, require as a compensation for it. Mainly for these reasons a premium to enter the contract is asked to the fixed leg (i.e. pension fund). Hence the premium calculation assume a primary importance for the Longevity risk market; in the next section I will introduce the pricing method I used to perform the several experiments about the Longevity Swaps premium.

## 4.1 Pricing methodology

The premium $\pi$ is proportional to the value of the possible future agreed payments arising from the fixed leg. Hence in order to price such instruments we need to apply a Risk-Neutral valuation: the contract settlement at each epoch $t$ should be fair for both parties at the signing of the contract in $t = 0$. Since in the case of Longevity Swap the underlying mortality rate is not tradable, the common non-arbitrage valuation methodology, which is based on the idea of replicating risk, can not be applied for this broad category of contracts.

Considering the fixed leg, its periodically fixed payments $K_t$ are determined by:

$$K_t \equiv \prod_{i=0}^{t-1}(1-(1-\pi)_i q_{x+i}^{BE}) \times N = {}_tp_x^{\pi} \times N, \quad t \geq 1 \tag{25}$$

where the product represent the formula to extract ${}_tp_x$ (probability of head aged $x$ to survive at least $t$ years) using the corresponding death probabilities. Is possible to see how the premium for the fixed leg $\pi$ represent the reduction of the best estimate mortality rates that the fixed leg pays for enter the contract; the fixed leg rate ${}_tp_x^{\pi}$ are higher than the best estimates ones ${}_tp_x^{BE}$. Hence ${}_tp_x^{\pi}$ are the premium adjusted best estimate survival probabilities, where:

1. with $\pi > 0$ refers to fixed leg hedged from Longevity risk
2. with $\pi < 0$ correspond to an hedging against mortality risk

While with regards to the floating leg, its payments will be equal to:

$$S_t = {}_tp_x^{obs} \times N, t \geq 1 \tag{26}$$

Then the only problem is to estimate the value of the risk premium $\pi$ applied to the fixed leg. There are several techniques to estimate this parameter, among the most used ones there are:

1. Wang transform method
2. Sharpe ratio method
3. Utility function pricing approach
4. Forward Force of Mortality
5. Consumption CAPM model

For the purpose of this work I focus just on the Wang transform. The usage of the Wang Transform refers to the necessity of convert the expected fixed payments into their Risk Neutral version. To determine the Risk neutral adjusted rate ${}_tp_x^{\pi}$ we need to solve the following:

$$g_\lambda(p) = \Phi(\Phi^{-1}(p) + \lambda) \tag{27}$$

where $\Phi()$ is the cumulative standard normal distribution, $p$ represent the best estimate of fixed survival rate ${}_tp_x^{BE}$ and $\lambda$ is the market price of risk. Hence the risk premium adjusted probabilities will be:

$$ {}_tp_x^{\pi} = g_{\lambda}(p) \tag{28} $$

So the Wang transforms adds automatically the longevity risk to the survival probabilities, transforming those into risk neutral measures so that no additional premium is required to the fixed leg. The fixed payments are then determined by discounting the multiplication of the risk neutral rates with the notional amount; the discount is made through the market interest rate term structure. While the floating payments are calculated by simply discounting the best estimates rates times the notional. The only issue arising from the Wang transform method is the determination of the market price of risk $\lambda$, which can be detected from the market annuity price [14]. Hence once the adjusted survival probabilities are estimated, in order to price the derivative contract, one just need to discount every cash flows to $t = 0$. So considering $B(0,t)$ the discount factor for the considered period, the following will give the fair value of the Swap in time $t = 0$:

$$ \text{Premium} = \sum_{t=0}^{T} N \times B(0,t) \times ({}_tp_x^{adj} - {}_t\, p_x^{BE}) \tag{29} $$

where $N$ is the upon agreed notional, ${}_tp_x^{adj}$ are the Wang-adjusted survival probabilities for population aged $x$, while ${}_tp_x^{BE}$ are the Best Estimates forecasted rates.

## 4.2 Numerical experiments

In order to price Longevity Swaps, first we have to forecast the matrix of mortality rates. As explained, the benchmark model in this area is the Lee-Carter model with the best fitted ARIMA modelling for the forecasting procedure; in comparison I developed a Lee-Carter integrated model with an Artificial Intelligence estimation for the $\kappa_t$ stochastic process. Hence the difference in the rates is accrued using two different $\kappa_t$ forecasted time series, while the $\alpha_x$ and $\beta_x$ are the classic fitted Lee-Carter parameters. The Lee-Carter can produce the forward mortality rates simply forecasting the time dependent parameter $\kappa_t$: so basically between the two different approaches (RNN, ARIMA), only the time series parameter will differ, leading to different mortality rates. The LC will first produce the *log* of central mortality rate $m_{x,t}$; then to obtain the death probabilities $q_{x,t}$ will be obtained as :

$$ q_{x,t} = 1 - e^{-m_{x,t}} \tag{30} $$

Then is possible to obtain the survival probabilities ${}_tp_x$, as $1 - q_{x,t}$. So once the two matrix of survival probabilities are computed is easy to obtain the needed rates ${}_tp_{x0}$, considering the reference age $x_0$ and a time horizon $t = 1, 2, ..., T$.

