Application of deep reinforcement learning in asset liability management

By Takura Wekwete

Presented at the Actuarial Society of South Africa's 2023 Convention
Sandton Convention Centre 11–12 October 2023

ABSTRACT

Asset Liability Management (ALM) is an essential actuarial and quantitative finance risk management technique. The ultimate objective is to maximise an institutional risk-taker's ability to fulfil its current and future liabilities. Optimal ALM is especially vital in environments with elevated interest rate movements, as has been experienced globally between 2021 and 2023. Traditional ALM's industry implementation is still heavily dependent on the judgement of professionals such as quants, actuaries, and investment managers. This traditional approach is usually combined with Redington immunisation theory, but this theoretical framework has numerous limitations and assumptions that are often imprecise in application. Furthermore, the over-reliance on human professional judgement acutely limits ALM performance due to restricted automation, human irrationality, and restricted scope for multi-objective optimisation. This paper addresses these limitations by introducing a new approach to ALM based on Deep Reinforcement Learning (DRL). Deep reinforcement learning incorporates deep learning into the reinforcement learning framework. Combining deep learning and reinforcement learning, which complement each other well, gives a deep reinforcement learning solution with both strong perceptive and strategic capabilities. This approach trains an autonomous agent (DRL ALM agent) that directly learns how to execute ALM through a combination of historical data and outcomes, its own trial and error, and continuous feedback from the environment. To implement this, we defined the

required DRL components: the DRL ALM decision-making agent, agent actions, the environment, states, reward functions, and the deep learning architecture. The results first demonstrate that DRL ALM can achieve duration-matching outcomes within 1% of the theoretical immunisation theory at a 95% confidence level. Second, compared to a benchmark weekly rebalancing traditional ALM regime, high-frequency DRL ALM achieved superior outcomes that were three times less sensitive to interest rate changes. DRL ALM also demonstrated capacity for increased automation, speed, flexibility, and multi-objective optimisation in ALM, thereby further reducing the negative impact of human limitations and improving risk management outcomes. The findings and principles presented in this study apply to various institutional risk-takers, including insurers, banks, pension funds, and asset managers. Overall, applying DRL to ALM provides a promising avenue for improving risk management outcomes compared to traditional approaches.

KEYWORDS
Reinforcement learning, deep reinforcement learning, asset liability management, duration, duration matching, Redington immunisation

CONTACT DETAILS
Mr Takura Asael Wekwete, Johannesburg, South Africa; Email: takurawekwete@gmail.com
Telephone: +27 (0)71 135 4142

KEY CONTRIBUTORS
Prof R Kufakunesu, Dept of Mathematics & Applied Mathematics, University of Pretoria, SA
Dr G van Zyl, Dept of Mathematics & Applied Mathematics, University of Pretoria, SA

1. INTRODUCTION

A thriving and well-managed insurance, banking, and risk management sector is vital for a country's sustainable economic growth. It encourages individuals and businesses to invest, spend, and accumulate wealth with reduced uncertainty about the future (Ward & Zurbruegg, 2000). Asset liability management (ALM), also known as asset liability modelling, is an actuarial and quantitative finance risk management technique used by institutional risk-takers such as insurers, pension funds, banks, and asset managers. A key aim of ALM is to derive an optimal investment asset allocation strategy for reducing interest rate risk exposure by taking into account the corresponding current and future liabilities. The inability to meet claim liabilities is one of the critical risks that institutional risk-takers address with ALM (Smink & van der Meer, 1997). Adequate implementation of ALM is especially critical in environments with rapidly changing interest rates, as has been experienced globally between 2021 and 2023. Poor ALM was a key factor in the demise of several high-profile banks in 2023, such as Silicon Valley Bank (SVB), Credit Suisse, and First Republic Bank (Geman, 2023; Daga, 2023; Barr, 2023).

A common ALM approach is duration matching, in which the asset allocation is chosen such that the timing of the future asset proceeds is aligned as much as possible with that of the expected liability outflows. The duration of a set of cash flows is defined as the average timing of the cash flows weighted by the size of the respective discounted cash flows. The most common theoretical framework used for deriving duration matching is Redington immunisation. One of the key objectives of immunisation is for the optimal asset allocation weights to be chosen such that the duration of the asset portfolio is as close as possible to the duration of liability cash flows (Fooladi & Roberts, 2000). Duration is a standard measure of the interest rate sensitivity of an asset or liability portfolio. Another secondary objective is to have the rate of change of the duration (known as the convexity) for the assets be greater than that of the liabilities where possible (Nieto et al., 2022).

In traditional ALM, the professional (actuary, quantitative analyst, or investment manager) typically initially estimates the duration of the claim liabilities based on projected liability cash flows. For insurance contracts, such analysis and modelling usually incorporate risk factors such as sum insured, insurance term, policyholder risk factors, types of risks covered, etc. Afterwards, the professionals typically select a portfolio of assets whose allocation into bonds, property, cash, and other securities has a duration within an acceptable threshold compared to that of liabilities. Assets that perfectly match the liabilities are not always available (Garrett, 2013; Kahlig, 2022).

Furthermore, this process must be repeated, for example, monthly or quarterly. The process usually involves multiple stakeholders or departments, such as asset managers, treasury, finance, reporting, pricing, product development, valuations, reserving, etc., and usually requires extensive data gathering across various data sources. This often presents a challenge in terms of the speed of the process and, therefore, the frequency at which ALM rebalancing can be done. All the while, the composition of the underlying liability risks is constantly changing on a daily basis (sometimes hourly), especially in insurance, banking, or asset management. Specific theoretical assumptions on the nature of interest rates underpin traditional immunisation (Kahlig, 2022). However, the real-world conditions in which these models are applied often deviate from these assumptions, which requires the professionals to monitor these deviations and make appropriate adjustments in their application based on their judgement. Deep reinforcement learning (DRL) provides an avenue for the ALM to be less reliant on theoretical assumptions (Bühler et al., 2018).

ALM often needs to be achieved simultaneously with other objectives, such as maximising risk-adjusted returns or targeting a given differential between asset duration and liability duration. Consequently, in the traditional approach, some human judgement is commonly applied to the final asset allocation at each iteration in the quest to incorporate the additional objectives as much as possible. The applied human judgement is usually based on the professional's past experiences, corrections for model biases, and balancing the importance of various additional objectives.

Unfortunately, this need for human intervention and judgement in the process

introduces several limitations. The first limitation is that the traditional ALM process becomes difficult to automate. Consequently, there is a limit on the frequency of the updates and asset allocation rebalancing that can be done in a given period. In the time between the duration matching rebalancing iterations, there is also a risk of a rapid change in the interest rate hedge between asset and liability portfolios. Deep reinforcement learning introduces the possibility of minimising the requirement for human intervention (Hariom et al., 2020).

Second, a significant drawback of the conventional ALM approach, Redington immunisation, is its limited capacity for facilitating exploration, experimentation, and error correction of various asset allocations within a comprehensive feedback loop. This is because the number of potential allocations within and across various asset classes is infinite in continuous state spaces. In addition, the investment environment and the composition of the risk-takers' liabilities are continuously changing. Hence, the traditional ALM duration matching approach is often not consistently and sufficiently exploratory of all the options within a learning process. Deep reinforcement learning provides a way for the ALM to be executed in a continuously improving feedback loop (Hariom et al., 2020; Englisch et al., 2023; Krabichler & Teichmann, 2023).

Third, the high reliance on human intervention and judgement exposes the traditional asset liability management investment processes to human behavioural irrationality and limitations. Humans are subject to emotions such as fear and greed, which can negatively lead to suboptimal asset allocation decisions. Humans are also vulnerable to a lack of consistency in applying the organisation's ALM investment policy. Humans are also susceptible to other irrationalities such as confirmation bias, overconfidence, recency bias, availability bias, and many other biases (Syed & Bansal, 2018; Rabbani et al., 2021; Chiu et al., 2022; Bondt et al., 2013).

Fourth, in addition to the primary ALM, the need to weigh and prioritise current objectives in the ALM process can be challenging to implement. This is currently being done in the industry either by a rules-based software approach or by human professional judgement. For example, suppose the ALM is carried out in a software tool such as spreadsheet software. In that case, it is often time-consuming and challenging to explicitly express all the other objectives and constraints. Deep reinforcement learning provides for the possibility to carry out optimisation of multiple objectives simultaneously (Englisch et al., 2023; Krabichler & Teichmann, 2023).

In this paper, we examine the feasibility, performance, and advantages of using DRL to implement ALM in relation to the issues outlined above for traditional ALM. Deep reinforcement learning incorporates deep learning into the reinforcement learning (RL) framework. A key benefit of this approach is that it creates an autonomous ALM decision-making agent that directly learns the objectives of ALM and which actions to take or not take with minimal supervision. Furthermore, this deep reinforcement learning asset liability modelling (DRL ALM) agent not only learns through a combination of historical

data and outcomes but also through its own trial and error and continuous feedback from the environment. DRL ALM presents the promise of introducing an autonomous decision-making agent that can be incorporated into company software systems, data pipelines, and ALM processes for automated and faster implementation compared to traditional approaches. DRL ALM also holds the promise of rational, consistent, and agile ALM implementation that is also able to learn from its own past actions and outcomes. Due to these strong advantages, there have been recent successful implementations by others to manage asset liability management problems with deep learning and RL (Englisch et al., 2023; Krabichler & Teichmann, 2023; Cheridito et al., 2020). Ultimately, the objective is to create a highly capable artificial intelligence (AI) asset liability management assistant that can complement and increase the productivity of professionals in the asset liability management field.

Section 2 formally outlines this paper's problem statement and research questions, namely, whether RL can be applied to manage, fulfil, and enhance ALM relative to the theoretical method of Redington immunisation. Furthermore, we set out to compare the performance of DRL ALM to a practical benchmark traditional ALM approach.

Section 3 discusses and synthesises existing papers and applications of RL in general and, more specifically, within the insurance and quantitative finance domains. We also look at recent related ALM work and outline the existing gaps, which are filled by the output of this paper.

Section 4 outlines and explains the theoretical framework of Redington immunisation which underpins most traditional ALM. This approach is fundamental to most of the ALM execution in the actuarial and quantitative finance fields.

Section 5 introduces and explains the Monte Carlo simulation processes used to generate data that were used to train and evaluate the RL ALM agent.

Section 6 introduces and explains the details of the DRL framework and the training process. We outline the building blocks of both deep learning and RL and how they complement each other well to give DRL ALM solutions with both strong perceptive and strategic capabilities.

In Section 7, we evaluate the performance of the RL ALM and contrast it against the theoretical immunisation and a practical benchmark traditional ALM approach. We conclude in Section 8 by summarising the main findings and future studies.

2. PROBLEM STATEMENT AND RESEARCH QUESTIONS

2.1 Problem statement

Implementing ALM using traditional ALM methods that rely heavily on Redington immunisation and professional judgement expose an organisation to severe limitations and business risks, such as limited automation of the ALM process, exposure to risks of human irrationality and human error, and constrained multi-objective optimisation. The problem statement of this paper is: Can DRL be used to successfully implement and

automate ALM by duration matching to mitigate the problems and effectively address the limitations of the traditional approach?

Reinforcement learning has been successfully applied in some other fields of quantitative finance. We explore whether this application can be extended to ALM primarily in the insurance and banking contexts. Furthermore, we investigate whether DRL can sufficiently and practically improve the critical risk management objectives of a typical institutional risk-taker (such as a life insurer) relative to immunisation.

2.2 Research questions

This paper presents investigations and findings guided by the following research questions:

(a) **Comparison of deep reinforcement learning to theoretical Redington immunisation**

The first research question was to investigate the following: At a given point in time, do the DRL ALM asset allocation results, and by extension, the duration results, give similar outcomes to Redington's immunisation theory ALM? If they differ, to what extent do they differ?

