risks

MDPI

Article

Using Neural Networks to Price and Hedge Variable Annuity Guarantees

Daniel Doyle [1] and Chris Groendyke [2,*]

[1] VA Hedging & Forecasting, Talcott Resolution, Windsor, CT 06095, USA; dpddoyle.doyle@gmail.com
[2] Department of Mathematics, Robert Morris University, Moon Township, PA 15108, USA
* Correspondence: groendyke@rmu.edu; Tel.: +1-412-397-6054

Received: 1 November 2018; Accepted: 20 December 2018; Published: 23 December 2018

Abstract: This paper explores the use of neural networks to reduce the computational cost of pricing and hedging variable annuity guarantees. Pricing these guarantees can take a considerable amount of time because of the large number of Monte Carlo simulations that are required for the fair value of these liabilities to converge. This computational requirement worsens when Greeks must be calculated to hedge the liabilities of these guarantees. A feedforward neural network is a universal function approximator that is proposed as a useful machine learning technique to interpolate between previously calculated values and avoid running a full simulation to obtain a value for the liabilities. We propose methodologies utilizing neural networks for both the tasks of pricing as well as hedging four different varieties of variable annuity guarantees. We demonstrated a significant efficiency gain using neural networks in this manner. We also experimented with different error functions in the training of the neural networks and examined the resulting changes in network performance.

Keywords: variable annuities; GMxB; hedging; neural networks

1. Introduction

Variable annuities (VAs) are equity-linked accumulation products that provide the policyholder tax-deferred growth on investments. They are often used as retirement savings vehicles and can provide protection against downside equity market risk. This protection stems from various riders that provide guaranteed minimum values of various contract elements; they are generically referred to as GMxB. A few common examples include the guaranteed minimum death benefit (GMDB), guaranteed minimum accumulation benefit (GMAB), guaranteed minimum withdrawal benefit (GMWB), guaranteed minimum maturity benefit (GMMB), and guaranteed minimum income benefit (GMIB). VAs saw dramatic increases in popularity after these guarantees were introduced in the 1990s and early 2000s in the US and many markets worldwide, including Canada, the UK, Japan, and in some European markets (Bauer et al. 2008; Ledlie et al. 2008), in part because of the protection against market downturns they provide. More recently, this popularity has fluctuated and has been trending downward: In 2008, VA sales in the US were estimated to be $156 billion, whereas by 2017, this figure had shrunk to $96 billion (LIMRA 2018). This product line nonetheless represents a very significant block of business for insurers, with industry-wide net assets in the US for VA products totaling approximately $2 trillion at the end of 2017 (Insured Retirement Institute 2018).

Two of the fundamental actuarial tasks associated with these VA products are (1) pricing the guarantee, and (2) provisioning (reserving) for the guarantee, which may involve determining how much capital to hold, or devising a method for hedging the guarantee. As VA guarantees can often be viewed as granting a financial option to the policyholder (Hardy 2003), it is perhaps not surprising that ideas from option pricing and hedging have often been utilized in the pricing and hedging of these types of guarantees. Two of the earlier examples of this include Brennan and Schwartz (1976),

who considered the pricing of a variable life insurance product with a guaranteed minimum benefit amount, and Boyle and Schwartz (1977), who considered both guaranteed minimum death benefits and guaranteed minimum maturity benefits. Boyle and Hardy (1997) compared two approaches—one based on the option pricing theory and the other based on Monte Carlo (MC) methods—to calculate the reserve for a guaranteed minimum maturity benefit. Similarly, Hardy (2000) compared three methods of reserving for VA guarantees: A value at risk (VaR) approach, dynamic hedging, and static hedging (i.e., purchasing option contracts). Other examples include Milevsky and Salisbury (2006), who concluded that GMWB products were typically underpriced, and Ko et al. (2010), who used a Black–Scholes approach to price a dynamic withdrawal benefit. Kolkiewicz and Liu (2012) developed a "semi-static" hedging strategy for hedging some types of GMWB that requires less frequent rebalancing than a fully dynamic hedging approach.

While many papers examine only relatively simple forms of GMxB VA guarantees, the current marketplace offers a wide variety of increasingly complex forms of guarantees on VA products. One popular variant is to offer some type of "reset" or "ratchet" option, which grants the policyholder the option to reset the guarantee amount to current levels at one or more points in time over the life of the contract. Some authors who have examined such guarantees include Armstrong (2001), who used MC methods to analyze strategies involving when to reset the guarantee, Windcliff et al. (2001), who numerically solved a set of linear complementarity problems to price complex VA options (so-called "shout" options), and Windcliff et al. (2002), who looked at the impact of factors such as volatility, interest rate, investor optimality, and product design on the price of the reset feature. Bauer et al. (2008) presented a comprehensive mathematical model to price many types of GMxB, while Bacinello et al. (2011) proposed a unifying framework for valuing VA guarantees using MC methods. For a detailed review of the literature on the pricing and reserving of VA guarantees, see Gan (2013).

A significant cost associated with maintaining a large portfolio of VA policies is the computation time for the valuation of the portfolio and of relevant Greeks for hedging the portfolio. As product design and model complexity increase, the need for efficient techniques to value and hedge VA guarantees is becoming increasingly important for insurance companies. Even for some of the more relatively simple types of guarantees, it is often infeasible or impossible to derive closed-form pricing or reserving equations for these guarantees, and as a result, MC methods are often employed. While very useful, these methods can also sometimes be computationally intense and costly. Hence, researchers have explored many techniques for approximating the value and Greeks of VA contracts and portfolios. For example, Feng and Volkmer (2012) used properties of geometric Brownian motion to develop analytical methods for the calculation of risk measures, such as VaR and conditional tail expectation (CTE) for VA guarantees. The same authors also later used spectral expansion methods to compute risk measures related to GMMB and GMDB types of guarantees on VA products (Feng and Volkmer 2014). Gan and Lin (2017) used a two-level approach to estimating the Greeks associated with a portfolio of VA guarantees. Under their metamodeling procedure, they proposed first pre-calculating some individual delta values at various levels chosen using Latin hypercube sampling, and then estimating the delta values of the portfolio using these pre-calculated values. In a similar vein, Gan (2018) used a metamodel approach consisting of a linear model with interactions, and using an overlapped group lasso method (Lim and Hastie 2015) to help with the selection of variable interactions for the model. For a review of the various types of metamodels that have been used in this capacity and a discussion of their advantages, see Gan (2018).

In recent years, several authors have explored the idea of utilizing neural networks and other machine learning techniques for the tasks of pricing and/or hedging VA guarantees. Machine learning refers to a system that is trained to obtain accurate outputs using sample data and associated known outputs, as opposed to being explicitly programmed to give the desired outputs (Chollet and Allaire 2018). These methods can drastically improve the efficiency of approximating the value of a VA portfolio, compared to brute force MC methods. A machine learning technique

known as kriging has been utilized by several researchers in the context of speeding up calculations related to VA guarantees. Kriging is a technique from spatial functional data analysis where a surface of solutions is approximated from a region of input space (Cressie 1993). It allows interpolation between previously calculated values so that, for small changes in the inputs, the resulting value can be approximated reasonably well. Gan (2013) used a *k*-prototypes clustering method (Huang 1998) to first obtain a number of representative contracts from a large portfolio of VA policies, and then used a kriging technique to estimate the values of the associated policy guarantees. Gan and Lin (2015) similarly used a two-step approach of first using a clustering technique to determine a relatively small number of representative policies and calculate the quantities of interest (such as delta or rho) for these policies; they then used universal kriging to estimate the values of these quantities for the remaining policies in the portfolio. Hejazi et al. (2017) reviewed a number of approaches falling into this category and examined these methods in the framework of general spatial interpolation. Gan and Valdez (2016) produced an empirical comparison of several methods of choosing points for the two-stage metamodeling procedures, including random sampling, and low-discrepancy sequences, among others; they concluded that the clustering and conditional Latin hypercube sampling methods produced the best results. The same authors used a version of this metamodeling procedure whereby they chose the representative VA contracts using conditional Latin hypercube sampling (Gan and Valdez 2018). Instead of then using kriging (which assumes a symmetric Gaussian distribution), they instead employed a shifted GB2 (generalized beta of the second kind) distribution; this choice allows the model to account for the tendency of market values of VA guarantees to exhibit skewness.