To remind, the benchmark estimation is calibrated on the best ARIMA(p,d,q) fitting for each country and gender time series; while the comparison is made on RNN's testing set lowest RMSE (explained above). Hence after the forecasting of the two different survival probabilities , in order to price and use the Longevity risk measure, the Wang transform was applied to the survival probabilities. In this case, the coefficient $\lambda$ is estimated through the annuity market price. For the purpose of this work I decided to adopt $\lambda = 0.1, 0.2, 0.3$ in order to see the difference accrued on the usage of a different market price of risk. Going further in the pricing, the procedure requires to estimate the interest rate for the contractual period established; in order to consider the actual value of payments, the cash flows must be discounted from their time period to the present. One way to evaluate the interest rate term structure is the calibration of the Nelson-Siegel-Svensson model [8]. In this case the several parameters are taken from the European Central Bank website (ECB), where the parameters are published daily. Hence, is possible to perform the longevity swap pricing; the procedure is applied both to the forecasted ARIMA both to the RNN's adjusted rates. For the evaluation of the price, Rstudio was used, implementing the Wang transform and the Nelson-Siegel term structure manually. In order to produce the prices, the formula 29 was implemented manually in R. It must be specified that the first survival probabilities used in the formula 29 are not the forecasted "spot" rates, since the collected data end at maximum in 2020, but the forward rate start from 2022 in order to produce the actual premium. This means that with better data availability, both for ARIMA and RNN's, the accuracy of the forecasted survival probabilities would have been more precise.

For the purpose of this work I decided to evaluate the price of the following contract:

1. Notional = $1,000,000$ €
2. Reference age = 55 years old population.
3. Duration = 10 years.

Then all the contracts premium where produce based on country and gender. As explained before, the market price of risk $\lambda$ used in the Wang transform in order to produce the "market" survival probabilities, can bring significant changes to the price of the contract. Although this parameter has to be estimated through market price of quoted life annuity for each country, for the purpose of this research I decided to focus just on $\lambda = 0.1, 0.2, 0.3$. In table 4.2 I report the price produced for UK and Belgian population with the underlying survival rate forecasted through deep learning integrated Lee-Carter model and with the ARIMA approach.

To be precise, the category "RNN" refers to price produced with the Deep Learning technique with the lowest RMSE on the testing set as in table 3; while "ARIMA" is the best combination (p,d,q) for each country and gender (computed with auto-arima [link]) used in the forecast procedure as in table 10. Hence the obtained price are differentiated by forecasting technique, gender,

Table 7: Longevity Swap price for UK and Belgium reference populations.

| **UK** | **RNN** | **ARIMA** | $\lambda$ |
|---|---|---|---|
| male | 57,447 € | 64,689 € | 0.1 |
| female | 47,477 € | 45,398 € | 0.1 |
| male | 104,102 € | 117,591 € | 0.2 |
| female | 85,706 € | 81,866 € | 0.2 |
| male | 141,644 € | 160,460 € | 0.3 |
| female | 116,204 € | 110,887 € | 0.3 |
| **Belgium** | **RNN** | **ARIMA** | $\lambda$ |
| male | 73,732 € | 71,099 € | 0.1 |
| female | 49,905 € | 46,094 € | 0.1 |
| male | 134,418 € | 129,480 € | 0.2 |
| female | 90,175 € | 83,123 € | 0.2 |
| male | 183,909 € | 176,978 € | 0.3 |
| female | 122,369 € | 112,593 € | 0.3 |

reference country and market price of risk $\lambda$. Here in figure 7 is possible to see how for England population (hence derived from UK mortality rates $q_{x,t}$) the two different approaches give very different results with respect to the experiments conducted on Belgian population.

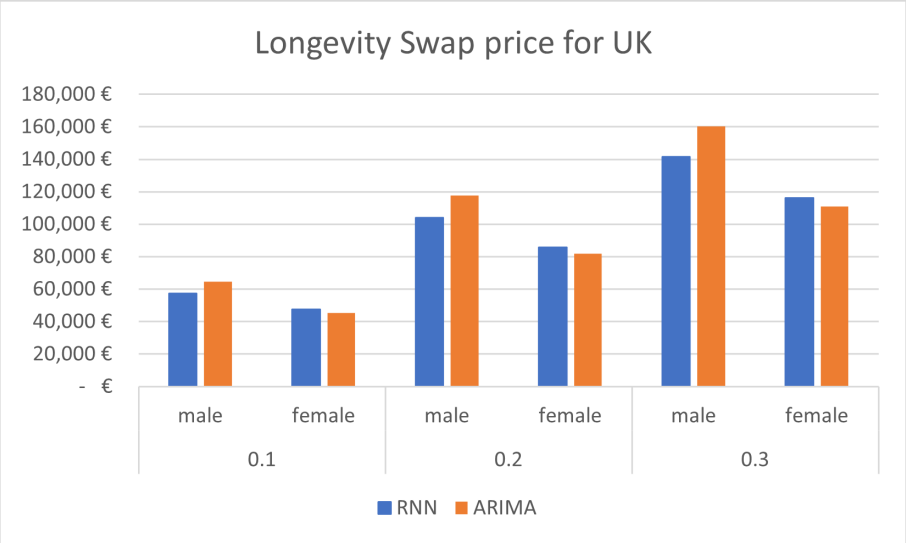


Figure 7: Barplot of Longevity swap price for UK contracts; distinguished by value of market price of risk $\lambda$ and by gender.