For this research question, we focused on the universe of assets comprised primarily of bonds, specifically zero-coupon bonds, for simplicity. We also investigated the robustness of the DRL solution under stress-testing.

(b) **Comparison of deep reinforcement learning to a traditional benchmark ALM strategy**

The second research question was to investigate the following: How do DRL's asset allocations compare to those of a benchmark practical traditional ALM strategy for hedging a changing liability portfolio? Does it result in better interest rate hedging, and to what extent?

For this research question, we tested and contrasted the daily asset allocation of RL within a 30-day month.

2.3 Research limitations

There are various duration-based asset liability management techniques. This paper focused on benchmarking RL performance against the commonly used traditional ALM approach of duration matching. We also limited our context to investments in long-position investments within traditional hedging asset classes such as bonds, property, and cash or cash-equivalents.

In fact, for the experimentation, we focused only on zero-coupon bonds for simplicity. However, this is not a very restrictive framework because coupon-bearing bonds can be modelled as a series of zero-coupon bonds with smaller face values and different maturities (Jarrow, 2004; Jarrow & Turnbull, 2000). Therefore, this paper's findings are also applicable when hedging is carried out with coupon-bearing bonds.

We did not consider complicated investment strategies, such as those with short positions or derivatives, because the risk-taking institutions we considered in this paper are usually restricted from taking such risky investments.

3. RELATED WORKS

In this section, we consider the existing literature related to this paper. In Section 3.1 we introduce the concept of the DRL which combines deep learning and RL. In Sections 3.2 and 3.3 we explore the wide-ranging applications of DRL. Thereafter, in Section 3.4 we explore applications in the actuarial field and quantitative finance. In Section 3.5 we explore a few of the recent applications of deep learning and RL in ALM, which indicates that this research and application area is still an emerging and novel one.

3.1 Introduction to deep reinforcement learning

The first ingredient of DRL is deep learning. Deep learning is well suited to tasks which require perception of complex patterns and correlations in data. The fundamental building units of deep learning are similar to those used in statistical linear regression for statistical predictions. However, in deep learning these predictive building units (nodes) are assembled into a large and deep network which mimics the structure of a human brain (Dong et al., 2021). Deep learning architectures, such as deep neural networks (DNNs), convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have been shown to significantly outperform classical statistical models (Richman & Wüthrich, 2023). Furthermore, deep learning solutions have been demonstrated to have superior results compared to traditional solutions within many complex applications in many areas including actuarial applications (Richman, 2022; Perla et al., 2021).

The second ingredient of DRL is reinforcement learning. There are three main categories of machine learning, namely supervised learning, unsupervised learning and RL. Reinforcement learning is guided by a specific goal(s) which is optimised for in a dynamic environment by an agent. This agent learns how to optimise for the goal(s) similar to how a child learns through trial and error, being rewarded for good behaviour and punished for undesired actions (Naeem et al., 2020). As such, RL is well suited to tasks requiring strategy and planning with a longer term perspective.

Therefore, the combination of deep learning and RL, which complement each other, gives a DRL solution with both strong perceptive and strategic capabilities. This approach is perfect for the ALM problem where perception of complex relationships and changes in asset and liability data is needed on top of the ability of making long-term strategic decisions.

3.2 Applications of deep reinforcement learning in various domains

Reinforcement learning has successfully been applied in a wide range of domains and applications. Deep reinforcement learning is often applied with the use of deep learning

models, such as artificial neural networks (ANNs), convolutional neural networks (CNNs), and recurrent neural networks (RNNs) (Dong et al., 2021). When deep learning is used in RL, this can be classified as DRL (Mousavi et al., 2018). Deep learning models have also, in their own right, been applied successfully in a wide range of domains, such as computer vision, speech recognition, natural language translation, time series analysis, autonomous driving, medical diagnosis, among many other applications (Dong et al., 2021; He & Droppo, 2016; Hsu et al., 2016; Sak et al., 2014; Qu et al., 2017; Altché and de La Fortelle, 2017).

Figure 1 illustrates a summary of some of the main application areas (Li, 2017; Li, 2022). From Figure 1, we can see the dynamic nature of RL in very diverse fields. In the diagram, ITS represents intelligent transport systems, and NLP represents natural language processing.

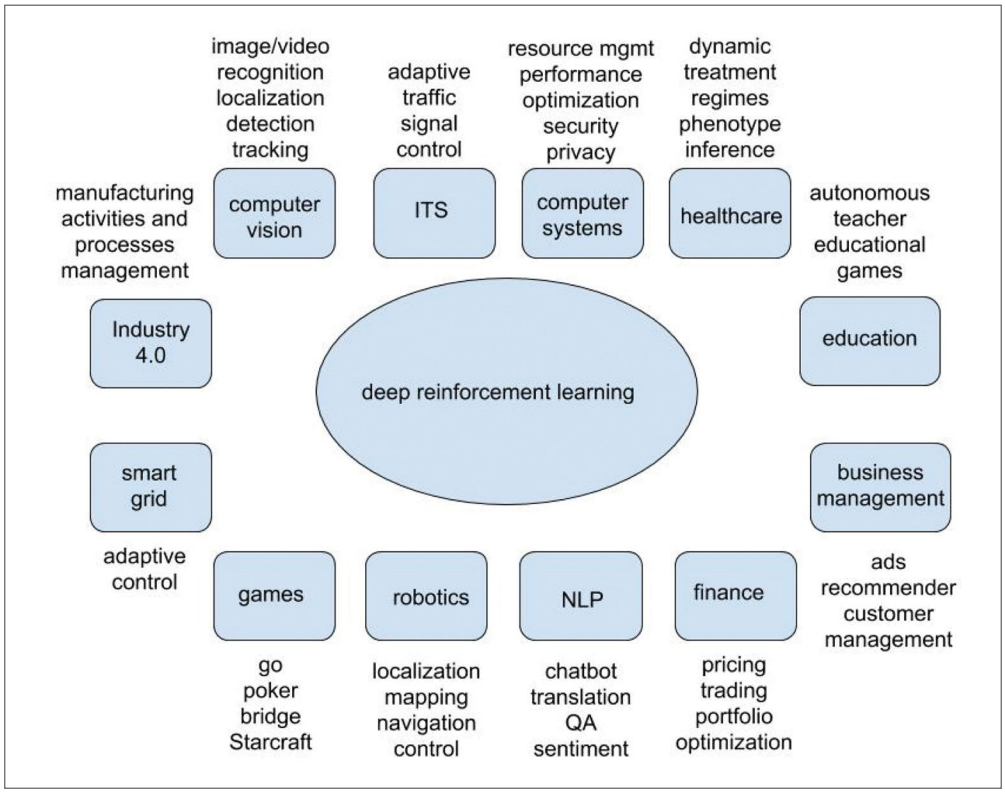

Figure 1 Deep reinforcement learning applications

3.3 Strengths and challenges of reinforcement learning

A key feature of model-free RL is that one does not need to assume a model or impose underlying dynamics of financial relationships beforehand (Sato, 2019). This is incredibly advantageous in applications where there exist uncertainties on the truth of underlying

dynamics or where the dynamics are consistently evolving. For example, this is the case for the investments of an insurer, and this model-free nature of RL gives flexibility and adaptability in such applications.

There has been an extensive exploration of the various questions and challenges encountered in applying model-free RL in financial portfolio optimisation (Sato, 2019). In this literature, popular RL approaches, such as Q-Learning, are explored. Q-learning is a model-free RL approach aiming to maximise the total value of the expected reward or gain defined within the environment.

The performance of RL was noted to have several challenges, unfortunately. First, it is highly dependent on the appropriate specification of the objective value function, which is often challenging to specify while capturing the critical objectives. Second, the solutions can be unstable if there is too much noise in the loss function. Third, RL is also susceptible to other issues such as over-fitting, the curse of high-dimensionality, dependence on many samples, interpretability and credit assignment problems. The credit assignment problem occurs when the consequences of a decision only materialise many iterations after the decision, which makes it difficult to attribute the decision to the outcome (Sato, 2019).

Devraj and Meyn also highlighted that Q-learning RL can sometimes be slow to converge, and alternative variations have been proposed (Devraj & Meyn, 2017). The training and learning process for a RL agent can be computationally intensive as the agent has to be exposed to various scenarios, their outcomes and rewards. The agent itself also requires time to implement its own trial and error and learn about the optimal actions and policy for a given situation.

This research has an advantage in that it prepares one for the theoretical and implementation challenges in the application. Unfortunately, the studies by Sato, Devraj and Meyn referred to above did not focus on demonstrating detailed practical applications of RL, particularly in the insurance context.

3.4 Applications of reinforcement learning in quantitative finance

There has been some recent successful application of RL in other areas of quantitative finance. For example, there have been successful demonstrations of the application of RL to create an automated trading bot which made profitable actions on unseen test data (Hariom et al., 2020). They also developed a derivatives option hedging strategy based on DRL, which performed reasonably well. They also used a RL-based portfolio allocation to maximise risk-adjusted returns on cryptocurrencies.

RL has also been applied in solving dynamic optimisation problems in quantitative finance, such as portfolio allocation, pricing and hedging contingent claims in model-free contexts (Kolm & Ritter, 2020; Dixon & Halperin, 2019). These applications show the flexibility of RL in various quantitative finance contexts. However, these applications were not applied in the context of ALM, which is where the gap in literature and application exists.

In the actuarial field, there has been the development of a Markov decision process model for a life insurer which was then successfully applied to RL algorithms to maximise the risk-adjusted return on capital for the company (Abrate et al., 2021). This literature did incorporate the future liability cash flows of the insurer, but the objective was not to implement ALM based on duration matching. Their research was applied in the context of insurance companies, which is more aligned to the research laid out in this paper. However, their objective was only to maximise risk-adjusted returns without much regard for the liability profile of the insured liabilities. This highlights a gap in applying ALM to decide the strategy to match the liabilities' duration.

Wüthrich and Merz introduced deep learning as one of the important methodologies in the toolkit of actuarial practice and statistical modelling (Wüthrich & Merz, 2023). However, there is no inclusion of RL or application to asset liability modelling.

3.5 Recent applications of deep learning to asset liability modelling

Cheridito et al. (2020) successfully applied a deep learning approach to estimate the tail risk measures of a portfolio of assets and liabilities. The tail risk measures used for assessing asset liability risk included value-at-risk and expected shortfall in order to ensure regulatory compliance with Solvency II (Cheridito et al., 2020). Although there was successful implementation of deep learning in asset liability modelling, the solution is static compared to a situation where RL is used. The solution is not able to continuously learn over time and adapt to changing market and risk conditions as is enabled by RL.

Krabichler and Teichmann (2023) recently successfully applied deep learning and RL to develop a methodology for hedging interest rate risk of asset and liability portfolios for retail banks. They applied the method of deep hedging to a situation of a runoff (no going concern) portfolio of retail banking liabilities. The deep learning ALM application demonstrated an out-performance of 2% per annum on a return on equity basis. The Tensorflow library was applied in the deep learning implementation (Krabichler & Teichmann, 2023).

Englisch et al. (2023) used a similar approach of deep hedging also in a similar retail banking context but compared the application to more benchmark strategies in a comprehensive manner. In addition, they also explicitly modelled yield curves in their scenario generation. They also introduced regulatory constraints such as liquidity coverage ratio to the problem. The results showed that deep learning-based ALM outperformed all the benchmarks whilst staying within regulatory limits in almost all yield curve scenarios. However, several challenges were noted, such as the difficulty of model explainability of the dynamic decision-making agent and the difficulty of appropriately defining loss functions that are preferences of an ALM strategy. There was also an extended application to unwinding swap portfolios, however, with some mixed results (Englisch et al., 2023).