Neural networks are a machine learning technique based loosely on a model of the brain. Inputs are fed into an input layer and then passed to 'neurons' in a hidden layer after being multiplied by weights. The neurons in this hidden layer apply an activation function, such as a sigmoid function or a rectified linear unit (ReLU), to the weighted sum of the inputs that they receive. They then pass on this information, again in the form of weights, to neurons either in the next layer, which may be another hidden layer, or the output layer. The central property of a neural network that is useful in machine learning is that a neural network with a single hidden layer of neurons can approximate any continuous function to an arbitrary degree of accuracy, depending on the number of nodes in the hidden layer (Cybenko 1989).

Figure 1 gives an illustration of the architecture of a neural network with a single hidden layer. Each circle is a node in the network and each line is the weight given to the value being sent from one neuron to a neuron in the next layer. A layer of neurons is known as "dense" or "fully connected" when each neuron in the layer is connected to every neuron in the previous layer. A "feedforward" architecture is one composed of a linear stack of layers where information travels only in one direction through the network, i.e., there are no feedback loops. In this paper, when describing neural networks, an n_1-n_2-n_3 architecture refers to a neural network with n_1 inputs, n_2 nodes in the hidden layer, and n_3 nodes in the output layer. (Note that all neural networks utilized here employ a feedforward architecture with a single, dense hidden layer.) For an introduction to neural networks and their structure, as well as the associated terminology and their use in deep learning problems, see Nielsen (2015) or Bishop (2006).

Neural network methods and ideas have seen wide use in a variety of applications and have been used to solve problems in the insurance field for at least 25 years. A couple of early applications of neural networks to insurance problems include predicting insolvency or financial distress of insurers (Park 1993; Brockett et al. 1994; Huang et al. 1994) and prediction and regression models related to morbidity or mortality in health and dental insurance (Tu and Guerriere 1992; Saemundsson 1996; Ismael 1999). For a comprehensive account of the early history of neural network use in actuarial science, see Shapiro (2002). The use of neural networks in the actuarial area has persisted to present day, with an ever-expanding variety of applications being explored. One very recent example involves using neural networks to more accurately predict claim runoff triangles for various lines of property and casualty insurance (Kuo 2018). Indeed, researchers have used neural networks to help

to speed up calculations related to the pricing and/or hedging of variable annuities with guarantees. Hejazi and Jackson (2017) incorporated neural networks in their approach to calculating the solvency capital requirement (SCR) for a large portfolio of VA products; they found that the neural networks yield accurate calculated values while greatly increasing the efficiency of these (otherwise tedious) calculations. Hejazi and Jackson (2016) built on the spatial interpolation framework of Gan and Lin (2015) using a neural network in place of alternative methods, such as kriging. They found that this neural network approach allows them to estimate the Greeks for a portfolio of VA contracts in a manner that is both accurate as well as more efficient than the alternatives.

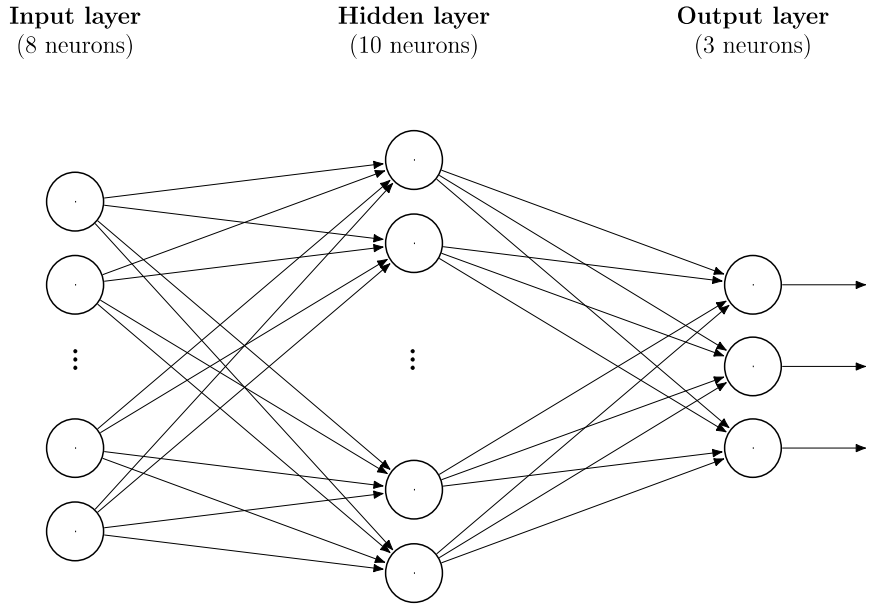


Figure 1. Sample neural network. This neural network is an example of a feedforward neural network with a single hidden layer of fully connected neurons. The architecture of this network can be described as 8-10-3.

In a similar vein, we also used neural networks in the present paper in order to exploit efficiency gains in the tasks of pricing and hedging four types of variable annuity guarantees: GMDB, GMIB, GMAB, and GMWB. Our work differs from the previous research on neural networks mentioned above in two respects. First, we demonstrated a neural network's performance on predicting liabilities for different market conditions. This demonstrates that a neural network can be used for any Greek calculation, not just deltas, and that once the network is adequately trained, daily MC calculations can be discarded. Second, we explored hedging VA guarantees with a neural network directly, which, to our knowledge, has never been previously explored. For the task of pricing these guarantees, we compared the performance of the neural network to the traditional method of using MC simulations, quantifying the gains in speed, and showing that these gains can be obtained with very little sacrifice in accuracy. With respect to the hedging of the guarantees, we implemented neural network models as an alternative to the traditional delta-rho methodology using MC simulations. We again found that utilizing these neural networks yields significant efficiency gains while very effectively hedging the liability. Further, we explored the impact of the error function used in the training of the neural network on the result for these four types of guarantees. Finally, we note that the neural network hedging methodology presented here is extremely flexible, allowing for different neural network architectures, error functions, and even underlying hedging assets.

The remainder of this article is organized as follows: Section 2 presents the models and methods (including assumptions made and data used) for the pricing and hedging of the various GMxB products, Section 3 gives the results of the analysis, comparing the results of the standard pricing

and hedging methodologies to those yielded by the neural networks, and Section 4 concludes with a discussion of the key results and ideas for future work.

2. Methods and Models

In this section, we introduce the assumptions, methods, models, and data used to price and hedge the various VA guarantees. For pricing, we used the traditional method (utilizing MC simulations to estimate the expectations needed for the pricing formulas in Section 2.2) as a baseline against which to compare the performance of the neural networks. Similarly, for hedging, we used dynamic rebalancing based on calculated delta and rho values as a baseline against which to compare the performance of the neural networks. We also discuss the training of the neural networks used for pricing and hedging.

2.1. Assumptions and Data

We made a number of standard assumptions regarding the various economic, mortality, and policy behavior processes related to the variable annuity products and their associated guarantees. Stock prices are assumed to follow the following stochastic process:

$$\frac{dS_t}{S_t} = r_t dt + \sigma_s dZ_s(t)$$

where S_t is the stock price at time t, σ_s is the volatility of the stock, $Z_s(t)$ is a standard Brownian motion process, and r_t is the short rate at time t, as defined by the Hull-White model (Hull and White 1990):

$$dr_t = (\theta(t) - a \cdot r_t)dt + \sigma_r dZ_r(t)$$

where $\theta(t)$ is determined from a zero-coupon bond yield curve, a is a mean-reversion parameter for the short rate, σ_r is the volatility of the short rate, and $Z_r(t)$ is a standard Brownian motion process. We assumed that the two standard Brownian motion processes $Z_s(t)$ and $Z_r(t)$ are independent.