Meaning that the forecasted rate for UK population are very different between ARIMA and RNN forecasting. I assembled the tables for the remaining

countries in 8; the contract's details remains the same, as well as for the several price "categories". As for Portugal male reference population, also for USA and UK male populations the produced prices with RNN are lower than the ARIMA ones; while the price for female population gives the opposite hint, with higher prices using the deep learning adjusted LC model.

Table 8: Tables of Longevity swap prices for each country.

| **Japan** | **RNN** | **ARIMA** | $\lambda$ | **Italy** | **RNN** | **ARIMA** | $\lambda$ |
|---|---|---|---|---|---|---|---|
| male | 62,991 € | 55,895 € | 0.1 | male | 54,599 € | 50,042 € | 0.1 |
| female | 31,675 € | 25,519 € | 0.1 | female | 35,117 € | 32,149 € | 0.1 |
| male | 114,442 € | 101,220 € | 0.2 | male | 98,876 € | 90,433 € | 0.2 |
| female | 56,721 € | 45,485 € | 0.2 | female | 63,001 € | 57,567 € | 0.2 |
| male | 156,083 € | 137,634 € | 0.3 | male | 134,452 € | 122,733 € | 0.3 |
| female | 76,339 € | 60,959 € | 0.3 | female | 84,937 € | 77,475 € | 0.3 |
| **Portugal** | **RNN** | **ARIMA** | $\lambda$ | **USA** | **RNN** | **ARIMA** | $\lambda$ |
| male | 77,465 € | 81,317 € | 0.1 | male | 83,415 € | 88,721 € | 0.1 |
| female | 37,558 € | 34,706 € | 0.1 | female | 60,935 € | 58,346 € | 0.1 |
| male | 141,235 € | 148,505 € | 0.2 | male | 152,387 € | 162,400 € | 0.2 |
| female | 67,463 € | 62,227 € | 0.2 | female | 110,581 € | 105,763 € | 0.2 |
| male | 193,245 € | 203,508 € | 0.3 | male | 208,889 € | 223,028 € | 0.3 |
| female | 91,052 € | 83,844 € | 0.3 | female | 150,656 € | 143,941 € | 0.3 |

While for the rest of the countries there is a general trend of RNN overpricing on ARIMA methods. The cause of higher or lower price for the contracts has to be sought in the produced survival probabilities. As explained the difference and innovation consist in a different approach to forecast $\kappa_t$, and consequently a different estimation for the central mortality rate $m_{x,t}$; once the survival probabilities are obtained, the diversity in the two prices are just related to the best estimates and adjusted probabilities used in 29. Hence the premium, adopting the Wang transform, is the result of the accrued difference between the market adjusted probabilities and the forecasted ones. I noticed how the lower the survival rate ${}_tp_{55}^{BE}$ the higher the difference between the latter and the Wang adjusted one. Schematically:

$$\begin{cases} NPA_t^{\text{arima}} > NPA_t^{\text{rnn}} & if \quad {}_tp_x^{\text{arima}} < {}_tp_x^{\text{rnn}} \\ NPA_t^{\text{arima}} < NPA_t^{\text{rnn}} & if \quad {}_tp_x^{\text{arima}} > {}_tp_x^{\text{rnn}} \end{cases} \tag{31}$$

where $NPA_t$ refers the the present amount of the $t^{th}$ cash flows for the Longevity Swap. Has to be noticed that this inequalities stand just for probabilities (for the Wang transform) that are approximately higher than 0.5; this is a result of the shape of the normal distribution which is a is a descending function for probabilities greater than 0.5. Hence the discrepancies in the price and survival probabilities should be found in the LC model, and more precisely in the $\kappa_t$ process (having all the other parameters set in both techniques). The lower (higher) survival probabilities is produced from an higher (lower) value of the stochastic process at time $t$.

Is possible to see in figure 8 a significant the difference between the RNN and ARIMA premium estimation for Portugal; due to the fact that for this country there is slightly higher difference in the two estimation of future values of $\kappa_t$ and consequentially of the mortality rate