Fontoura et al. (2019) applied DRL to a relatively simple ALM problem which optimised for the ratio of asset values to liability values. Although this application was successful, it

was applied to a simpler problem where the institution only aimed to have asset values being greater than liabilities regardless of the duration mismatch and interest rate hedging. This solution leaves institutions exposed to adverse interest rate movements, which are the biggest risk factor for long-term liabilities in most cases. Therefore, in practice, comprehensive ALM generally requires one to also hedge for interest rate movements through managing duration mismatch. Furthermore, another limitation of Fontoura et al.'s solution is that it was only applicable to deterministic liability movements, which is rarely the case in reality (Fontoura et al., 2019).

3.6 Research gaps addressed

From the literature discussed it is evident that RL has been applied with some success in quantitative finance and actuarial science (Hariom et al., 2020; Kolm & Ritter, 2020; Dixon & Halperin, 2019; Abrate et al., 2021). Although the recent application of deep learning and RL for asset liability modelling was successful in banking it was based on approaches customised specifically for retail banking (Krabichler & Teichmann, 2023; Englisch et al., 2023).

There is a gap in that there has not been a general purpose DRL application to ALM which takes into account interest rate hedging for stochastic liabilities. That is, an application that can replicate and improve on the well-established approach of Redington's immunisation theory applicable in all actuarial and quantitative finance spheres.

ALM implementation is more complicated compared to other quantitative finance applications, such as return maximisation, because the observed data for which optimisation is sought is always multi-dimensional. Furthermore, ALM requires one to consider patterns in both assets and liability data for a relatively complicated objective compared to the other problems. Duration matching assets to liabilities is a slightly more complex objective than maximising return, for example.

4. FOUNDATIONS OF TRADITIONAL ASSET LIABILITY MANAGEMENT

4.1 Traditional immunisation theoretical framework

The traditional ALM approaches rely heavily on Redington's immunisation theory which is based on satisfying Redington's three conditions (Redington, 1952). The first condition requires that the present value of the assets (A) be equal to that of the liabilities (L), that is:

$$A = L\ , \tag{1}$$

where $A = \int_0^\infty A_t e^{-rt} dt$ and $L = \int_0^\infty L_t e^{-rt} dt$. A_t and L_t are the asset cash flow and liability cash flow at time t, respectively. r represents the constant continuously compounded discounted rate.

The second condition requires that the first derivatives of the asset and liability values with respect to the discount rate (r) must be equal:

$$\frac{\partial A}{\partial r} = \frac{\partial L}{\partial r}. \tag{2}$$

The above means that the asset sensitivity (asset duration) to interest rates must be the same as the liability value sensitivity (liability duration). This duration is also known as the Macaulay duration.

The third condition requires that the second derivative of the asset value must be greater than or equal to that of the liability value:

$$\frac{\partial^2 A}{\partial r^2} \geq \frac{\partial^2 L}{\partial r^2}. \tag{3}$$

The above means that the rate of change of the asset duration, that is, the asset convexity, is greater than the rate of change of the liability duration (liability convexity).

Redington's immunisation theory is underpinned by a few theoretical assumptions. It assumes a flat term structure of interest rates, that is, the spot interest rates s_t are equal for all times t. Of course, these assumptions are often not the case in reality but are a helpful simplification for developing a theoretical framework. The framework also assumes that when interest rates change, they change by the same amount at all future time points; that is, the interest rate changes are parallel shifts down or up (Kahlig, 2022).

In deriving the solutions that satisfy Redington immunisation, one usually solves for the asset allocation that satisfies the first two conditions by solving for the arising simultaneous equations. From the set of possible solutions, one then verifies or selects (if there is more than one potential solution) the one that satisfies the third condition by plugging in the solution in the second derivative (Garrett, 2013).

If the second and third conditions are both satisfied, then we would have perfect matching, and we know that either the present value of assets (A) will increase by more than that of liabilities (L) if the interest rate decreases; or the present value of assets (A) will decrease by less than that of liabilities (L) if the interest rate increases. In practice, it is not always the case that one can satisfy or prove Condition 3. This would be imperfect interest rate hedging. However, even if only Conditions 1 and 2 are met, the resulting asset value movements would move more or less in line with liabilities; this would still be much better interest rate protection than a situation where there is none at all. Therefore, we will focus on explicitly deriving for Conditions 1 and 2 in Section 4.2.

Due to the underlying assumptions of Redington's immunisation outlined above, there are several challenges and limitations to implementing it in the real world (Kahlig, 2022). These include:

— Interest rates are not flat in most real-world cases. Since the underlying model assumes a constant interest rate, one must carefully choose an appropriate constant interest rate for all calculations.
— Immunisation only provides hedging against small changes in the interest rate. If changes are large, the hedge is imperfect.

— Immunisation requires frequent rebalancing of the asset weights to keep the duration of the assets the same (or close enough) to that of the liabilities. This is especially challenging if the market conditions do not easily correspond to the underlying model assumptions and significant judgement is required in the rebalancing.
— It is not always possible to know the exact timing of asset and liability cash flows, as is assumed by the immunisation theory. In our problem, we assume these are known for simplification but, in reality, one may need to apply estimations or probability distribution assumptions.
— Assets of the required maturity to achieve immunisation may not exist in the market. A solution for this is not provided in the vanilla approach.

These limitations require that, when implemented in practice, the user of the model constantly monitors the results and conditions (e.g., the market yield curve, market volatility, etc.). If the requirements differ significantly from the underlying assumptions, then the user should make adjustments based on judgement. This is one of the significant limitations of the traditional approach in that it requires constant human supervision and applying adjustments to the raw results.

4.2 Traditional asset liability management practical implementation

Immunisation sets out two main objectives for duration matching. The first objective is to ensure that the present value of the asset portfolio is equal to that of the liabilities. The second objective is to ensure that the Macaulay duration (simply referred to as duration in the rest of the paper) of the assets portfolio is equal to that of the liabilities (Weil, 1973).

Based on Equations 1 and 2, we get the system of equations with two equations:
(1) *PresentValue (Assets) = PresentValue(Liabilities)*; and
(2) *Duration(Assets) = Duration(Liabilities)*.

In the scenario with two zero-coupon bonds, Z_1 and Z_2 the above system of equations translates to the following:

$$\begin{aligned}&(i)\ \ \omega_1 PV(Z_1)+\omega_2 PV(Z_2)=PV(Liability)\\&(ii)\ \ \omega_1 Duration(Z_1)+\omega_2 Duration(Z_2)=Duration(Liability)\end{aligned}$$

The duration of the asset portfolio is the weighted duration of the respective assets. ω_1 and ω_2 are the respective weights which add to 1.

For the two asset problem, the system of equations above can be expressed in matrix formulation shown by Equation 4:

$$\begin{bmatrix}1 & 1\\ T(Z_1) & T(Z_2)\end{bmatrix}\begin{bmatrix}\widehat{\omega_1}\\ \widehat{\omega_2}\end{bmatrix}=\begin{bmatrix}1\\ T(L)\end{bmatrix}. \tag{4}$$

The solution is then determined by taking the inverse of both sides of the equation as follows:

$$W^* := \begin{bmatrix} \widehat{\omega_1} \\ \widehat{\omega_2} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ T(Z_1) & T(Z_2) \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ T(L) \end{bmatrix}, \tag{5}$$

where $T(Z_1)$, $T(Z_1)$ and $T(L)$ represent the duration of Bond 1, Bond 2, and liabilities, respectively.

From the above, we can see that the allocation weights are a function of the asset terms and the liability duration, which can be written as follows:

$$W^* = f\left(T(Z_1), T(Z_2), T(L)\right), \tag{6}$$

where f is a function where $f : \mathbb{R}^3 \rightarrow \mathbb{R}^2$ and it represents an operation in the Python program which executes the operations in Equation 5. The function takes in a 3-dimensional input (liability duration and two bond terms) and gives a 2-dimensional output (the weights for the two bonds for duration matching). The above can alternatively be represented by Equation 7 in generalised matrix form as:

$$W^* = M^{-1}E\,, \tag{7}$$

where:

— W^* represents the traditional ALM allocation weights for duration matching,
— M is a matrix with the second row capturing the different term-to-maturities of the respective zero-coupon bonds at the given time, and
— E is a matrix with the expected liability cash flow timing (liability duration).

Since one needs to calculate the inverse of M, this approach requires that the matrix be invertible. There are several mathematically equivalent ways to describe invertibility, such as any one of:

— the determinant of matrix M is non-zero,
— the columns of matrix M are linearly independent, and
— the rank of the matrix is n, where matrix M has dimensions $n \times n$.

In particular practical circumstances, it is possible that the data inputs may not always meet the invertibility requirement, which presents challenges to using this traditional approach to ALM. Other challenges and limitations of the conventional approach were provided in Section 4.1.

In a Python program, the solution for W^* (Equation 7) for the traditional approach was executed using the numpy library's linear algebra solvers with the `numpy.linalg.solve(M,E)` command.

5. MONTE CARLO SIMULATION OF ASSET LIABILITY MANAGEMENT ENVIRONMENT

5.1 Monte Carlo data generation

To generate some training data for the RL agent, we simulated 10 000 Monte Carlo scenarios, also known as economic scenarios. We used 10 000 simulations because they are sufficient to be associated with acceptable changes in errors of resultant estimates (1% or less). Furthermore, this number of simulations is the level at which the RL agent would be sufficiently trained.

In each scenario, we assumed the data to be for five years or 60 months, that is, 60 time points. In each of the 10 000 scenarios, there was a liability duration value and bond term-to-maturity simulation value at all times. We simulated the data which can arise from the environment, such as that of an insurance company or bank. In these contexts, there is some randomness in the change of the liability duration from one-time step to the next.

One of the critical advantages of RL is that one can train the agent in an environment based on simulated or synthetic data and still achieve successful real-world applications. This approach means that RL can be applied in a manner that saves cost and time by not requiring development and training on real-world data. Many successful applications of RL for even tricky tasks, such as in autonomous driving, are trained primarily from a simulated environment (Osiński et al., 2020).

5.2 Liability duration simulation

For a risk-taker such as an insurance company or pension fund, the liability duration increases when the institution covers a new risk which is expected to claim later than the current risk pool's weighted liability duration. The opposite is also true. Alternatively, the liability duration will reduce if shorter duration insurance policies lapse from the insurance risk pool.

The movements in liability duration from one point to the next are generally random because there is randomness in the nature of the trends due to new business, lapses (terminations or expiry) and the claims that the risk taker experiences between time points. The process for generating the *ith* simulation's liability duration and reflecting its randomness is given in Equation 8:

$$D_{it} = D_{i,t-1} + \Gamma_{it} \times \delta_{it} \ , \tag{8}$$

where:

— D_{it} is the liability duration for the *ith* simulation at time *t*. Here, $i \in \{1,2,..,N\}$ and $t \in \{1,2,..,T\}$; where the number of simulations (N) is 10 000 and the number of time steps is (T) is 60 months (5 years).

— $D_{i,t-1}$ is the liability duration for at the previous time point $t-1$.

— Γ_{it} is a binomial distributed variable with two possible values +1 or –1, with probabilities P_t and $1-P_t$, respectively.
— P_t is randomly sampled from a uniform distribution with a range of [0,1], that is, $P_t \sim U(0,1)$. Therefore, determines the probability by which the liability duration increases or decreases.
— δ_{it} is the magnitude of the absolute movement in the liability duration, which is randomly sampled from a uniform distribution with a range of $[0;\Delta]$, that is, $\delta_{it} \sim U(0,\Delta)$.
— Δ is the maximum possible change from one-time step to the next and is set at 0.5 years in the training data. Therefore, for this training data simulation, we assumed the liability duration could change by an absolute value of up to 0.5 years from one step to the next.