The size of the wealth account associated with a VA product at time t is denoted by W_t and is assumed to follow a similar process to the stock price:

$$dW_t = W_t(r_t - \delta) + W_t\, \sigma_s dZ_s(t)$$

where δ is the fair fee charged by the insurer for the guarantee. While our assumption was that the wealth account is tied to a single stock index, Ng and Li (2013) considered a more complex multiasset framework. We made a static policyholder behavior assumption with no lapses such that 10% of the benefit base for the GMWB is withdrawn at $t = 1, 2, 3, \ldots, T$, and no other withdrawals are made for any other benefit or time. Thus, for $t = 1, 2, 3, \ldots, T$ in the case of the GMDB benefit, we have:

$$W_{t+} = \max(W_{t-} - \gamma_t,\, 0)$$

where γ_t is set to 10% of the benefit base. Deaths are assumed to follow the Gompertz-Makeham mortality model where the force of mortality for a life age x is given by:

$$\mu_x = \alpha e^{\beta x} + \lambda$$

We fit the parameters of this model using the 1994 VA MGDB ANB life table (American Academy of Actuaries 2003) and the MortalityLaws package in R (Pascariu and Canudas-Romo 2018; R Core Team 2017). The fitted parameters were $\alpha = 0.00025331$, $\beta = 0.07095565$, and $\lambda = 0.00001436$. For this study, we assumed that the VA product was issued to a policyholder age 65 and that the term of the contract was $T = 10$ years.

For the stock prices, the data we used are the S&P 500 index daily opening prices from 1980 to 1999 (Yahoo Finance 2018); we estimated the value of σ_s using the 5-year rolling historical volatility

of this index. For interest rates, we used daily Treasury Yield rate curves from 1990 to 1999 (U.S. Department of the Treasury 2018). For the parameters in the Hull-White model, we used values of $a = 0.35$ and $\sigma_r = 0.02$, and the value of $\theta(t)$ was determined from that day's Treasury yield curve.

2.2. VA Guarantee Pricing Using MC Simulations

Bacinello et al. (2011) give a thorough overview of pricing each of the common GMxBs in the VA market place. Below is a brief description of the GMxB guarantees and the formulas used to price them. For all results in this paper, 100,000 MC simulations were used to estimate the required expected values; we found that this was adequate to provide stable results.

A GMDB promises the policyholder that if the policyholder dies at some point during the life of the contract, they will receive the maximum of their account value and a guaranteed amount. This guaranteed amount will vary with the specifics of the contract. Typically, the guaranteed amount will be the initial premium, the initial premium accumulated at a specified interest rate, or the maximum account value at any anniversary of the account; we used the latter specification. A GMDB pays off if the policyholder dies during the life of the contract, and the account value is less than the guaranteed amount. If the GMDB is payable at the end of the month of death, the expected present value of the benefits is:

$$EPV_{GMDB} = \sum_{k=1}^{N} {}_{\frac{k-1}{12}|\frac{1}{12}}q_x \cdot E\left[\exp\left(-\int_0^{k/12} r_t dt\right) \cdot \max(G - W_{k/12}, 0)\right]$$

where ${}_{\frac{k-1}{12}|\frac{1}{12}}q_x$ is the probability of a policyholder age x dying in month k, N is the number of months until the expiration of the contract, and G is the largest anniversary value seen to date.

The GMAB insures that the policyholder will receive the greater of their account value and some guarantee amount if they survive to the end of the accumulation phase of the policy. A GMAB pays off if the policyholder is alive and the account value is less than the guaranteed value at the end of the contract so that the expected present value of the benefits is:

$$EPV_{GMAB} = {}_Tp_x \cdot E\left[\exp\left(-\int_0^T r_t \, dt\right) \cdot \max(G - W_T, 0)\right]$$

where T is the time until the expiration of the contract and ${}_Tp_x$ is the probability of a policyholder age x surviving T years.

The GMIB insures that the policyholder will receive a payment stream whose present value is equivalent to the greater of the account value at annuitization, or some minimum guaranteed amount. In the simplest case, the guaranteed amount is an annuitization factor multiplied by the maximum of the initial account value accumulated with interest and the highest anniversary value of the account. If the policyholder chooses a life-contingent annuity, then longevity risk along with financial risk becomes a concern (Marshall et al. 2010). For simplicity, we followed the assumption made by these authors that all policyholders annuitize uniformly after exactly ten years. A GMIB pays off if the account value is less than the price of an annuity-certain. The annuity-certain used in pricing the GMIB in this study is a 20-year annuity due that pays 5% of the ratcheted guarantee, or the account value, whichever is greater. Thus, the expected present value of the benefits is:

$$EPV_{GMIB} = E\left[\exp\left(-\int_0^T r_t \, dt\right) \cdot \max(a_{20}(t) \cdot 0.05 \cdot G - W_T, 0)\right]$$

where $a_{20}(t)$ is the present value of a 20-year annuity-due that begins payment at time T.

The GMWB insures that the policyholder will be able to withdraw some fixed amount from their account every year until the benefit amount is depleted, even if the account value is zero. The benefit base is typically set equal to the initial premium and depletes when a withdrawal is made. Pricing a GMWB involves making important assumptions about policyholder behavior. Milevsky and Salisbury (2006) and Bacinello et al. (2011) both explored the differences between making

a static assumption, where the policyholder only withdraws the amount contractually specified, and making a dynamic assumption, where the policyholder makes optimally timed withdrawals. Luo and Shevchenko (2015) considered the case of a contract with both GMWB and GMDB benefits, and how their interaction may impact on policyholder behavior. A GMWB pays off if the account value has been depleted, but the policyholder still has the right to make withdrawals. Assuming no lapses and a static withdrawal strategy, where the policyholder is allowed to withdraw up to 10% of the initial investment each year, the expected present value of the benefits is:

$$EPV_{GMWB} = \sum_{n=1}^{T} E\left[\exp\left(-\int_0^n r_t\, dt\right)\cdot\max(0.1\cdot B_n - W_n, 0)\right]$$

where B_n is the benefit base, ratcheted in proportion to the account value less withdrawals.

Assuming that fees are deducted from the account at the end of each month, the expected present value of the fees is:

$$EPV_{Fees} = \sum_{k=1}^{N} E\left[\exp\left(-\int_0^{k/12} r_t\, dt\right)\cdot\frac{\delta}{12}\cdot W_{k/12}\right]$$

and, finally, the liability for a contract on a given day is calculated as the expected present value of the benefits minus the expected present value of the fees.

2.3. Delta-Rho Hedging

A delta-rho hedging methodology was used as a baseline for the neural network's hedging performance. We assumed that two asset classes were available to be used for the hedge: Shares of the S&P 500 index and a zero-coupon bond with expiration set equal to the expiration of the VA contract, hereafter referred to for simplicity as "stocks" and "bonds". For each day, the number of stocks and bonds were found such that the combined delta and rho of the portfolio of the GMxB benefit, the stock, and the bond are equal to 0. Finding the deltas and rhos for a benefit on a given day involves two steps. First, four different values are calculated via simulation: The liability if the stock index immediately increased by 1, the liability if the stock index decreased by 1, the liability if the yield curve immediately shifted 1 basis point (bp) up, and the liability if the yield curve shifted 1 bp down. Second, the delta is found by taking the difference between the liabilities where the account value was changed and dividing by two, and the rho is found by taking the difference between the liabilities where the yield curve was changed by one bp and dividing by two. For introductory coverage of Greeks and hedging, see, e.g., McDonald (2012) or Hull (2017).

Because of how the Greeks must be calculated, delta-rho hedging increases the amount of time needed to manage a VA portfolio fourfold, because four separate sets of simulations must be run in order to calculate the delta and the rho. For companies with large VA portfolios that can already take a long time to valuate just the liabilities, a fourfold increase in computational time will be expensive, and a hedging solution that does not require as much runtime is preferable.

2.4. Neural Network Pricing Methodology

An alternative to pricing the liability at every day with a Monte Carlo simulation is to train a neural network to predict the liability based only on the same inputs fed to the simulation. (Note that not all of the inputs that are fed to the Monte Carlo simulation are fed to the network, because they are the same for all simulations, i.e., the mortality parameters and Hull-White interest rate model parameters.) Thus, the list of inputs to the pricing neural network is as follows:

- Time until maturity of the contract;
- Spot rate of the yield curve;
- S&P 500 opening price;
- Price of a zero-coupon bond maturing at the end of the contract;

- Volatility of the S&P 500;
- The first three components of a principal component analysis of the yield curve, which are generally interpreted as the height, slope, and curvature of the yield curve;
- Fee for the guarantee;
- "Moneyness" (M) of the guarantee, defined as $M = (G - V)/G$, where V is the account value. For the GMWB, G is the amount that the policyholder is allowed to withdraw in a given year. For the other guarantee types, G is the amount guaranteed to the policyholder on their death, survival, or annuitization, respectively. Because this amount ratchets up on policy anniversaries, the guarantee amount represents the highest value seen to date at a policy anniversary.