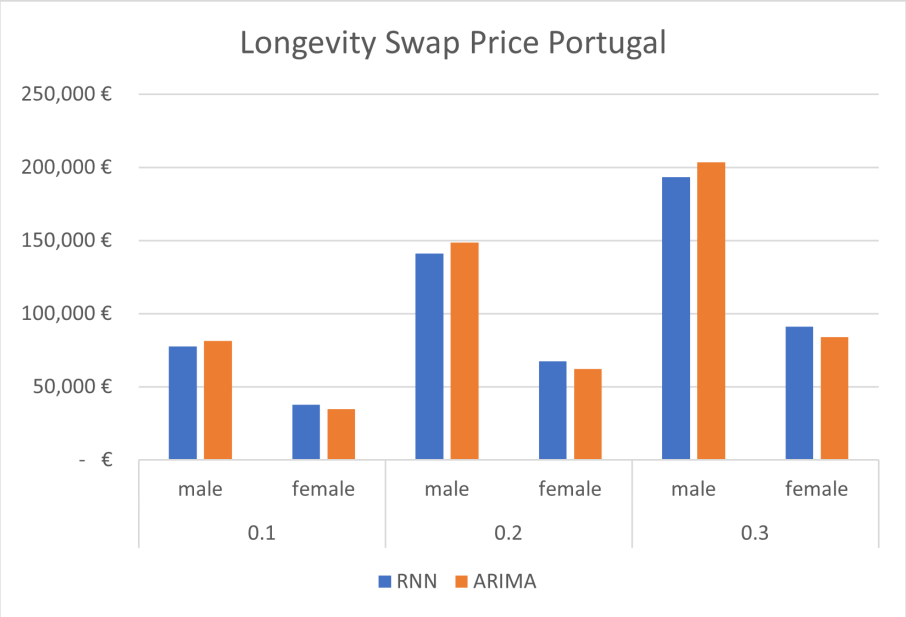


Figure 8: Barplot of Longevity swap price for Portuguese contracts; distinguished by value of market price of risk $\lambda$ and by gender.

Has to be noticed how the value of $\lambda$ influences significantly the price of the derivatives. The market price of risk is able to change the probabilities both with an upward shift (longevity risk) both with a downward (mortality risk); in table 4.2 is possible to see the accrued difference using the 3 values of $\lambda$. Is clear how the market price of risk using the Wang transform produces a significantly higher rate: the higher $\lambda$ parameter, the higher is the premium required from the market to hedge longevity risk.

Table 9: Survival probabilities ${}_tp_{55}$(used for the pricing) for US male leveled by market price of risk $\lambda$.

| Forecasted | $\lambda = 0.1$ | $\lambda = 0.2$ | $\lambda = 0.3$ |
|---|---|---|---|
| 0.993036 | 0.9947531 | 0.9960827 | 0.9971021 |
| 0.985653 | 0.9889211 | 0.9915218 | 0.9935705 |
| 0.977785 | 0.9825716 | 0.986448 | 0.9895564 |
| 0.96934 | 0.9756488 | 0.9808293 | 0.9850415 |
| 0.960475 | 0.9682896 | 0.974781 | 0.9801198 |
| 0.951045 | 0.9603762 | 0.9682068 | 0.9747125 |
| 0.941045 | 0.9519046 | 0.961101 | 0.9688116 |
| 0.930456 | 0.9428548 | 0.9534434 | 0.9623964 |
| 0.919304 | 0.9332449 | 0.9452453 | 0.9554724 |
| 0.907617 | 0.9230978 | 0.9365226 | 0.9480487 |
| 0.895467 | 0.9124721 | 0.9273226 | 0.9401622 |

In figure 9 I reported the prices solution for Longevity Swap contract for 55 years old male and with a market price of risk $\lambda = 0.2$. Is possible to see in the

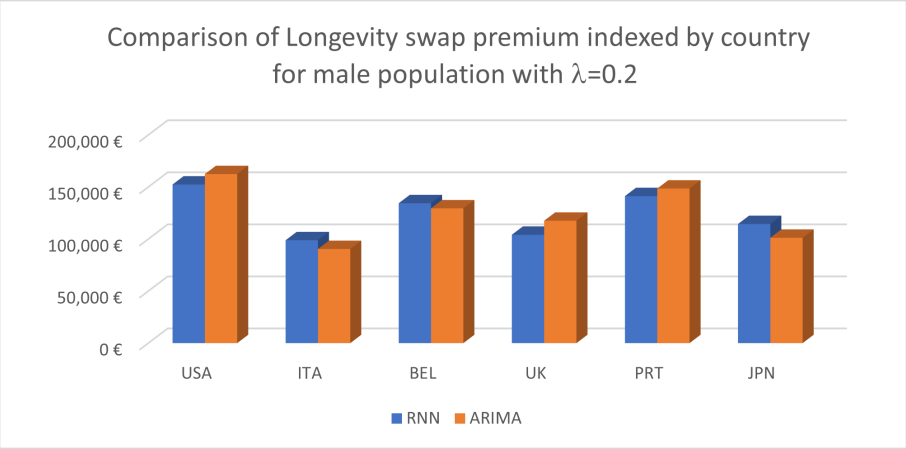


Figure 9: Price of longevity swap for 55 years old male and with $\lambda = 0.2$

barplot 9 that the RNN's produced prices are for half of the countries higher and half lower than the respective forecasted with ARIMA . The difference is accrued in the usage of the different stochastic process $\kappa_t$ in the Lee-Carter model. As explained in section 3 the comparison is made between the best fitted ARIMA (p,d,q) and the best RNN tuning for the selected country and gender; as in table 6, the only adopted Deep Learning parametrization outperform RMSE-wise the ARIMA estimation. Then is possible to think that the iterative forecasting could better perform in the future (from year 2022 on), since the result obtained on the test set (approximately 2010-2019) can be interpreted as very optimistic about the usage of AI over classical stochastic models.