5.3 Bond term-to-maturity simulation

We assumed that the liabilities would be matched primarily by zero-coupon bonds. In the simplified version of our problem, two bonds are available in the market: (1) a short-dated zero-coupon bond and (2) a long-dated zero-coupon bond. For each simulation *i* we randomly sampled the term of the short-dated bond from a range of uniform distributions with a range between 10 and 20 years. For the long-dated bond from a range of uniform distribution with a range between 30 to 40 years. The bonds' terms are defined as:

— $T(Z_1)_i$ for first zero-coupon bond term for *ith* simulation. Sampled from a uniform distribution of parameters (10 years, 20 years), that is, $T(Z_1)_i \sim U(10,20)$.
— $T(Z_2)_i$ for second zero-coupon bond term for *ith* simulation. Sampled from a uniform distribution of parameters (30 years, 40 years), that is, $T(Z_2)_i \sim U(30,40)$.

For a given simulation, once sampled, the term to maturity of the short and long bonds was assumed to remain the same at all time points. We assume that the risk-taker can find zero-coupon bonds of simulations' given maturities in the market at all times throughout that given simulation. In this research question, we are primarily interested in checking the solution bond allocations as liability duration changes. We are not mainly interested in the bonds' term-to-maturity variability but in the liability duration and its impact on the asset allocation for duration matching.

We assumed two bonds because a multiple (*P*) bond allocation task can be represented as a two-bond allocation task by grouping all the other *P*–1 as one notional combined bond. Hence, this conceptual bond would be a weighted term of these *P*–1 bonds. To test the viability of RL to ALM, it is sufficient to start with a two-bond scenario to assess viability.

5.4 Visualisation of Monte Carlo simulations

A visualisation example showing a subset of typical simulation results from scenario generators in Sections 5.2 and 5.3 are shown in Figure 2.

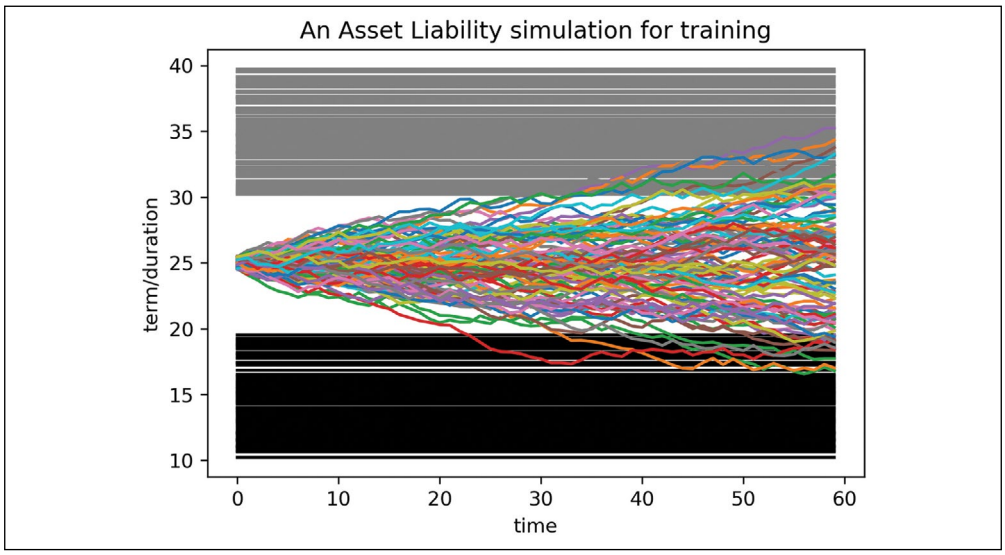

FIGURE 2 An example of a set of Monte Carlo simulations

The lines in colour represent pathways of the evolution of liability over time generated from Section 5.2 with each pathway representing one scenario. There are many scenarios, each with a different pattern.

The horizontal lines in colour represent the maturity terms of different zero-coupon bonds over time generated from Section 5.3 with each line representing one scenario. The grey lines represent the long-duration bonds and the black lines represent short-duration bonds. For a given scenario, a pair of lines is generated, one grey and one black.

6. REINFORCEMENT LEARNING ASSET LIABILITY MANAGEMENT

6.1 Deep reinforcement learning framework components

In this section, we discuss the key components of the RL model, namely, the environment, states, actions, agent and reward function. These components are necessary to define a RL model to be successful (Hariom et al., 2020).

These components allow for the essence of RL, which is to learn through interactions (Arulkumaran et al., 2017). Hence, these components are interlinked and related.

6.1.1 REINFORCEMENT LEARNING ENVIRONMENT

The first component to consider is the environment. The environment represents the operating environment where a risk-taking institution is exposed to both assets and liabilities, such as an insurance company, pension fund, bank, or asset manager. For simplicity, one can regard the standard environment to be that of an insurance company that writes insurance contracts which pay out benefits in event of death, disability, medical or critical illness.

6.1.2 REINFORCEMENT LEARNING STATES

The second component to consider are the states (S) of the process, which are the key metrics which can be observed within the environment by the agent (explained in Section 6.1.4) and used for decision-making. At any given time, the state is the current term to maturities for the securities and also the liability duration. In the two zero-coupon bond situations we have, this means we can observe the two asset terms ($T(Z_1)_t$ and $T(Z_2)_t$) and the liability duration (D_t). This corresponds to a 3-dimensional state problem, which requires additional considerations compared to a 1-dimensional state problem. The history of liability duration is also observable and is an essential factor for solving the problem; hence, this is also an additional state.

6.1.3 REINFORCEMENT LEARNING ACTIONS

The third component to be considered for RL is the actions which can be taken. This represents the actions the agent (explained in Section 6.1.4) can take at time t, generally represented as a_t. In this context, the actions are the weight allocations to respective zero-coupon bonds for duration matching. Possible actions are the weights of security i's allocation in the assets portfolio ω_i, for $i \in \{1, 2, .., P\}$ where there are P assets. For the two asset problem we have ω_1 and ω_2, that is, $P = 2$.

We are not considering short-selling; therefore, for a successful solution, we will require the weights to range between zero (0) and one (1). Therefore, these actions are continuous in nature and not discrete. We are only considering long positions because regulations generally bar the institutions such as insurance companies from taking short positions. Furthermore, for ALM, short positions are usually unsuitable for that risk-management objective.

6.1.4 REINFORCEMENT LEARNING AGENT

The agent is the autonomous entity which performs the actions and acts within the defined environment. In addition, the agent observes the outcomes of specific actions through feedback from the environment – the input is in the form of a reward function (explained in Section 6.1.5). The agent's objective is to learn the behaviour or actions a_t which leads to desired outcomes for a given state S. This optimal behaviour is known as the policy π (Arulkumaran et al., 2017). The agent learns the policy through a combination of trial and error and by observing the historical correlations and patterns between states, actions and rewards.

The policy π is a mapping from the states to actions. However, due to the inherent uncertainty, there is a probability estimation of the best actions. Therefore, the policy maps states (S) to probability distribution over all possible actions (A) (Arulkumaran et al., 2017). This is expressed in Equation 9:

$$\pi : S \rightarrow p(A = a \mid S) \,. \tag{9}$$

The agent will take action with the highest probability at a given time (Arulkumaran et al., 2017). Often the agent relies on machine learning, as is the case in this paper, to decipher the correlations and complex patterns between states, actions and rewards. This is the case in this paper, where the agent uses deep learning to determine the policy. Hence, this problem can be called DRL as we combine RL and deep learning. More on the deep learning techniques used is given in Section 6.2.

6.1.5 REINFORCEMENT LEARNING REWARD FUNCTION

A very important component of RL is the reward function. The reward function is the feedback sent by the environment to evaluate the last action by the agent. An appropriate reward might be the inverse of the absolute difference between the asset duration and the liability duration.

The objective of RL optimisation is to determine the best agent's policy to maximise the reward function. This is equivalent to minimising a penalty function or an error function. The mismatch error between the asset duration and liability duration for a given scenario i and at a given time point t for this research question is given by Equation 10:

$$e_{it} = \omega_{1it} T(Z_1)_{it} + \omega_{2it} T(Z_2)_{it} - D_{it} \ . \tag{10}$$

For a given scenario i, we square and sum the duration mismatch errors at all time points to get the scenario error as represented by Equation 11:

$$Simulation\ Sum\ Square\ Errors\ \left(SSE_i\right) = \sum_{t=1}^{T} e_{it}^2 \, . \tag{11}$$

We square the errors because, in the standard duration matching problem, we want to minimise the duration absolute difference without regard to whether the asset duration is smaller or not. The $\left(SSE_i\right)$ function will be the feedback to the agent on how well its current learnt policy, from a combination of past learning and trial and error, performed on the scenario. The agent will implement more policy elements that reduce the error and fewer of those that increase the error. Hence, there is a reinforcement of appropriate policy elements.

In practice, the policy is not updated after every scenario but after a set of scenarios called a batch. This is because one scenario will not be sufficient grounds to update a policy because of limited data. If there are N simulations and each batch is of size B scenarios, there will be N/B batches. The batch SSE will be used to adjust the agent's policy iteratively and this is given by Equation 12:

$$Batch\ Sum\ Square\ Errors\ (SSE) = \sum_{i \in Batch} \sum_{t=1}^{T} e_{it}^2 \, . \tag{12}$$

The range of scenarios in the first batch will be from 1 to B; for the second batch, they will be B+1 to 2B; for the third batch will be from 2B+1 to 3B, and so forth. The order in which the batches are shown to the agent should not have any interdependence. If each scenario is independent of the next, as is the case in this data, then this condition is automatically met.

The expectation is that during the agent's learning or training phase, the batch SSE will reduce over time, signalling an improvement in the agent's ability to implement the ALM. The agent learning process is explained further in Section 6.2.3.

6.2 Reinforcement learning asset liability management implementation

This section explains how we practically implemented the DRL.

In our problem, we apply a type of RL called Policy Search because it was the most appropriate (Sigaud & Stulp, 2019). In our situation, the actions are continuous as they are asset allocation weights. Furthermore, it is difficult to pre-determine the values (rewards) of specific actions for a given state because of the continuous nature of the activities, the vastness of the state space, and the nature of the problem. In such a problem, it is best that the autonomous agent finds its optimal policy directly by exploring various behaviours and reinforcing those that perform well concerning the reward function (Sigaud & Stulp, 2019).

6.2.1 AGENT SPECIFICATION IN OBJECT-ORIENTATED PROGRAMMING IN PYTHON PROGRAMMING

As discussed in Section 6.1.4 the agent represents the entity that performs the actions within the environment, that is, the entity which makes the asset allocations for ALM. We set up the agent using object-oriented-programming (OOP) or class-based programming in Python. First, we define an agent class which is then used to create OOP objects which have certain attributes, modules and functions.

In our implementation of the deep learning capabilities we chose to use the TensorFlow library within the OOP implementation. TensorFlow is a powerful Open-Source library developed by Google which makes the development and deployment of machine learning models more accessible, faster and scalable (Tensorflow.org, 2022). It is especially advantageous to use TensorFlow in deep learning because deep learning models are computationally intensive. TensorFlow's efficiencies derive from its graph database structure which is a network of interconnected computational nodes (tensors). Tensors are multi-dimensional arrays which are used to represent the data, its subsequent transformations and computational results (Lang, 2022). How the data is represented differs from the standard relational format of data.

The agent class has various capabilities arising from its modules. The agent class has the following modules:

— The constructor function which defines an `_init_` function, which in turn defines the RL model parameters such as the time steps, batch size (B), number

of environment features and some parameters for the deep learning model. The constructor defines a computational graph in TensorFlow, which captures the logic of the data flow, the reward function (as specified in Equation 12) and also the deep learning model (as specified in Section 6.2.2).

— The batch trainer module defines a key function which trains the agent object. This function goes through one pass of the data (one epoch) and trains the deep learning network based (explained in Section 6.2.2) on practical experience from trial and error and currently learnt policy. The training is, however, done in batches of the data.