The sole output of the neural network is the value of the liability for the guarantee. A single hidden layer of ten neurons was used for this neural network, so that the architecture of the network is 10-10-1. Matching the number of neurons in the hidden layer to that in the input layer seems like a reasonable heuristic; this approach has been used to good effect in the past by various researchers (e.g., Møller 1993) and produced good results for this study, in the sense that we encountered no problems with overfitting. Nonetheless, when we experimented with other architectures, we found that, in general, increasing the number of nodes in the hidden layer to 15 or 20 only results in marginally better performance, at the expense of longer training times.

2.5. Neural Network Hedging Methodology

An alternative to the classic hedging described above is hedging by training a neural network to buy/sell assets that will minimize the percentage change in the net position of the portfolio of VA liabilities and assets from day to day. This is done by constructing a neural network with the same inputs as the neural network described above to price the guarantees. Like the neural network used for pricing, the network contains a single hidden layer of ten nodes. However, instead of the output layer only containing one node for the liability value, the output layer contains nodes that correspond to different assets used to hedge the portfolio. In this paper, the only assets we assumed available for constructing the hedging portfolio were the same two assets used in the delta-rho hedging methodology, namely "stocks" and "bonds" as described above. Thus, the architecture of the network can be described as 10-10-2. The logic for choosing this architecture parallels that for the pricing neural network, though other configurations could be used. Also, while we only chose to use the two specified assets for this study, we note that one of the benefits of this methodology is its flexibility; it could be made to work with many different sets of available assets.

To summarize the methodology, this hedging neural network uses the same inputs as used in the pricing neural network described above and puts out the quantity of each asset to buy/sell and can be trained to buy/sell assets that will minimize some error function related to the goal of creating the hedge. This reduces the computational cost for obtaining the information needed to hedge the portfolio because a single pass through a neural network involves significantly less computation than simulating a large number of economic paths four different times to find the delta and the rho.

2.6. Neural Network Training

An important aspect of using a neural network for machine learning is training the weights between the nodes in order to better learn the relationship between model inputs and outputs. The simplest algorithm is a so-called "gradient descent", where the derivative of some error function (typically a sum of squares error function for regression applications, or a cross-entropy error function for classification problems) is taken with respect to each weight. Then, each weight is updated by moving it in the negative direction of its element of the gradient, scaled by some learning rate. A challenge associated with this approach is that if the learning rate is too large, the network will skip over global or local minima and never converge, and if the learning rate is too small, training will take a long time to converge. For this study, the error function chosen was a key determinant in

the performance of the neural network; in Section 3.2, we explore the ramifications of changing the error function.

Alternatives to gradient descent include the resilient backpropagation algorithm (RPROP), scaled conjugate gradient descent (SCGD), and the Broyden-Fletcher-Goldsharb-Shanno algorithm (BFGS). (Riedmiller 1994; Møller 1993) RPROP is an algorithm that updates each weight only based on the sign of the gradient. It then increases the size of the step for each weight as long as that weight maintains the sign for its gradient. If the sign of the gradient for a particular weight changes, the size of the step for that weight is decreased and the step size begins moving in the opposite direction. RPROP is fast, initially, because it uses only first order information but can be slower to converge overall because the use of only first order information makes it difficult for the algorithm to converge towards the end of training.

SCGD is an algorithm that improves on simple gradient descent using a second order approximation to the error function to find an appropriate learning rate at each iteration. BFGS uses a more complete second order approximation and also a line search to determine the appropriate step size and direction for each weight.

For the results in this paper, the BFGS algorithm was used as implemented in the nnet package in R (Venables and Ripley 2002; R Core Team 2017). However, parts of the analysis were also run with the SCGD algorithm, with comparable performance. We found that the RPROP algorithm provides a faster initial reduction in error but converges to a desired level of accuracy more slowly than with the other two algorithms, and sometimes fails to converge entirely.

We prepared the dataset by splitting the data into a training set and a testing set. Specifically, 80% of the data were used for the training set, while the remaining 20% were used for testing; we randomly chose the points to be placed into each set. There were in total 2530 data points, 2023 of which were used for training, while the remaining 507 were used for testing. This method of partitioning the data for training and testing (i.e., a randomly chosen 80/20 split) is rather common in neural network applications.

3. Results

In this section, we describe the results obtained, comparing the results of the neural networks with those from the traditional pricing and hedging methods described in Section 2.

3.1. Pricing

We first calculated the fair value (price) of each of the VA GMxB riders described above as the price that makes the expected present value of the fees equal to the expected present value of the corresponding benefit; these calculations were done via MC simulations, using the formulas, data, and methodology described in Section 2.2. Table 1 gives the calculated prices for each of the four GMxB guarantees.

Table 1. Prices of GMxB guarantees (in bp) calculated via Monte Carlo (MC) simulations.

GMDB	GMAB	GMIB	GMWB
33	93	2.6	95

Our main goal was not to conduct an exhaustive study of the fair prices of the various guarantees under varying assumptions, but rather to study the relative effectiveness of using neural networks to approximate the results of the MC simulations. As such, we were not overly concerned about the specific prices obtained for these particular guarantees. We do note, however, that these calculated prices seem reasonable, and are generally consistent with the prices obtained for similar VA guarantees in previous studies, though the wide variety of product structures, methodologies, and choices of assumptions make direct comparisons problematic. For the GMDB, Bauer et al. (2008) reported typical

fees charged in the industry of 25–60 bp. Bacinello et al. (2011) studied the prices of multiple guarantees (and combinations thereof) under a variety of parameter values and reported prices significantly less than this; in fact, they concluded that the standalone GMDB is "practically valueless". For the GMWB, our calculated price lies a bit above the range of typical fees charged reported by Bauer et al. (2008) of 25–75 bp. Milevsky and Salisbury (2006) reported a range typical charges of 30–45 bp, but found that the fair price should be in the range of 73–160 bp. Figure 2 gives the value of the liability over the life of the contract for the GMAB and GMWB guarantees. Note that the liability values can spike up or down very rapidly, potentially making the task of hedging these liabilities increasingly difficult.

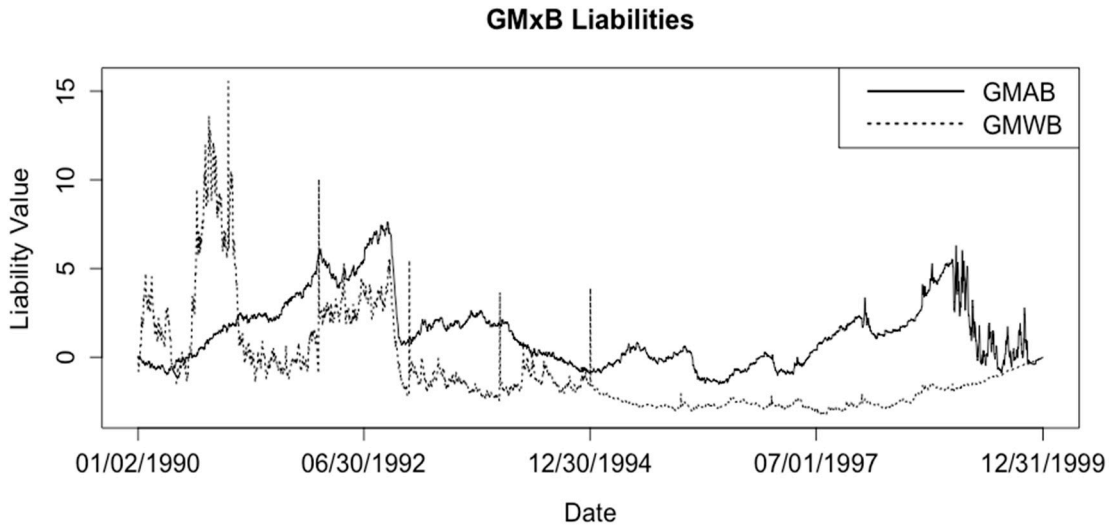


Figure 2. Liability values of two of the four GMxB guarantees over the period 1990–1999.