Analyzing the produced price in a general way, is possible to notice that the the most evident price differences occur when ARIMA estimates for $\widehat{\kappa}_t$ differ significantly from the observed ones (test-set). As explained before all the RNN outperform in term of RMSE the ARIMA approaches on the testing set; hence is possible to think that the RNN forecast for the future years (from 2022 on) could be more accurate and realistic. Beside this, as is possible to see that an RNN approach leads to a major premium size for 3 countries and a lower price for the remaining 3. To have a general idea of the amplitude of this add on/off to the price, I calculated the percentage variation with respect to the benchmark price (ARIMA) for each country, gender and market price of risk $\lambda$. The average percentage variation of the price is then calculated in a broad way for each country, and then aggregated between the latter. The obtained coefficient (absolute mean percentage) shows that, on average, the RNN prices differs 6.7% from the benchmark estimation. This difference is more evident for some country, like USA and Japan, where the ARIMA estimation differs largely from the observed ones in the testing set. However it must be notice that I perform all the price experiments with the same contract detail, which is just a default one for the purpose of this research; the prices as well as the magnitude of the discrepancy between the two approaches(RNN vs ARIMA) can vary a lot considering different reference age and higher Notional amount for the contract.

## 4.3 Some considerations

To summarize, in the first place the experiments I conduct shows clearly how, considering the forecasting procedures of the LC, an RNN approach always leads to a better fitting than the canonical ARIMA. Precisely in most of the case, Gated Recurrent Unit outperform largely LSTM, manifesting less computational complexity and less need for data availability. As matter of fact also LSTM architecture in some country and gender produces very accurate forecast, even if compared to GRU's. Since all the RNN architecture I propose and report outperform in term of RMSE (on the test data) the ARIMA methods, I think is possible to consider the AI techniques as the most accurate in the forecasting phase. As I show, the discrepancy in the forecasted test set present a huge misleading trend by ARIMA approaches; while on the other side is a fact that some parametrization of RNN really fit the observed data. Going further in the context of Longevity Swaps, the produced prices evidences the differences between the two approaches. For some countries RNN has produced higher prices for others the opposite; the key, in my opinion, is to think that RNN's estimates (considering the assumptions in Chapter 3) produce the most accurate and fair prices in the context of longevity risk.

# 5 Conclusion

In the context of mortality, the forecasting phase is the main difficulty for the several reason I explained. Considering the applications mortality rates have in many fields, is clear how forecasting them properly is crucial for many "operators". The lack of accuracy of the classic Lee-Carter forecasting is evident comparing with deep learning integrated models. In the context of longevity improvements in the mortality field, the benchmark estimation technique of the time-dependent parameter of the canonical model need a revision due to the lack of consistency in most of the countries object of my analysis. Longevity is currently growing as one of the main risks in insurance and pension business, as well as for the underlying market for this risk. For these reasons a revision in the mortality models, carried out on the stochastic process indexed by time, should represent a critical issue.

In this research I first propose a deep learning integrated Lee-Carter model with RNN's techniques for the forecasting of the time-dependent parameter $\kappa_t$; the results are then applied to Longevity Swap pricing to see the inherent difference with the benchmark procedures. The comparison is made with ARIMA models, the classical framework for the forecast of $\kappa_t$. As alternative, I decided to focus on LSTM and GRU deep learning architectures. After a brief presentation about LC framework, I introduced the deep learning techniques object of the research. Starting from a plain vanilla Neural Network and its shortcoming in the context of time series forecasting, I then introduced LSTM and GRU as two of the most suitable RNN techniques for the topic. For the investigation I performed the numerical application on six countries world-wide and differentiated by gender. Hence I assessed the advantages of RNN's techniques over the canonical ARIMA, showing the misleading trend and the low performance in front of AI approaches. The result I obtained were then applied to index Longevity Swap pricing. Considering a default contract, I calculated the price for the derivatives contracts for the populations of the countries using both the classical forecasted mortality rate both a suitable RNN technique for the forecasting procedures. I then showed the discrepancy in the two produced price, focusing also on how I obtained different results differentiating by country and gender.

The purpose of this research is to show how the usage of Deep Learning techniques in the mortality field can lead to better underlying assumption for Derivative contract and life-business related market. The result I obtained are very optimistic surely, showing a total dominance of RNN's approach over the classical methods. But as explained in the mortality rate forecasting section, these AI approach can also produce very ambiguous result in the iterative forecasting of $\kappa_t$; consequentially all these "deviated" training and procedures, would produce very unrealistic price for the related hedging contracts.

In conclusion, I think the field of longevity risk has become of central importance to the insurance and retirement industry, to the point where the need to revisit classical modeling has become an important challenge and target. The contribution of the proposed investigation is founded primarily on the research

and the usage of new techniques in the field of mortality rate forecasting. Then since the market and research in the field of longevity risk pricing are not yet fully saturated, this article make a significant contribution for the fair value calculation for this hedging instruments.

# References


[1] GEP Box and GM Jenkins. "Control". In: *Halden-Day, San Francisco* (1970).

[2] M Martin Boyer and Lars Stentoft. "If we can simulate it, we can insure it: An application to longevity risk management". In: *Insurance: Mathematics and Economics* 52.1 (2013), pp. 35–45.