— The training module defines a function which calls the batch trainer module function over many passes (epochs) of the data in the training process.

— The predict module uses the trained model object to predict a given set of data or states. This function returns the action (asset allocation weights) for a given state (asset terms and liability duration)

— The restore module restores the saved trained RL models, which can be used for predictions or even additional training.

Another advantage of TensorFlow is that its computational graph structure is more efficient when the data and computations increase because they inherently store the relationships in the data together with the data itself (Pang et al., 2020). In using TensorFlow, one needs to not only learn the TensorFlow syntax but also clearly define all the computational relationships in the process. TensorFlow also requires one to specify the tensor dimensions accurately, and to also specify data types and formats in a precise and consistent manner when defining the graph relationships. If this is done correctly, TensorFlow achieves efficiency from its ability to execute operations in low-level and efficient C++ code on central processing units (CPUs), more powerful graphics processing units (GPUs) or tensor processing units (TPUs). When implemented in a Python program, TensorFlow is used in conjunction with the `numpy` and `keras` libraries (Geron, 2019).

6.2.2 DEEP LEARNING WITH RECURRENT NEURAL NETWORKS

In the direct Policy Search approach, the agent uses an artificial neural network (ANN) to map the state to action. A neural network provides the agent with a mechanism to pick out complex relationships within the mapping from state to action which improves the reward (Arulkumaran et al., 2017). A typical simple vanilla FeedForward ANN is shown in Figure 3 (Hua et al., 2019).

In the figure, information and processing flow in one direction from inputs, through the layers and to output; hence, they are referred to as *FeedForward*. Such a traditional ANN assumes that the next information and outputs are independent of the previous. However, in this problem, the liability duration at time t has some correlation to the level and trend of the liability duration at times before t. There are also correlations in the actions over

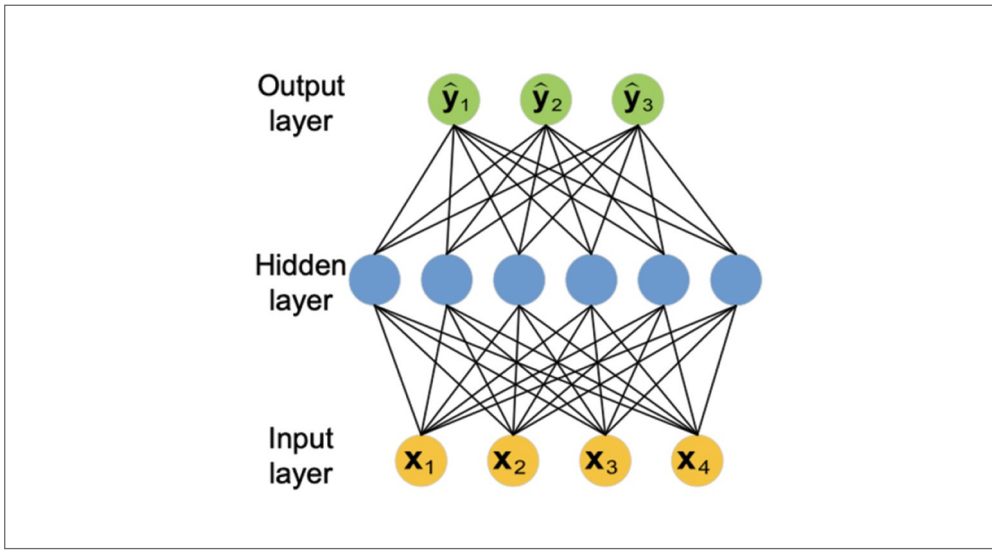

FIGURE 3 An illustration of a simple artificial neural network

time because of the correlation in the states. Moreover, there is path dependence and time-series properties in the states and actions. Therefore, a traditional ANN is not best suited for this problem (Hariom et al., 2020). In the experimentation we carried out in the paper we confirmed that use of a general ANN yielded poor results, which confirmed the need for an improved learning method.

A recurrent neural network (RNN) is a special type of ANN that can capture time-dependent relationships in the states and actions. Therefore, an RNN can factor in the historical information and outputs for the following outputs and is more suited for this problem. An RNN has a time dimension to it where it uses its previous output as input in its current calculation. It also has a dynamic internal state which persists throughout time and uses that to determine the time-dependent predictions (Staudemeyer & Morris, 2019). A typical simple RNN is shown in Figure 4.

The right-hand side of Figure 4 shows the RNN in expanded format. This shows the RNNs calculate the current output ($\hat{y}_t$) from the existing data (x_t) and the hidden state (output) of the previous iteration (h_{t-1}) (Hua et al., 2019).

Standard RNNs have limitations in that they cannot detect relationships of more than ten time steps apart (Staudemeyer & Morris, 2019). Long-Short-Term-Memory Recurrent Neural Networks (LSTM-RNN) are a special type of RNN which can learn longer-term time-dependent relationships (Smagulova & James, 2019). LSTM-RNN can pick up time-dependent relationships of up to 1 000 time steps apart and hence are much more dynamic (Staudemeyer & Morris, 2019). LSTM-RNNs have been successful in a wide range of domains which have temporal or sequential data, such as natural language processing (NLP) (He & Droppo, 2016; Hsu et al., 2016; Sak et al., 2014; Qu et al., 2017), autonomous

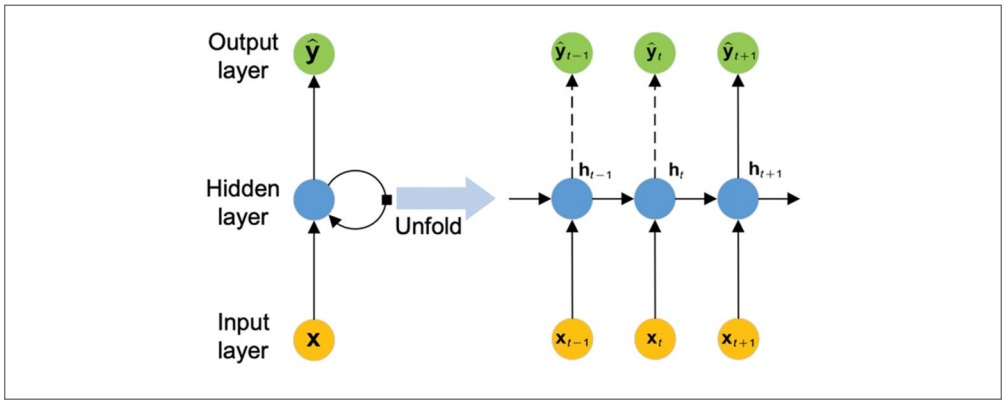

FIGURE 4 An illustration of a recurrent neural network

driving (Altché & de La Fortelle, 2017) and time-series analysis techniques (Mallinar & Rosset, 2018). LSTM-RNN networks are enhanced by the incorporation of memory blocks (cells) with the ability to better persist relevant information for the process. There have been many versions of the LSTM-RNN since the first version which was introduced by Hochreiter and Schmidhuber (1997). However, a typical memory block (cell) of an LSTM-RNN in shown in Figure 5.

This memory is also recurrent (as it is still an RNN) – the outputs (c_t and h_t) are used in the next iteration at $t+1$ and so forth. The memory block achieves its temporal memory capabilities from the three main modules and gates within it (Hua et al., 2019): (1) The Forget Gate control how much of the information passed from the previous cell is irrelevant and discarded in the current cell, (2) The Input Gate controls how much new information which was not present in the previous cell is captured in the current cell, and (3) The Output Gate controls which of the information in the cell is used to determine the block's output, which is then passed to the following stages of the RNN.

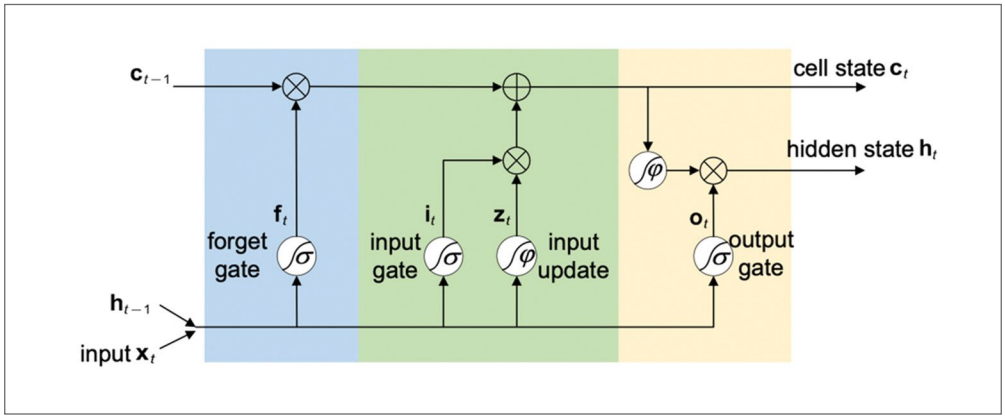

FIGURE 5 An illustration of a long-short term memory RNN cell

We also use an LSTM-RNN with multiple cells and layers, and hence, we categorise this as a deep neural network. In this situation, we used an LSTM-RNN with three hidden layers. The first hidden layer had 62 neurons. The second and third hidden layers have 46 neurons each. The combination of deep learning with RL makes this application a DRL application. The training of the agent is done in batches of the training data. Within each batch of simulations, the agent applies a combination of exploration and past learnings to output weights and their respective rewards. At the end of each batch, the relationships between the states and rewards are persisted and fed through the RNN, which finds the relationship between the state profiles and the rewards by optimising for its own weights and biases. Gradient descent optimisation methods are used to train and determine the optimal weights and biases of the RNN. In order to find the optimal weights for any neural network in general, the gradient descent algorithm (or similar) is used to update the weights iteratively. This method updates the weights in the direction which results in the most significant reduction in the value of the error function (or surface) towards the global minimum (Staudemeyer & Morris, 2019). This involves determining partial derivatives of the weights (parameters) of individual neurons or cells concerning the respective weights (parameters). The impact of some of these is indirect in cases where there are multiple layers, and this requires the chain rule to determine the impact of weights which are more removed from the output. This approach is commonly known as back propagation.

Optimisation of LSTM-RNN follows the same philosophy but tackles the time dimension challenge, which is not present in other neural networks. Here the method used is called back propagation through time (BPTT) (Sherstinsky, 2020). If R represents the reward function (or objective function more generally) and Θ represents all the parameters of the LSTM-RNN, then the approaches rely on the calculations of the partial derivatives $\frac{\partial R}{\partial \Theta}$.

For many parameters, the chain rule will be repeatedly applied to determine the partial derivatives, as they will not have a direct link to the objective function. This approach requires that the activation functions (functions which compute outputs) used at each neuron or cell be differentiable or the derivative be estimable with numerical methods (Sherstinsky, 2020; Staudemeyer & Morris, 2019). After each update, the LSTM-RNN is used to predict the output. The differences between the predicted output from the actual are the errors which are used to carry out another round of updates to the parameters . This process is repeated for a specified number of times or, alternatively, until an acceptable level of error is achieved. This iterative process and deep learning architecture is illustrated in a simple manner in Figure 6.