For each of the four VA GMxB riders described above, we trained the 10-10-1 pricing neural network using the training methodology described in Section 2. We then compared the results of the outputs produced by the neural network to those calculated using MC simulations. In all cases, the liability values were split 80/20 into training and testing sets.

In general, we saw strong evidence that the neural network performs quite well relative to the standard MC simulation methodology. We measured the performance of neural network approximations using R^2, a measure has been used by other researchers to measure neural network accuracy, as in, e.g., Hattab et al. (2013). There are, however, several other options for neural network performance metrics; a few such possibilities include correlation (Chen and Sutcliffe 2012), relative error (Hejazi and Jackson 2016), and mean square prediction error (Garcia and Gençay 2000). We expect that any of these metrics would have yielded substantially the same results. Table 2 gives the R^2 values calculated when comparing the neural networks values to the corresponding values calculated by MC simulations.

Table 2. Values of R^2 for GMxB guarantees calculated via neural networks and MC simulations.

GMDB	GMAB	GMIB	GMWB
0.9972	0.9952	0.9986	0.9870

Using this R^2 measure, we can see that the neural network performs quite well for all four types of guarantees, though the performance for the GMWB is considerably worse than for the other three guarantees; this is primarily due to the presence of a few outliers. A scatterplot of the results for the GMDB is shown in Figure 3; from this figure, we can see a high degree of accuracy and no evidence of systematic biases or patterns. The scatterplots for the other guarantees show similar results.

The main advantage of the neural network approach over MC methods lies in the potential time savings in completing the required calculations. For the machine used in this study (i7-6500U CPU@2.50 GHz, 2601 MHz, 2 Cores, 4 Logical Processors), it takes about eight hours to price the set of four liabilities over the 10-year time frame using the MC simulations. The neural network approach, by contrast, runs in just over one minute. As mentioned previously, training a neural network involves repeatedly passing each training input into the network, evaluating the gradient of the network weights for the entire training data set, and updating the weights based on this information. Therefore, generally speaking, the larger the network and the larger the size of the training data, the longer the training time will be. However, once the network is trained, evaluating the values of the entire dataset occurs quickly and scales with the size of the network, not the size of the training data. Evaluating the entire dataset once trained never took longer than 3 milliseconds for any of the models used. Note that this is an advantage that a neural network possesses over a method like kriging, which involves distance calculations between the input it is fed and the entire training data it has been attuned to, causing its complexity to scale with the size of its training data. Thus, for the pricing task, the neural network methodology provided very significant time savings at the cost of a miniscule sacrifice of accuracy. Recall that the present study involves a single VA contract; the improvement in efficiency could be substantial when scaled to a portfolio of such contracts.

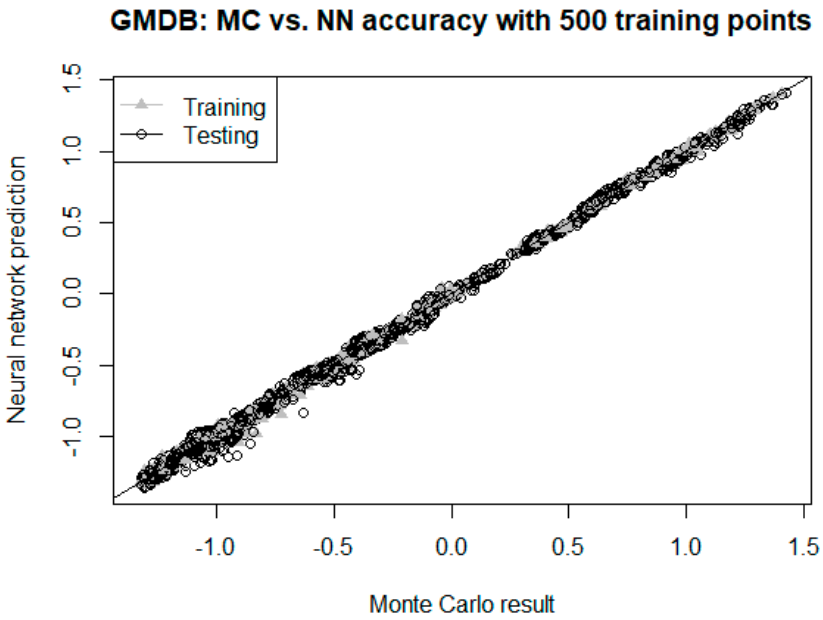


Figure 3. Scatterplot of the results of using a neural network to approximate the MC pricing methodology for the guaranteed minimum death benefit (GMDB). For this guarantee, the neural network (NN) achieves an R^2 better than 0.997.

3.2. Hedging

There are different criteria that have been proposed for evaluating the effectiveness of a hedge. Cotter and Hanly (2006) provided five different criteria for evaluating a hedge: Variance of returns, semivariance of returns, the lower partial moment of returns, the value at risk, and the conditional value at risk. For simplicity, this paper only compared hedging performances numerically with the variance/standard deviation of returns. We should note, however, that certain hedging positions that the neural network produces would lead to a much larger value at risk and conditional value at risk than the comparable delta-rho hedging methodology. We assumed that the portfolio was rebalanced daily. We first consider in detail the results of the GMDB, and then discuss results from the other guarantees.

We used the following hedging metric, which is a slightly modified version of HE_1 from Cotter and Hanly (2006) that uses standard deviation instead of variance:

$$HE_1 = 1 - \frac{SD_H}{SD_U}$$

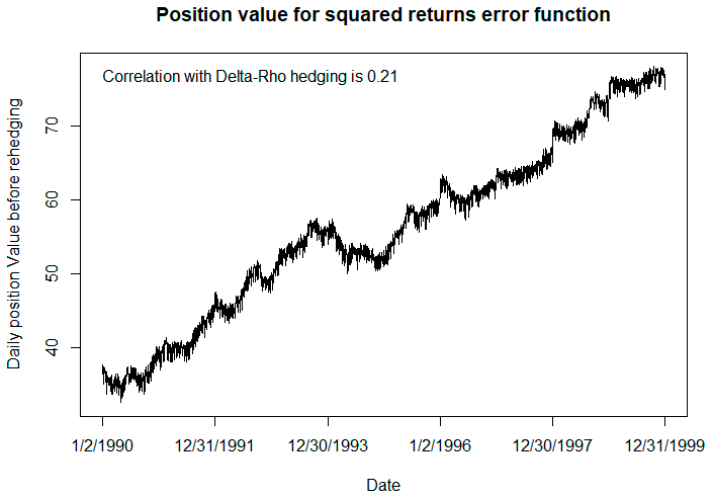
where SD_H is the standard deviation of the portfolio's returns with hedging and SD_U is the standard deviation of the portfolio's returns without hedging. Using the standard deviation instead of the variance provides a more conservative metric for the degree to which a hedging strategy reduces the portfolio's risk. For this metric, a value close to 1 is good because hedging reduces the standard deviation of returns almost completely. A value close to 0 or negative is bad, because that means that hedging either produces similar or worse standard deviations of returns than leaving the portfolio unhedged.

As discussed in Section 2.6, neural network training requires the choice of a suitable error function. The first error function used to train the network is the sum of the squared errors of the percentage change of the position value (net difference of assets and liabilities) from 0 for each day:

$$Error = \sum_{i=2}^{D} \left[\frac{A_i - L_i}{A_{i-1} - L_{i-1}} - 1 \right]^2$$

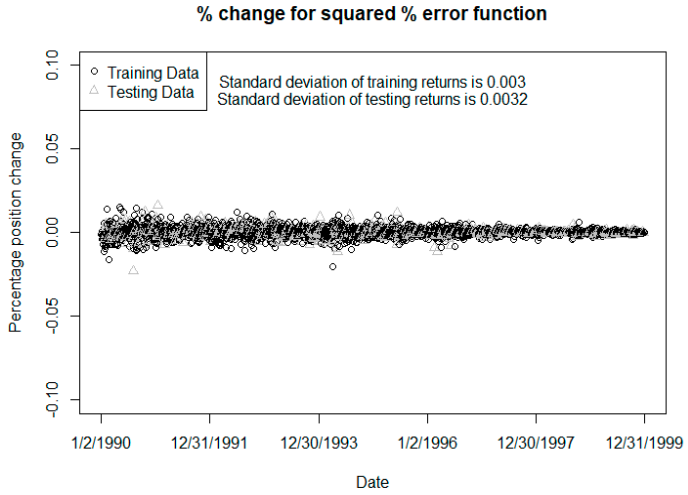
where A_i and L_i are the values of the assets and liabilities on day i, respectively, and D is the total number of days under consideration for the contract. Assets are the stocks and bonds used to hedge a given guarantee and liability is the guarantee itself. Graphs of the total value and the percentage changes of the hedged position produced from this error function for the GMDB are shown in Figure 4 below.