[3] Jorge M Bravo. "Forecasting Longevity for Financial Applications: A First Experiment with Deep Learning Methods". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2021, pp. 232–249.

[4] Jorge M Bravo et al. "Addressing the life expectancy gap in pension policy". In: *Insurance: Mathematics and Economics* 99 (2021), pp. 200–221.

[5] Andrew JG Cairns, David Blake, and Kevin Dowd. "Pricing death: Frameworks for the valuation and securitization of mortality risk". In: *ASTIN Bulletin: The Journal of the IAA* 36.1 (2006), pp. 79–120.

[6] Kyunghyun Cho et al. "On the properties of neural machine translation: Encoder-decoder approaches". In: *arXiv preprint arXiv:1409.1259* (2014).

[7] Philippe Deprez, Pavel V Shevchenko, and Mario V Wüthrich. "Machine learning techniques for mortality modeling". In: *European Actuarial Journal* 7.2 (2017), pp. 337–352.

[8] Manfred Gilli, Stefan Große, and Enrico Schumann. "Calibrating the nelson-siegel-svensson model". In: *Available at SSRN 1676747* (2010).

[9] Donatien Hainaut. "A neural-network analyzer for mortality forecast". In: *ASTIN Bulletin: The Journal of the IAA* 48.2 (2018), pp. 481–508.

[10] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[11] Maintainer Rob J Hyndman. "Package 'demography'". In: (2012).

[12] Rob J Hyndman and Yeasmin Khandakar. "Automatic time series forecasting: the forecast package for R". In: *Journal of statistical software* 27.1 (2008), pp. 1–22.

[13] Ronald D Lee and Lawrence R Carter. "Modeling and forecasting US mortality". In: *Journal of the American statistical association* 87.419 (1992), pp. 659–671.

[14] Yijia Lin and Samuel H Cox. "Securitization of mortality risks in life annuities". In: *Journal of risk and Insurance* 72.2 (2005), pp. 227–252.

[15] Warren S McCulloch and Walter Pitts. "A logical calculus of the ideas immanent in nervous activity". In: *The bulletin of mathematical biophysics* 5.4 (1943), pp. 115–133.

[16] Marvin Minsky and Seymour Papert. "Perceptrons." In: (1969).

[17] Andrea Nigri et al. “A deep learning integrated Lee–Carter model”. In: *Risks* 7.1 (2019), p. 33.

[18] Arthur E Renshaw and Steven Haberman. “A cohort-based extension to the Lee–Carter model for mortality reduction factors”. In: *Insurance: Mathematics and economics* 38.3 (2006), pp. 556–570.

[19] Ronald Richman and Mario V Wüthrich. “A neural network extension of the Lee–Carter model to multiple populations”. In: *Annals of Actuarial Science* 15.2 (2021), pp. 346–366.

[20] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. “Learning representations by back-propagating errors”. In: *nature* 323.6088 (1986), pp. 533–536.

[21] Richard M Van Slyke and Roger Wets. “L-shaped linear programs with applications to optimal control and stochastic programming”. In: *SIAM journal on applied mathematics* 17.4 (1969), pp. 638–663.

[22] Andrés Villegas, Vladimir K Kaishev, and Pietro Millossovich. “StMoMo: An R package for stochastic mortality modelling”. In: *7th Australasian Actuarial Education and Research Symposium*. 2015.

[23] Shudong Yang, Xueying Yu, and Ying Zhou. “LSTM and GRU neural network performance comparison study: Taking Yelp review dataset as an example”. In: *2020 International workshop on electronic communication and artificial intelligence (IWECAI)*. IEEE. 2020, pp. 98–101.

# 6 Appendix

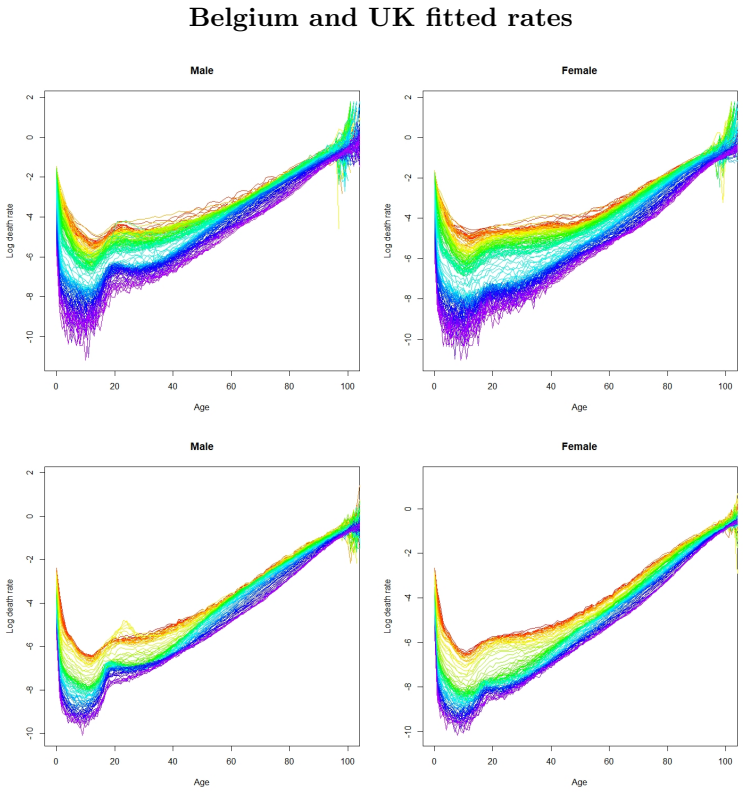


Figure 10: Belgium(above) and UK(below) fitted mortality rates.