Within our DRL application, the training data set is divided into batches of 1 000 simulations in each batch. So if there are 10 000 simulations in total, there would be ten batches, with each batch having 1 000 simulations. The reason is that we want to update the agent's learnt policy after having a sufficient level of information (the 1 000 simulations per batch). We also want to allow the agent to apply the learnt policy to a new set of simulations

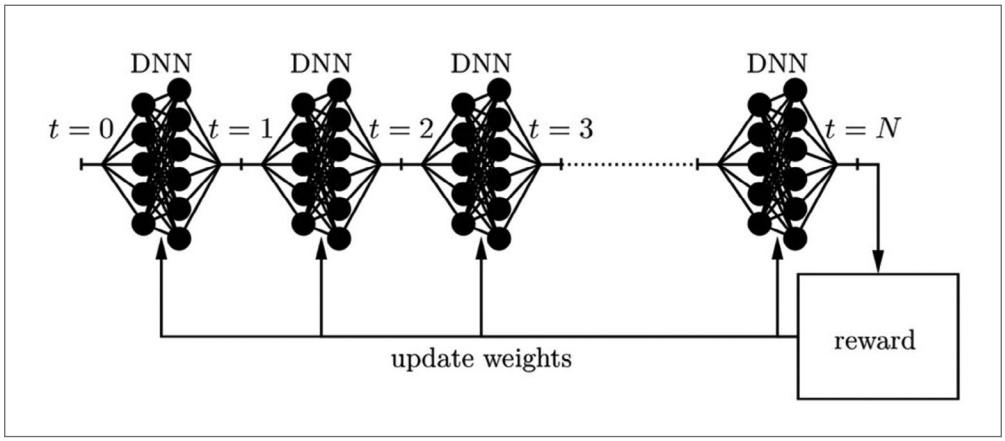

Figure 6 A simple representation of RNN and RL architecture

and verify and improve its policy. Therefore, the LSTM-RNN weights and biases are updated, and this represents the updating of the policy π of the RL agent.

6.2.3 THE REINFORCEMENT AGENT TRAINING STEPS

The main steps at a high level are outlined in Algorithm 1.

Algorithm 1 Main steps for implementation of Deep Reinforcement Learning for Asset Liability Management

1. Create the agent class (see section 6.2.1), with its methods including the batch trainer module, training function, prediction function and the restore function
2. Use the agent class to create an agent object (which includes the creation of a computational graph for use in TensorFlow)
3. Launch a TensorFlow session using `tf.Session`
4. Specify the parameters such as batch size (100), number of features (3), number of training epochs (100) also the name of the model for records
5. Call the training function which triggers the batch trainer module
6. **repeat**
7. **for all** batches in each epoch do
8. Determine the indices of the data that corresponds to that batch
9. Run the TensorFlow computational graph (from Step 2) on the batch data simulations
10. Let the agent apply existing policy and exploration on the batch
11. Record the states, actions (asset allocations) and rewards for that batch
12. Update the weights of the LSTM-RNN based on gradient descent (as in Section 6.2.2)
13. Update the agent's policy π in line with LSTM-RNN update
14. Update the policy which will be applied in the next batch
15. Print the time taken for the batch processing and print the batch SSE
16. **end for**
17. **until** all epochs are complete
18. Save the trained agent object

The algorithm summarises many of the granular and intricate steps which are taken in implementing DRL for ALM. Epochs refer to the number of times the entire dataset of 10 000 simulations is cycled through during training.

We trained the RL agent on the training data simulations. The training process takes about one hour in total on a dataset of 10 000 simulations, with batches of 1 000 and 100 epochs. The trained agent object was saved in memory for retrieval in the testing stages using the restore function of the agent class.

The training process progress was captured by the evolution of the batch sum square error (SSE) in Figure 7.

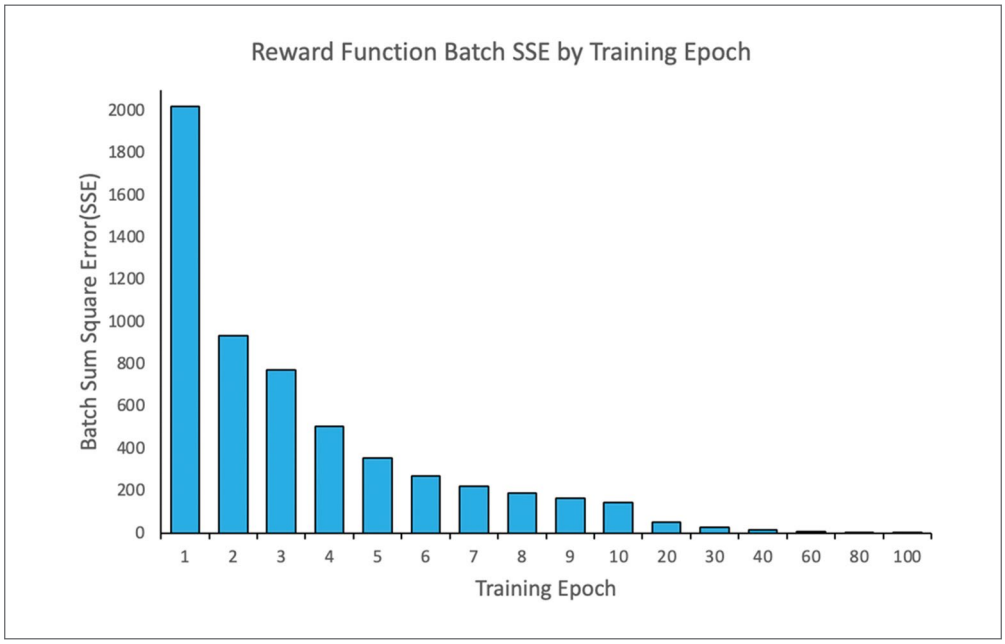

FIGURE 7 Reward function batch SSE by training epoch

We can see the exponential reduction in the batch mean SSE over time as the agent learns the optimal policy for the given data. After each iteration, the agent reinforces its behaviour or policy π towards actions that are more correlated with minimising the SSE. By the 100th epoch, the agent's asset allocations manage to result in asset allocations that give an asset duration very close to the liability duration in the training data.

For proper testing, however, we will analyse the performance of the DRL agent on unseen test data in the next sections. In the next section, we will also compare the results of the RL agent to that of the traditional ALM.

7. RESULTS – EVALUATION AND COMPARISON

7.1 Comparison of deep reinforcement learning ALM to theoretical Redington immunisation

In this section, we compare the performance results of the RL ALM and contrast it against theoretical immunisation. In Section 7.1.1 and Section 7.1.2 we first evaluate the primary research question, which was: at a given point in time, do the DRL ALM asset allocation results, and by extension, the duration results, give similar outcomes to Redington's immunisation theory ALM? If they differ, to what extent do they differ?

In Section 7.1.3 we also evaluated the performance of the RL approach under stress conditions with much higher volatility than trained on. In this section, we discuss some of the interesting key findings from the stress testing, which we implemented on the RL ALM agent. The purpose of the stress testing was to do further experimentation on the flexibility of RL to different operating conditions.

7.1.1 SAMPLE CASE EVALUATION

We generated 1 000 test scenarios using the same approach and parameters used to create the training data, that is, the simulation processes outlined in Section 5.1. We used this as our standard testing data (Test Data 1). An example of a path from the test data is shown in Figure 8.

In this example, we were interested in whether RL can give appropriate duration matching asset allocations at a given time point. We used five time points, that is, at

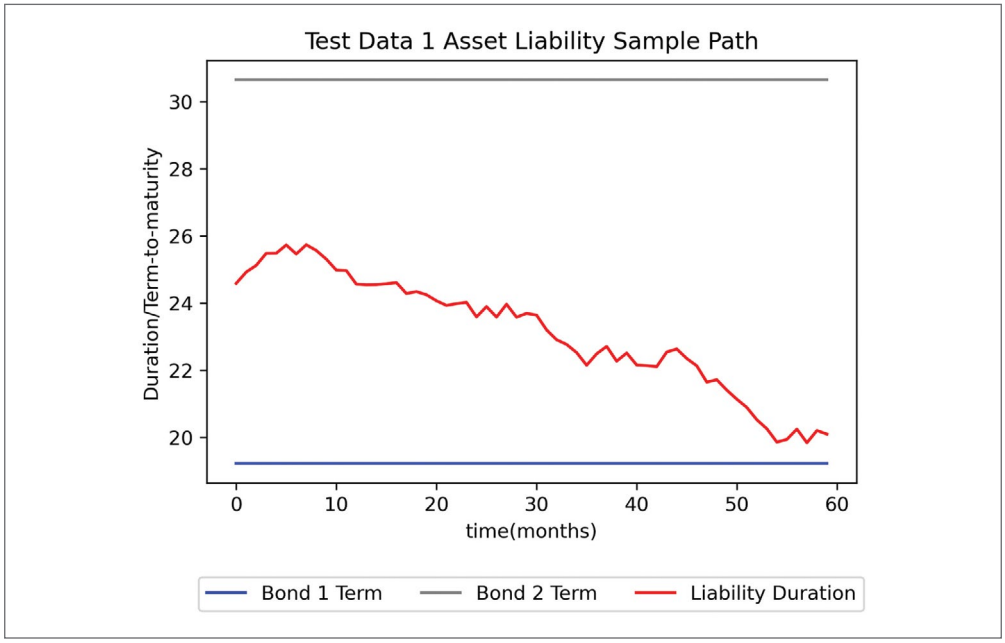

FIGURE 8 Monte Carlo simulation testing sample path

12 months, 24 months, 36 months, 48 months and 60 months as test time points. The bond maturities and liability duration at these time points are given in Table 1.

Table 1 Test sample path values at annual time-points (months)

Time-points	12	24	36	48	60
Liability duration	24.97	24.02	22.15	1.65	20.10
Bond 1 Term	19.22	19.22	19.22	19.22	19.22
Bond 2 Term	30.66	30.66	30.66	30.66	30.66

We first applied the Redington immunisation ALM on the sample Test 1 data, and we got the traditional asset allocations in Table 2.

Table 2 Test sample path annual time-points (months) Redington immunisation allocations

Time-points	12	24	36	48	60
Liability duration	24.97	24.02	22.15	21.65	20.10
Bond 1 term	19.22	19.22	19.22	19.22	19.22
Bond 1 allocation	49.7%	58.0%	74.4%	78.8%	92.3%
Bond 2 term	30.66	30.66	30.66	30.66	30.66
Bond 2 allocation	50.3%	42.0%	25.6%	21.2%	7.7%

We applied the Redington immunisation approximation solution derived in Equation 5. A function which executed this formula was defined in Python.

We then also applied the DRL ALM outlined in Section 6.2.3 on the sample Test 1 data. The saved trained agent object was retrieved memory using the restore function of the agent class and ran on the test data. We got the RL asset allocations shown in Table 3.

Table 3 Test sample path annual time-points (months) RL allocations

Timepoints	12	24	36	48	60
Liability duration	24.97	24.02	22.15	21.65	20.10
Bond 1 term	19.22	19.22	19.22	19.22	19.22
Bond 1 allocation	49.2%	58.1%	72.8%	76.7%	93.7%
Bond 2 term	30.66	30.66	30.66	30.66	30.66
Bond 2 allocation	50.8%	41.9%	27.2%	23.3%	6.3%

One can already observe that the asset allocations are relatively close to each other for the sample scenario between the Redington immunisation's allocations and the RL allocations.

Based on its respective asset allocations to zero-coupon bond 1 and zero-coupon bond 2, we then calculated the traditional ALM method's outcomes of the asset duration outcomes using the formula shown by Equation 13:

$$D\left(Trad_{it}\right) = \omega_{1it} T(Z_1)_{it} + \omega_{2it} T(Z_2)_{it} \ , \tag{13}$$

where:

— $D\left(Trad_{it}\right)$ is the Redington immunisation asset duration at time at time t for this *ith* test scenario.
— ω_{1it} is the derived Redington immunisation asset allocation for zero-coupon bond 1 time at time t for this *ith* test scenario.
— ω_{2it} is the derived Redington immunisation asset allocation for zero-coupon bond 2 time at time t for this *ith* test scenario.
— $T(Z_1)_{it}$ is the observed term-to-maturity for zero-coupon bond 1 time at time t for this *ith* test scenario.
— $T(Z_2)_{it}$ is the observed term-to-maturity for zero-coupon bond 2 time at time t for this *ith* test scenario.