This error function does an excellent job hedging the guarantees from the perspective of minimizing the standard deviation of returns. However, it does not shy away from overinvesting in the market to achieve this low standard deviation. The reason for this is that VA liabilities are volatile, and thus investing in stocks and bonds, which are less volatile, would decrease the risk of the whole portfolio. Near the end of the time period, the network uses around \$60–\$70 of assets to hedge −\$0.50 to −\$1 of liabilities, as opposed to the delta-rho hedging, which mostly uses −\$10 to −\$20 to hedge the same amount. While this performs the job of minimizing the standard deviation of returns, it would lead to a higher value at risk by requiring more capital to manage the portfolio and would hence decrease the portfolio's Return on Equity.



(**a**)

Figure 4. *Cont.*



(**b**)

Figure 4. Results for hedging for GMDB using squared error function: (**a**) The daily position value over the 10-year life of the variable annuities (VA) contract; (**b**) the daily percentage change in the position over the life of the contract, including both training and testing.

An alternative to the above error function is an error function that ignores errors below a certain threshold, k:

$$Error = \sum_{i=2}^{D} \left[\max\left(\left(\frac{A_i - L_i}{A_{i-1} - L_{i-1}} - 1 \right)^2 - k, 0 \right) \right].$$

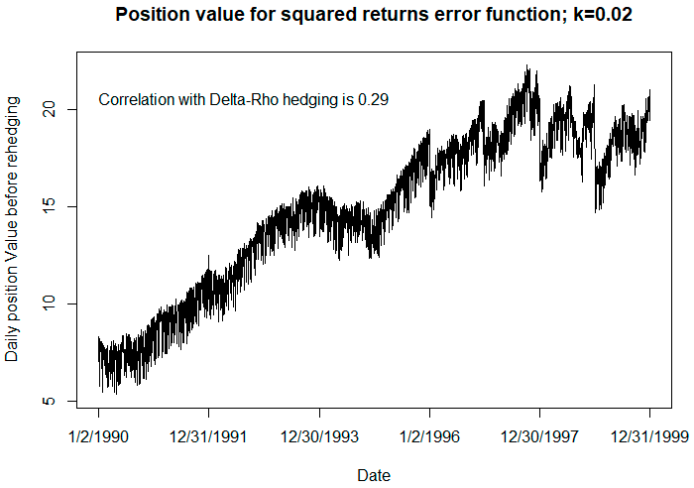
The idea behind this error function is to not reward the network for overhedging small changes in the liabilities, i.e., returns that are within k of 1, and while this error function does not discourage overhedging, it at least does not encourage it. Note that the previous error function is a special case of this error function where $k = 0$.

This error function has mixed results; see Table 3 for the values of HE_1 for different values of k. For values of k equal to 0.01 and 0.02, the network reduces the total level of assets used to hedge the guarantee, while still providing a significant decrease in the standard deviation of returns, as was intended. However, once k increases to 0.03 and 0.04, the network converges to a solution that almost exactly matches the actual account value of the GMDB account, while also using more assets than the previous strategies. For all values of k, the network generalizes well from training to testing and outperforms the delta-rho strategy considerably, in terms of its ability to reduce the standard deviation of returns, while providing a significant time savings. (Note that the delta-rho strategy does not require the data to be split into training and testing sets; the values of HE_1 reported for this strategy apply to the entire dataset).

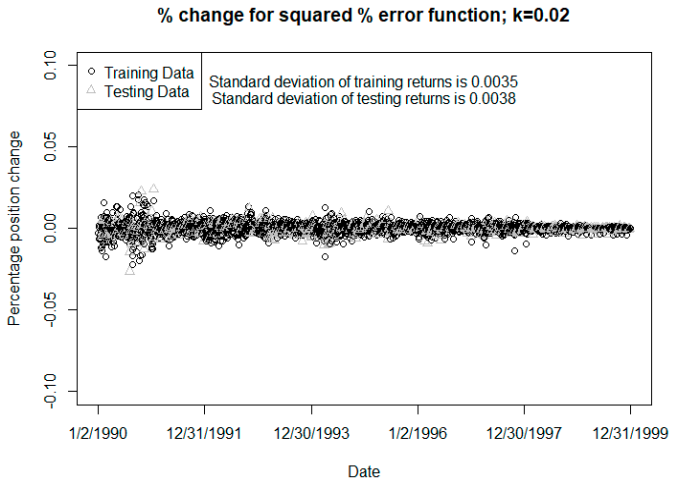
Table 3. Hedging metrics for different hedging strategies and error functions for GMDB.

Hedging	Delta-Rho	k = 0	k = 0.01	k = 0.02	k = 0.03	k = 0.04
Training HE_1	0.8805	0.9965	0.9964	0.9958	0.9932	0.9895
Testing HE_1		0.9962	0.9961	0.9955	0.9929	0.9891
Time (s)	~4200	178	134	139	63	51

Figure 5 shows the total value and the percentage changes of the hedged position produced from the $k = 0.02$ error function for the GMDB. Compared to the original error function, we can see that the $k = 0.02$ error function is more likely to result in greater swings in the daily positions.



(**a**)



(**b**)

Figure 5. Results for hedging for GMDB using $k = 0.02$ error function: (**a**) The daily position value over the 10-year life of the VA contract; (**b**) the daily percentage change in the position over the life of the contract, including both training and testing.

Table 4 gives the hedging metrics from the GMAB. For this guarantee, we experimented with values of k ranging from 0 to 0.01; we found that for this guarantee, smaller values of k tended to produce better results. The main reason that smaller values of k were advantageous here is that this guarantee tends to be less volatile than some of the others, such as the GMDB. Hence, the threshold can be lowered accordingly, producing better results in terms of the hedging metric. While the neural network does not outperform the MC simulations to as large of an extent as in some of the other guarantees, these results are nonetheless encouraging: We can see that the performance of the neural network is roughly comparable to, or superior to the delta-rho hedging methodology for the smaller values of the threshold k, while performing at a significant decrease in time required for the calculations.

Table 4. Hedging metrics for different hedging strategies and error functions for guaranteed minimum accumulation benefit (GMAB).

Hedging	Delta-Rho	$k = 0$	$k = 0.0025$	$k = 0.005$	$k = 0.0075$	$k = 0.01$
Training HE_1	0.9956	0.9991	0.9967	0.9961	0.9957	0.9957
Testing HE_1		0.9984	0.9945	0.9937	0.9931	0.9930
Time (s)	~4200	207	70	76	65	67

Table 5 gives the hedging metrics from the GMIB. Similarly to the GMAB, we used somewhat smaller values of k for this guarantee, for the same reason, namely that the value of this guarantee was relatively less volatile, thus requiring a lower threshold. The results are similar to those for the GMAB: The neural network provides a comparable hedging performance to delta-rho hedging at significant savings in computational time. We also note that in the case of $k = 0$ for this guarantee, the neural network uses significantly less assets to form the hedge than in the cases of the other threshold values.

Table 5. Hedging metrics for different hedging strategies and error functions for guaranteed minimum income benefit (GMIB).

Hedging	Delta-Rho	$k = 0$	$k = 0.001$	$k = 0.002$	$k = 0.003$	$k = 0.004$
Training HE_1	0.9648	0.9966	0.9938	0.9937	0.9937	0.9937
Testing HE_1		0.9981	0.9647	0.9645	0.9645	0.9645
Time (s)	~4200	429	63	27	26	27

Table 6 gives the hedging metrics from the GMWB. For this guarantee, we see that while the neural network does better for the smaller values of k, it significantly outperforms the delta-rho method in terms of our hedging metric, for all values of k. Due to poor performance, it was necessary to slightly modify the neural network architecture for this guarantee, though; instead of the 10-10-2 design used for the other guarantees, we instead implemented a 10-13-2 architecture, as this was the minimum hidden layer size needed to get adequate performance.