**Italy and USA fitted rates**

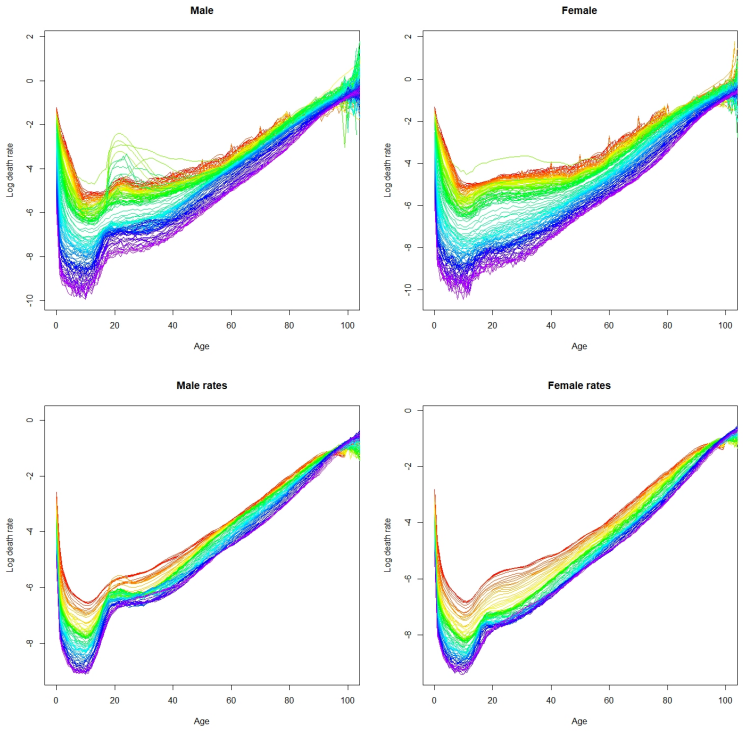


Figure 11: Fitted mortality rates for Italy(above) and USA (below).

**Portugal and japan fitted rates**

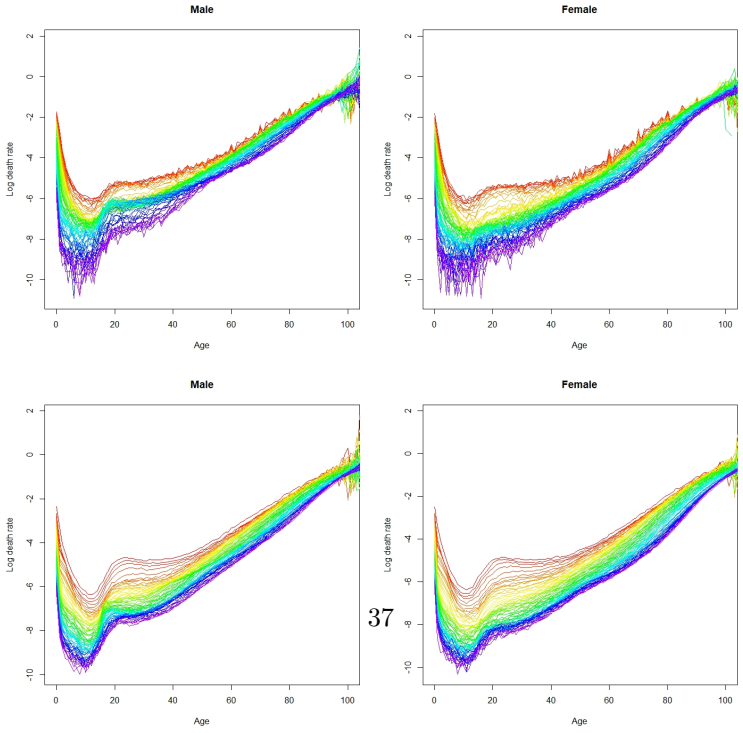




Figure 12: Fitted mortality rates for Portugal(above) and japan (below).