Based on its respective asset allocations to zero-coupon bond 1 and zero-coupon bond 2, we also calculated the RL model's asset duration outcomes using the formula shown by Equation 14:

$$D\left(RL_{it}\right) = \omega_{it} T(Z_1)_{it} + \omega_{it} T(Z_2)_{it} \ , \tag{14}$$

where:

— $D\left(RL_{it}\right)$ is the DRL asset duration at time t for this *ith* test scenario.
— ω_{1it} is the derived DRL asset allocation for zero-coupon bond 1 time at time t for this *ith* test scenario.
— ω_{2it} is the derived DRL asset allocation for zero-coupon bond 2 time at time t for this *ith* test scenario.
— $T(Z_1)_{it}$ is the observed term-to-maturity for zero-coupon bond 1 time at time t for this *ith* test scenario.
— $T(Z_2)_{it}$ is the observed term-to-maturity for zero-coupon bond 2 time at time t for this *ith* test scenario.

The results for the above calculations are given in Table 4 for both the Redington immunisation ALM and DRL ALM methods. We also show the differences between the calculated asset portfolio duration of the Redington immunisation ALM method and the DRL ALM asset durations in Table 4.

TABLE 4 Test sample path annual time-points (months) duration differences

Timepoints	12	24	36	48	60
Liability duration	24.90	24.02	22.15	21.65	20.10
Immunisation ALM duration	24.90	24.02	22.15	21.65	20.10
DRL ALM duration	25.04	24.01	22.33	21.89	19.95
Duration difference (years)	0.1	–0.0	0.2	0.2	–0.2
Duration difference (%)	0.3%	0.0%	0.8%	1.1%	(0.8%)

We can see that, as expected, the Redington immunisation approach's asset duration gives shows an exact match to the liability duration. This is as expected as this is theoretical analytical approach, albeit with certain limiting underlying assumptions discussed in Section 4.1. We can see that overall there is a slight difference. This shows that RL ALM performs just as well as the Redington immunisation ALM on matching durations at a given time. The mean deviation in asset duration is 0.07 years, and the mean percentage difference is 0.28%, which are both negligible. The results above are only for one sampled test data scenario. In the next section, we aggregate the differences across all the 1 000 scenarios in Test Data 1 so that we can assess the RL ALM performance across various scenarios.

7.1.2 AGGREGATE EVALUATION ON TEST DATA

For each of the 1 000 test scenarios, we compared the duration of the asset allocation from the RL approach to that of the Redington immunisation ALM. We did this at five time points, that is, after 1, 2, 3, 4 and 5 years. At each time point, we plotted the histogram of the differences between the RL and Redington immunisation approaches' duration differences as seen in Figures 9 to 13.

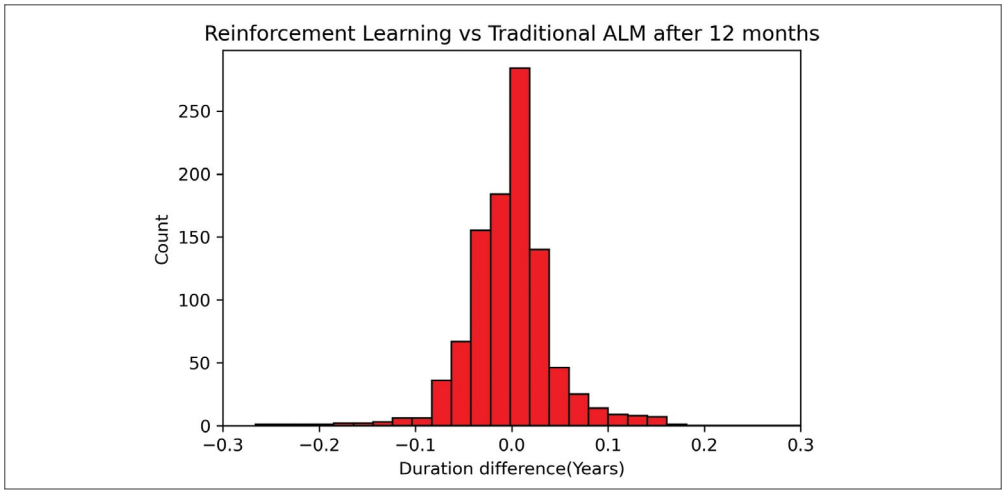

FIGURE 9 Aggregate duration differential at 12 months

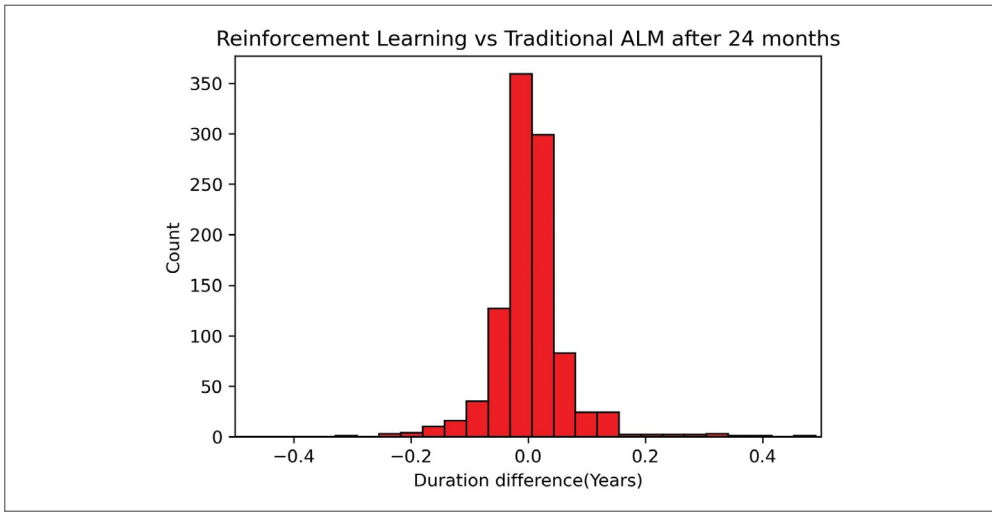

FIGURE 10 Aggregate duration differential at 24 months

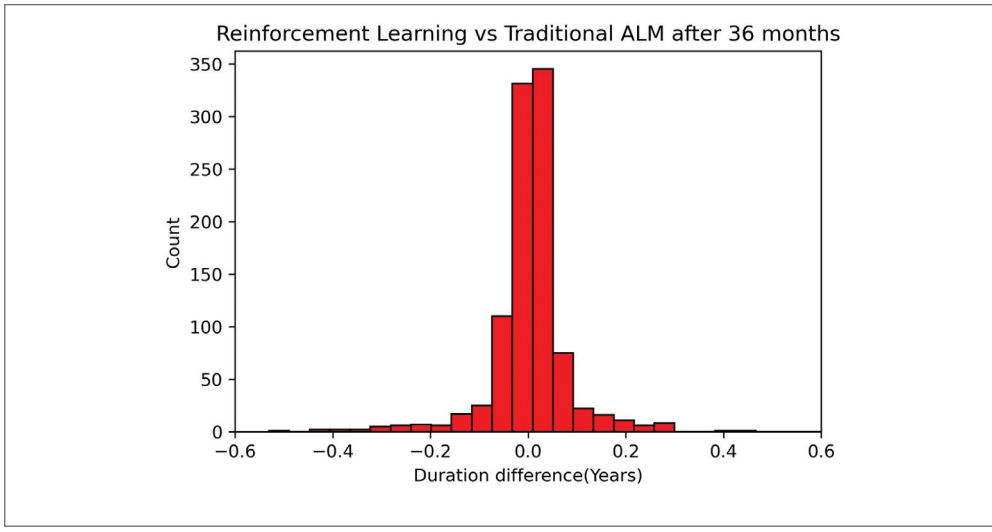

FIGURE 11 Aggregate duration differential at 36 months

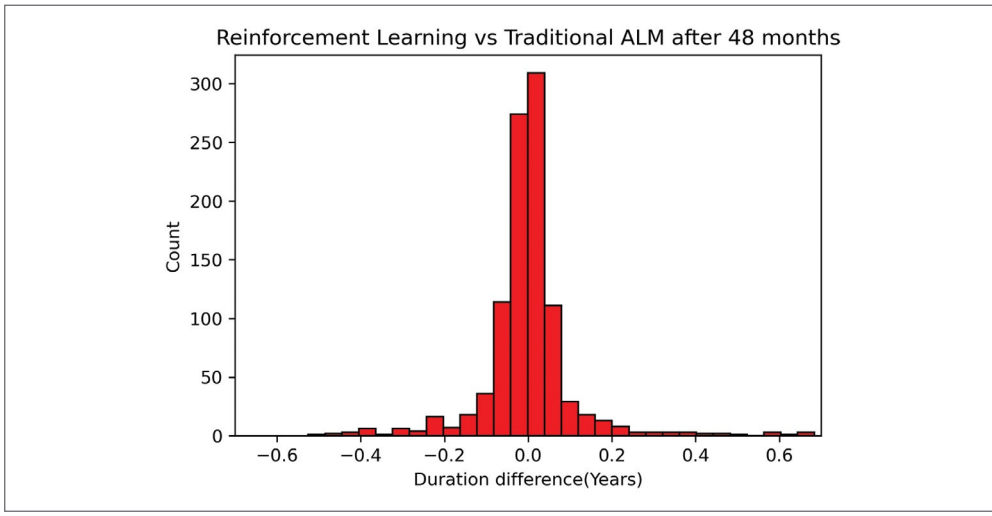

FIGURE 12 Aggregate duration differential at 48 months

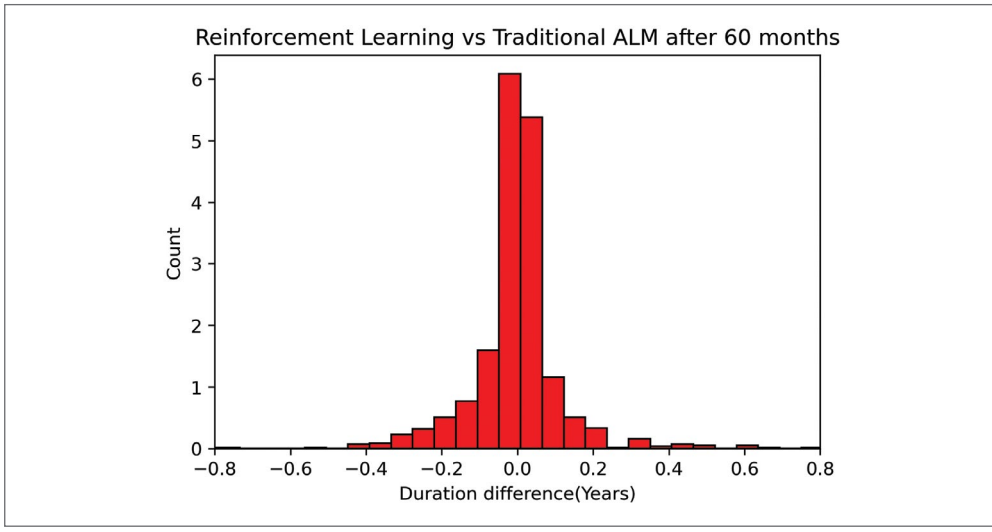

FIGURE 13 Aggregate duration differential at 60 months

The 95% confidence interval (*CI*) of the difference can be estimated by the expression shown in Equation 15 (Guignard et al., 2021):

$$95\%CI\left(Lower;Upper\right)=\left(\hat{\mu}-1.96\times\hat{\sigma};\hat{\mu}+1.96\times\hat{\sigma}\right). \tag{15}$$

From Equation 15, we estimated the 95% confidence interval of the duration difference at time 12 months to be (–0.09; 0.09), at 24 months is (–0.13; 0.13), at 36 months is (–0.16;

0.17), at 48 months is (–0.22; 0.21) and at time 60 months is (–0.25; 0.24). All of these values were mostly within 1% of the theoretical Redington immunisation duration.