Table 6. Hedging metrics for different hedging strategies and error functions for guaranteed minimum withdrawal benefit (GMWB).

Hedging	Delta-Rho	$k = 0$	$k = 0.01$	$k = 0.02$	$k = 0.03$	$k = 0.04$
Training HE_1	0.9717	0.9991	0.9966	0.9961	0.9960	0.9955
Testing HE_1		0.9978	0.9896	0.9869	0.9863	0.9828
Time (s)	~4200	400	55	46	49	47

4. Conclusions

The results of the previous section demonstrate that neural networks can indeed be used effectively to price as well as hedge various types of VA GMxB guarantees. When using neural networks in this manner, we see a significant gain in efficiency (speed) when compared to the traditional pricing and hedging methodologies. In some cases, the performance of the neural network was markedly superior, whereas in other cases, the performance is comparable to traditional methods, but in all cases, the relative efficiency gains were quite significant.

In some hedging scenarios, the neural network can tend to overinvest to assets to form the hedge portfolio, sometimes resulting in unreasonably large capital requirements. Using different tolerance thresholds in the error function when training the network can largely control this tendency and can significantly impact on the results; there is often a tradeoff between quality of the hedge (as measured by our HE_1 metric) and the cost of capital required for the hedging assets. Of course, whether this tradeoff is worthwhile will depend on the cost of capital relative to the benefit of the more efficient calculations, cost of computing resources, and other considerations.

In this paper, we have demonstrated that there can be significant efficiency gains over traditional models when we utilize neural networks for the pricing of various VA guarantees; we found these gains consistently over four of the more popular types of GMxB guarantees: GMDB, GMAB, GMDB, and GMWB. We have also shown that it is possible to use neural networks to hedge these types of VA guarantees, as an alternative to the traditional delta-rho methodology using MC simulations. We have experimented with different tolerances (thresholds) in the error function used to train the neural

networks and have examined the resulting hedging performance using a variant of the HE_1 metric. In all cases, the neural networks exhibited promising results.

There are several possible avenues for future work in this line of research. We have shown that the error function can have a significant effect on the performance of a neural network; while we have experimented with various thresholds, it may also be worthwhile to consider different forms of error functions and to measure the resulting performance, especially given the large impact that the error function and threshold has been shown to make in terms of both hedging performance and speed. In addition, while the HE_1 hedging metric we used here seems to be a reasonable method for capturing the performance of the hedge for these guarantees, many other metrics are possible and have been used by various researchers. It would be instructive to repeat this type of study using some of the other measures. In addition, it may be helpful to generalize the results obtained here by applying this methodology to other time periods. The time period used here was chosen arbitrarily in order to demonstrate the methodology, but a deeper look into the relative performance of the neural network over different times periods spanning a broad range of interest rate and stock market conditions would seem to be warranted. Further, while our model accounts for mortality, it does not incorporate elements of policyholder behavior, such as lapses or dynamic withdrawal behavior. As neural networks are very flexible, they could certainly be trained to incorporate these important aspects of variable annuities. Finally, one of the most significant benefits of using the neural network to hedge is the flexibility of choice of hedging assets. While we used only two assets here, namely a stock index and bond, the neural network approach could be used with virtually any number and/or type of assets. There is an opportunity to explore this idea and determine which combinations of asset types are most effective for hedging the various types of VA guarantees.

Author Contributions: Conceptualization, D.D.; Methodology, D.D. and C.G.; Software, D.D.; Validation, D.D. and C.G.; Formal Analysis, D.D.; Investigation, D.D.; Resources, D.D.; Data Curation, D.D.; Writing—Original Draft Preparation, D.D.; Writing—Review and Editing, D.D. and C.G.; Visualization, D.D. and C.G.; Supervision, C.G.; Project Administration, D.D. and C.G.

Funding: This research received no external funding.

Acknowledgments: The authors would like to thank two anonymous reviewers, whose suggestions greatly improved the quality of this manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

American Academy of Actuaries. 2003. *Actuarial Guideline XXXIV: Variable Annuity Minimum Guaranteed Death Benefit Reserves*. Washington, DC: American Academy of Actuaries.

Armstrong, Michael J. 2001. The Reset Decision for Segregated Fund Maturity Guarantees. *Insurance: Mathematics and Economics* 29: 257–69. [CrossRef]

Bacinello, Anna Rita, Pietro Millossovich, Annamaria Olivieri, and Ermanno Pitacco. 2011. Variable Annuities: A Unifying Valuation Approach. *Insurance: Mathematics and Economics* 49: 285–97. [CrossRef]

Bauer, Daniel, Alexander Kling, and Jochen Russ. 2008. A Universal Pricing Framework for Guaranteed Minimum Benefits in Variable Annuities. *ASTIN Bulletin: The Journal of the IAA* 38: 621–51. [CrossRef]

Bishop, Christopher. 2006. *Pattern Recognition and Machine Learning*. Berlin: Springer.

Boyle, Phelim P., and Mary R. Hardy. 1997. Reserving for Maturity Guarantees: Two Approaches. In Honor of Prof. J. A. Beekman. *Insurance: Mathematics and Economics* 21: 113–27. [CrossRef]

Boyle, Phelim P., and Eduardo S. Schwartz. 1977. Equilibrium Prices of Guarantees under Equity-Linked Contracts. *The Journal of Risk and Insurance* 44: 639–60. [CrossRef]

Brennan, Michael J., and Eduardo S. Schwartz. 1976. The Pricing of Equity-Linked Life Insurance Policies with an Asset Value Guarantee. *Journal of Financial Economics* 3: 195–213. [CrossRef]

Brockett, Patrick L., William W. Cooper, Linda L. Golden, and Utai Pitaktong. 1994. A Neural Network Method for Obtaining an Early Warning of Insurer Insolvency. *Journal of Risk and Insurance* 61: 402–24. [CrossRef]

Chen, Fei, and Charles Sutcliffe. 2012. Pricing and Hedging Short Sterling Options Using Neural Networks. *Intelligent Systems in Accounting, Finance and Management* 19: 128–49. [CrossRef]

Chollet, François, and Joseph J. Allaire. 2018. *Deep Learning with R*. Shelter Island: Manning Publications Company.

Cotter, John, and Jim Hanly. 2006. Reevaluating Hedging Performance. *Journal of Futures Markets: Futures, Options, and Other Derivative Products* 26: 677–702. [CrossRef]

Cressie, Noel. 1993. *Statistics for Spatial Data*, 2nd ed. Hoboken: Wiley.

Cybenko, George. 1989. Approximation by Superpositions of a Sigmoidal Function. *Mathematics of Control, Signals and Systems* 2: 303–14. [CrossRef]

Feng, Runhuan, and Hans W. Volkmer. 2012. Analytical Calculation of Risk Measures for Variable Annuity Guaranteed Benefits. *Insurance: Mathematics and Economics* 51: 636–48. [CrossRef]

Feng, Runhuan, and Hans W. Volkmer. 2014. Spectral Methods for the Calculation of Risk Measures for Variable Annuity Guaranteed Benefits. *ASTIN Bulletin: The Journal of the IAA* 44: 653–81. [CrossRef]

Gan, Guojun. 2013. Application of Data Clustering and Machine Learning in Variable Annuity Valuation. *Insurance: Mathematics and Economics* 53: 795–801.

Gan, Guojun. 2018. Valuation of Large Variable Annuity Portfolios Using Linear Models with Interactions. *Risks* 6: 71. [CrossRef]

Gan, Guojun, and Sheldon X. Lin. 2015. Valuation of Large Variable Annuity Portfolios under Nested Simulation: A Functional Data Approach. *Insurance: Mathematics and Economics* 62: 138–50. [CrossRef]

Gan, Guojun, and Sheldon X. Lin. 2017. Efficient Greek Calculation of Variable Annuity Portfolios for Dynamic Hedging: A Two-Level Metamodeling Approach. *North American Actuarial Journal* 21: 161–77. [CrossRef]

Gan, Guojun, and Emiliano A. Valdez. 2016. An Empirical Comparison of Some Experimental Designs for the Valuation of Large Variable Annuity Portfolios. *Dependence Modeling* 4. [CrossRef]

Gan, Guojun, and Emiliano A. Valdez. 2018. Regression Modeling for the Valuation of Large Variable Annuity Portfolios. *North American Actuarial Journal* 22: 40–54. [CrossRef]

Garcia, René, and Ramazan Gençay. 2000. Pricing and Hedging Derivative Securities with Neural Networks and a Homogeneity Hint. *Journal of Econometrics* 94: 93–115. [CrossRef]

Hardy, Mary. 2000. Hedging and Reserving for Single-Premium Segregated Fund Contracts. *North American Actuarial Journal* 4: 63–74. [CrossRef]

Hardy, Mary. 2003. *Investment Guarantees: Modeling and Risk Management for Equity-Linked Life Insurance*. Hoboken: John Wiley & Sons, vol. 215.