Table 10: Best ARIMA(p,d,q) for each country.

| Country | Best ARIMA | Country | Best ARIMA |
|---|---|---|---|
| **USA** | | **UK** | |
| male | ARIMA(0,2,2) | male | ARIMA(0,2,2) |
| female | ARIMA(0,1,0) | female | ARIMA(0,1,1) |
| **Italy** | | **Japan** | |
| male | ARIMA(0,2,2) | male | ARIMA(0,1,3) |
| female | ARIMA(0,1,1) | female | ARIMA(0,1,1) |
| **Portugal** | | **Belgium** | |
| male | ARIMA(0,1,1) | male | ARIMA(1,1,0) |
| female | ARIMA(0,1,1) | female | ARIMA(2,1,1) |

Table 11: RNN parametrization for USA and Italy by gender-

| USA | Model | Layer 1 | Layer 2 | Batch size | Epochs | Activation | RMSE |
|---|---|---|---|---|---|---|---|
| Male | GRU_1 | 5 | 0 | 6 | 60 | relu | 1.852517 |
| Male | LSTM_1 | 5 | 0 | 6 | 84 | relu | 0.542309 |
| Male | LSTM_2 | 10 | 5 | 6 | 45 | relu | 2.031578 |
| Male | GRU_1 | 8 | 0 | 6 | 147 | relu | **0.357089** |
| Male | GRU_1 | 8 | 0 | 8 | 72 | relu | 2.524158 |
| Male | GRU_1 | 8 | 0 | 8 | 174 | relu | 0.384254 |
| Female | GRU_1 | 8 | 0 | 8 | 100 | relu | 1.708639 |
| Female | GRU_2 | 10 | 5 | 8 | 100 | relu | 1.176006 |
| Female | GRU_1 | 8 | 0 | 8 | 260 | relu | 0.686256 |
| Female | GRU_1 | 5 | 0 | 6 | 210 | relu | 1.304324 |
| Female | LSTM_1 | 10 | 0 | 6 | 220 | relu | **0.438988** |
| Female | LSTM_2 | 3 | 3 | 6 | 260 | relu | 0.643964 |
| **Italy** | **Model** | **Layer 1** | **Layer 2** | **Batch size** | **Epochs** | **Activation** | **RMSE** |
| Male | LSTM_1 | 5 | 0 | 6 | 70 | relu | 2.223601 |
| Male | LSTM_1 | 10 | 0 | 6 | 60 | relu | 1.809554 |
| Male | LSTM_2 | 10 | 5 | 10 | 95 | relu | 1.015774 |
| Male | LSTM_2 | 10 | 5 | 6 | 70 | relu | **0.894108** |
| Male | GRU_2 | 8 | 5 | 6 | 180 | relu | 1.286334 |
| Male | GRU_2 | 5 | 5 | 6 | 50 | relu | 0.927548 |
| Female | LSTM_1 | 10 | 0 | 6 | 150 | relu | 2.1905 |
| Female | GRU_1 | 10 | 0 | 10 | 80 | relu | 2.295094 |
| Female | GRU_1 | 6 | 0 | 4 | 35 | relu | 2.133661 |
| Female | LSTM_2 | 10 | 6 | 6 | 40 | relu | **0.948641** |
| Female | GRU_1 | 10 | 0 | 8 | 50 | relu | 1.00273 |
| Female | LSTM_2 | 8 | 6 | 10 | 75 | relu | 1.56602 |

Table 12: UK and Japan Recurrent Neural Networks parametrization.

| **UK** | **Model** | **Layer 1** | **Layer 2** | **Batch size** | **Epochs** | **Activation** | **RMSE** |
|---|---|---|---|---|---|---|---|
| Female | LSTM_1 | 10 | 0 | 10 | 225 | relu | 1.823201 |
| Female | GRU_1 | 10 | 0 | 6 | 80 | relu | 1.346289 |
| Female | GRU_2 | 8 | 4 | 10 | 250 | relu | 2.229094 |
| Female | GRU_1 | 4 | 0 | 10 | 190 | tanh | 1.181998 |
| Female | LSTM_2 | 8 | 4 | 8 | 75 | relu | 1.540118 |
| Female | LSTM_1 | 10 | 0 | 8 | 80 | relu | 1.412982 |
| Male | LSTM_1 | 10 | 0 | 10 | 170 | relu | 2.245425 |
| Male | LSTM_1 | 10 | 0 | 10 | 400 | tanh | 3.155587 |
| Male | GRU_2 | 8 | 5 | 6 | 60 | relu | 2.290737 |
| Male | GRU_2 | 8 | 5 | 10 | 200 | tanh | 2.253668 |
| Male | GRU_1 | 8 | 0 | 8 | 100 | relu | 1.143459 |
| Male | LSTM_1 | 5 | 0 | 10 | 240 | relu | 2.018462 |
| **Japan** | **Model** | **Layer 1** | **Layer 2** | **Batch size** | **Epochs** | **Activation** | **RMSE** |
| Female | LSTM_2 | 10 | 5 | 8 | 60 | relu | 2.385545 |
| Female | LSTM_2 | 10 | 5 | 10 | 95 | relu | 1.091808 |
| Female | GRU_1 | 4 | 0 | 8 | 150 | relu | 0.871357 |
| Female | LSTM_1 | 6 | 0 | 6 | 90 | relu | 1.136775 |
| Female | GRU_1 | 10 | 0 | 8 | 65 | relu | 1.268107 |
| Female | GRU_1 | 5 | 0 | 10 | 100 | relu | 1.295658 |
| Male | LSTM_1 | 8 | 0 | 10 | 105 | relu | 0.908357 |
| Male | LSTM_2 | 10 | 5 | 10 | 55 | relu | 0.90571 |
| Male | LSTM_2 | 8 | 5 | 12 | 200 | relu | 0.883853 |
| Male | GRU_1 | 3 | 0 | 10 | 200 | relu | 0.840978 |