Hattab, Nour, Ridha Hambli, Mikael Motelica-Heino, and Michel Mench. 2013. Neural Network and Monte Carlo Simulation Approach to Investigate Variability of Copper Concentration in Phytoremediated Contaminated Soils. *Journal of Environmental Management* 129: 134–42. [CrossRef] [PubMed]

Hejazi, Seyed Amir, and Kenneth R. Jackson. 2016. A Neural Network Approach to Efficient Valuation of Large Portfolios of Variable Annuities. *Insurance: Mathematics and Economics* 70: 169–81. [CrossRef]

Hejazi, Seyed Amir, and Kenneth R. Jackson. 2017. Efficient Valuation of SCR via a Neural Network Approach. *Journal of Computational and Applied Mathematics* 313: 427–39. [CrossRef]

Hejazi, Seyed Amir, Kenneth R. Jackson, and Guojun Gan. 2017. A Spatial Interpolation Framework for Efficient Valuation of Large Portfolios of Variable Annuities. *arXiv*. arXiv:1701.04134.

Huang, Zhexue. 1998. Extensions to the K-Means Algorithm for Clustering Large Data Sets with Categorical Values. *Data Mining and Knowledge Discovery* 2: 283–304. [CrossRef]

Huang, Chin-Sheng, Robert E. Dorsey, and Mary Ann Boose. 1994. Life Insurer Financial Distress Prediction: A Neural Network Model. *Journal of Insurance Regulation* 13: 131.

Hull, John. 2017. *Options, Futures, and Other Derivatives*, 10th ed. London: Pearson.

Hull, John, and Alan White. 1990. Pricing Interest-Rate-Derivative Securities. *The Review of Financial Studies* 3: 573–92. [CrossRef]

Insured Retirement Institute. 2018. Fourth Quarter 2017 Annuity Sales Report. Insured Sales Institute. Available online: http://www.irionline.org/newsroom/newsroom-detail-view/iri-issues-fourth-quarter-2017-annuity-sales-report (accessed on 23 August 2018).

Ismael, Mohammad Belall. 1999. Prediction of Mortality and In-Hospital Complications for Acute Myocardial Infarction Patients Using Artificial Neural Networks. Ph.D. thesis, Duke University, Durham, NC, USA.

Ko, Bangwon, Elias S. W. Shiu, and Li Wei. 2010. Pricing Maturity Guarantee with Dynamic Withdrawal Benefit. *Insurance: Mathematics and Economics* 47: 216–23. [CrossRef]

Kolkiewicz, Adam, and Yan Liu. 2012. Semi-Static Hedging for GMWB in Variable Annuities. *North American Actuarial Journal* 16: 112–40. [CrossRef]

Kuo, Kevin. 2018. DeepTriangle: A Deep Learning Approach to Loss Reserving. *arXiv*. arXiv:1804.09253.

Ledlie, M. Colin, Dermot P. Corry, Gary S. Finkelstein, Alan J. Ritchie, Ken Su, and Colin Wilson. 2008. Variable Annuities. *British Actuarial Journal* 14: 327–89. [CrossRef]

Lim, Michael, and Trevor Hastie. 2015. Learning Interactions via Hierarchical Group-Lasso Regularization. *Journal of Computational and Graphical Statistics* 24: 627–54. [CrossRef]

LIMRA. 2018. U.S. Individual Annuities Survey. LIMRA Secure Retirement Institute. Available online: https://www.limra.com/Posts/PR/Data_Bank/_PDF/2008-2017-Annuity-Sales-Estimates.aspx (accessed on 23 August 2018).

Luo, Xiaolin, and Pavel V. Shevchenko. 2015. Valuation of Variable Annuities with Guaranteed Minimum Withdrawal and Death Benefits via Stochastic Control Optimization. *Insurance: Mathematics and Economics* 62: 5–15. [CrossRef]

Marshall, Claymore, Mary Hardy, and David Saunders. 2010. Valuation of a Guaranteed Minimum Income Benefit. *North American Actuarial Journal* 14: 38–58. [CrossRef]

McDonald, Robert. 2012. *Derivatives Markets*, 3rd ed. London: Pearson.

Milevsky, Moshe A., and Thomas S. Salisbury. 2006. Financial Valuation of Guaranteed Minimum Withdrawal Benefits. *Insurance: Mathematics and Economics* 38: 21–38. [CrossRef]

Møller, Martin Fodslette. 1993. A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning. *Neural Networks* 6: 525–33. [CrossRef]

Ng, Andrew Cheuk-Yin, and Johnny Siu-Hang Li. 2013. Pricing and Hedging Variable Annuity Guarantees with Multiasset Stochastic Investment Models. *North American Actuarial Journal* 17: 41–62. [CrossRef]

Nielsen, Michael A. 2015. Neural Networks and Deep Learning. Available online: http://neuralnetworksanddeeplearning.com (accessed on 8 February 2018).

Park, Jeongsun. 1993. Bankruptcy Prediction of Banks and Insurance Companies: An Approach Using Inductive Methods. Ph.D. thesis, University of Texas at Austin, Austin, TX, USA.

Pascariu, Marius D., and Vladimir Canudas-Romo. 2018. MortalityLaws: Parametric Mortality Models, Life Tables and HMD (Version 1.5.0). Available online: https://CRAN.R-project.org/package=MortalityLaws (accessed on 1 August 2018).

R Core Team. 2017. *R: A Language and Environment for Statistical Computing*. Vienna: R Foundation for Statistical Computing, Available online: https://www.R-project.org/ (accessed on 14 September 2017).

Riedmiller, Martin. 1994. *RPROP: Description and Implementation Details*. Technical Report. Karlsruhe: University of Karlsruhe.

Saemundsson, Sigurḥur Rúnar. 1996. Dental Caries Prediction by Clinicians and Neural Networks. Ph.D. thesis, Duke University, Durham, NC, USA.

Shapiro, Arnold F. 2002. The Merging of Neural Networks, Fuzzy Logic, and Genetic Algorithms. *Insurance: Mathematics and Economics* 31: 115–31. [CrossRef]

Tu, Jack V., and Michael R. Guerriere. 1992. Use of a Neural Network as a Predictive Instrument for Length of Stay in the Intensive Care Unit Following Cardiac Surgery. Paper presented at Annual Symposium on Computer Application in Medical Care, Washington, DC, USA, October 30–November 1.

U.S. Department of the Treasury. 2018. Daily Treasury Yield Curve Rates. Available online: https://www.treasury.gov/resource-center/data-chart-center/interest-rates/Pages/TextView.aspx?data=yieldAll (accessed on 19 January 2018).

Venables, William N., and Brian D. Ripley. 2002. *Modern Applied Statistics with S*, 4th ed. New York: Springer. Available online: http://www.stats.ox.ac.uk/pub/MASS4 (accessed on 17 May 2018).

Windcliff, Heath, Peter A. Forsyth, and Kenneth R. Vetzal. 2001. Valuation of Segregated Funds: Shout Options with Maturity Extensions. *Insurance: Mathematics and Economics* 29: 1–21. [CrossRef]

Windcliff, Heath, Martin Le Roux, Peter Forsyth, and Kenneth Vetzal. 2002. Understanding the Behavior and Hedging of Segregated Funds Offering the Reset Feature. *North American Actuarial Journal* 6: 107–24. [CrossRef]

Yahoo Finance. 2018. ^GSPC Historical Prices | S&P 500 Stock—Yahoo Finance. Available online: https://finance.yahoo.com/quote/%5EGSPC/history/ (accessed on 19 January 2018).