We can see that the mean difference between the two approaches is close to zero (0) at all five (5) test time points. Furthermore, the distribution is symmetric, which shows that there is no bias in the performance of RL compared to the traditional approach. All the 95% confidence intervals at the five (5) test time points are narrow around zero, and within 1% of the Redington immunisation duration results. This means that at a 95% confidence level, the allocations of the RL ALM are statistically the same as those from the theoretical immunisation results.

7.1.3 AGGREGATE EVALUATION ON STRESS TEST DATA

The stress test conditions explored correspond to a situation where there are significant shifts in either the level or shape of the yield curve in a short space of time. These periods would cause high volatility in the liability duration.

In this experiment, we repeated exactly the same experiment as in Section 7.1.2 with the exception of one difference. The only difference was that the test data used for evaluating the RL ALM had a maximum possible duration change (Δ) of one year between time points instead of half a year (0.5) as in Section 7.1.2.

We named this second test simulation data Test Data 2. This means that in this experiment we test how the same RL ALM model would be able to perform on unseen data which is, in addition, much more volatile compared to the data it had seen before. The aim was to test the RL ALM on its adaptability to new conditions.

Figures 14 to 18 show histograms of the duration differences at 12 months, 24 months, 36 months, 48 months and 60 months, respectively.

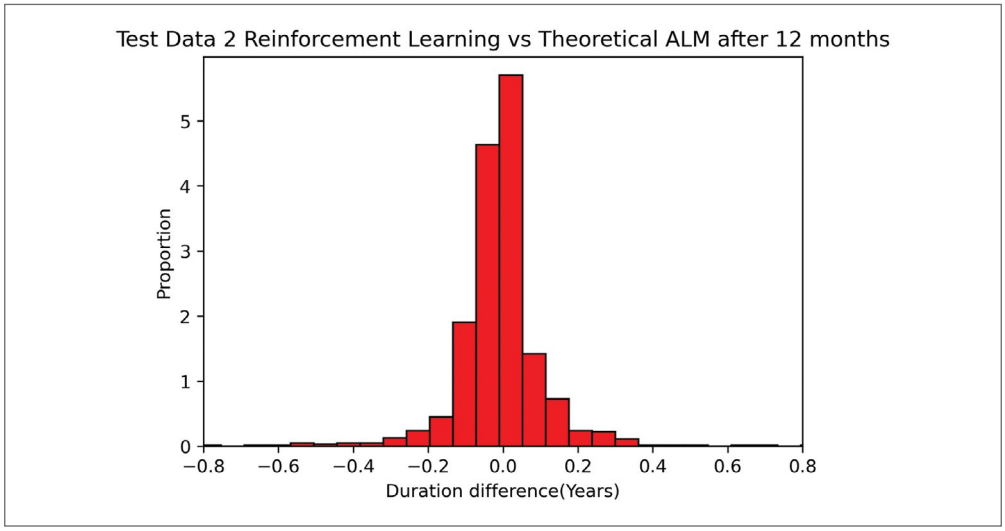

FIGURE 14 Aggregate duration differential at 12 months

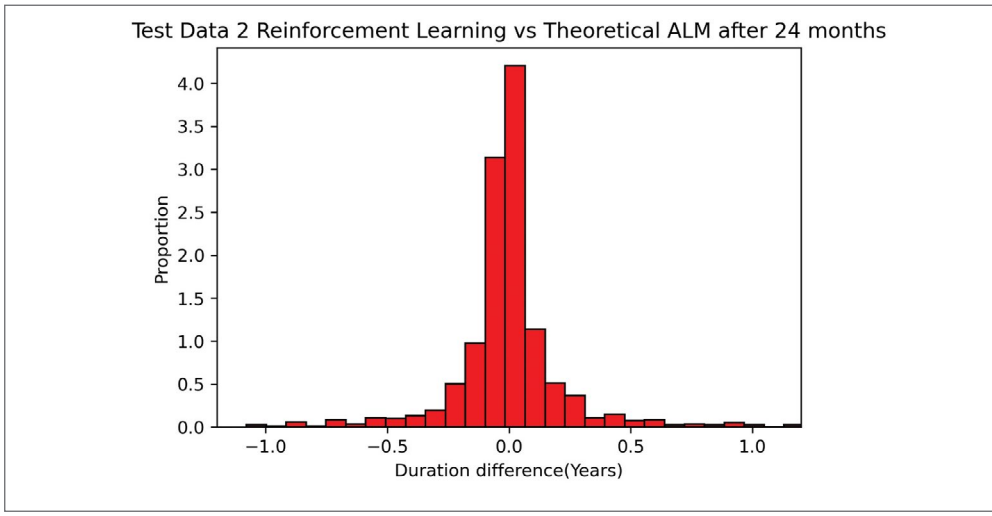

FIGURE 15 Aggregate duration differential at 24 months

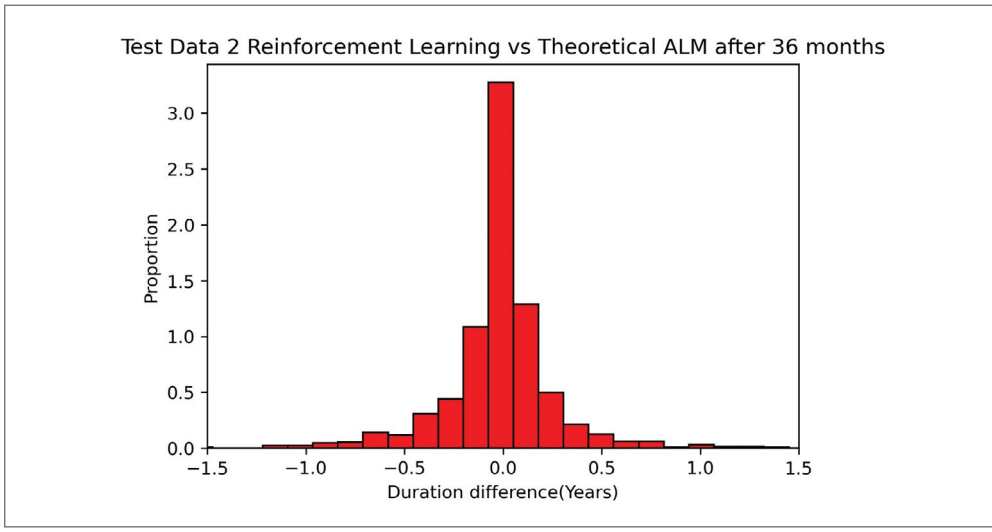

FIGURE 16 Aggregate duration differential at 36 months

The 95% confidence interval of the duration difference at 12 months is (–0.71; 0.71), at 24 months is (–0.24; 0.24), at 36 months is (–0.44; 0.44), at 36 months is (–0.56; 0.56), at 48 months is (–0.62; 0.62) and at 60 months is (–0.71; 0.71).

Although the confidence intervals from the stress testing above are wider than what we saw in Section 7.1.2, it is remarkable that they are still very narrow (relative to the range of possible duration values of up to 40 years or more). The mean difference between RL duration and the theoretical level is still close to 0 at all five test time points. Furthermore, the distribution is still symmetric, which shows that there is still no bias in the performance of RL.

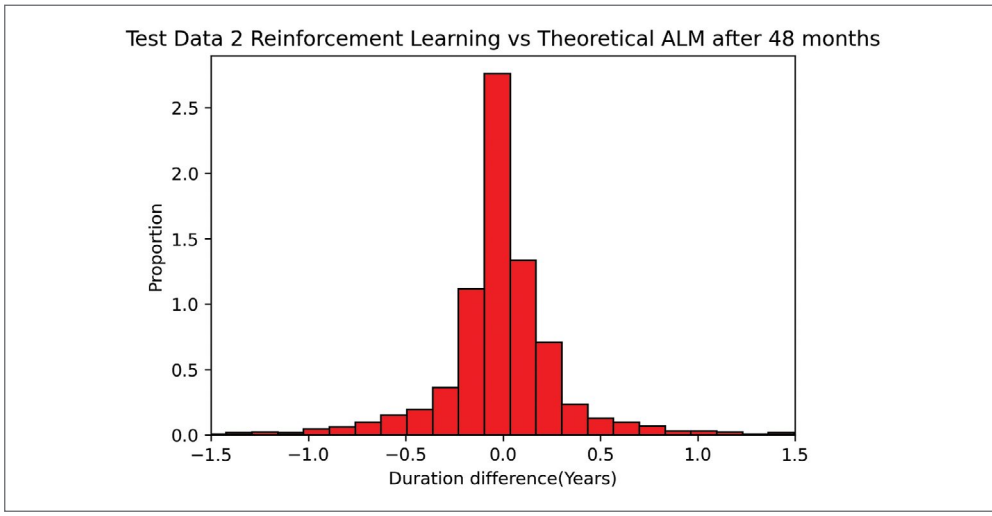

FIGURE 17 Aggregate duration differential at 48 months

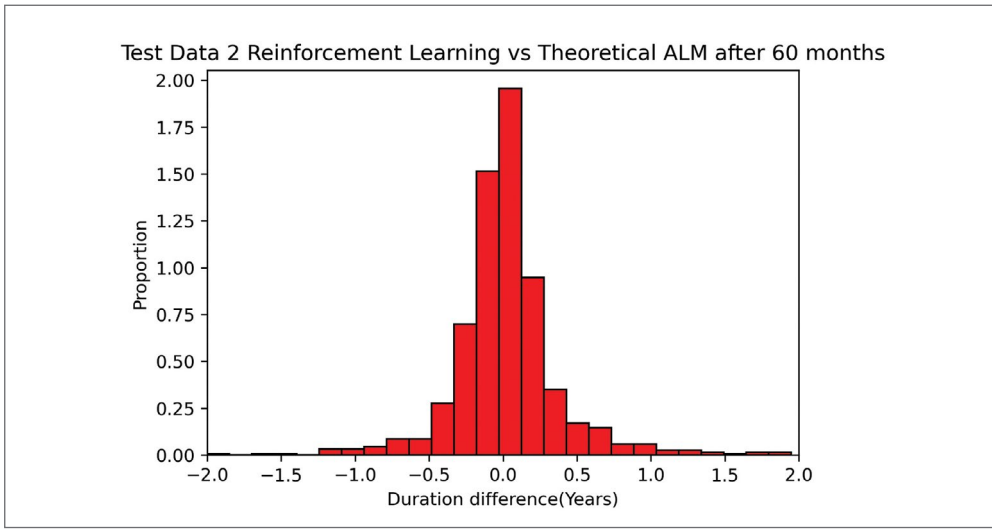

FIGURE 18 Aggregate duration differential at 60 months

All the 95% confidence intervals at the five test time points are narrow around zero, and they all include zero within approximately 2–3% of the theoretical duration results. This is still an exceptional level of performance despite the test data being much more volatile compared to the simulations on which it was trained. In practical applications where the conditions differ from theoretical assumptions, this level of robustness would outperform the traditional approaches.

7.2 Deep reinforcement learning ALM compared to a benchmark traditional ALM approach

In this section, we discuss the evaluation of the second research question, which was: How does DRL's asset allocations compare to those of a benchmark real-world traditional ALM strategy in hedging a changing liability portfolio? Does it result in better interest rate hedging and to what extent?

In the first research question, we found that DRL gives similar results to Redington immunisation at a given point in time. However, in order to fully assess the performance benefits of DRL ALM in practice it was important to compare against a typical ALM implementation that may be in force in practice. We contrasted the benefits of DRL ALM implementable on a daily basis against a traditional ALM approach which is usually feasible on a monthly or on a weekly level for efficient organisation. In order to be conservative in our comparison, we assumed a weekly rebalancing for the traditional approach. We assessed the differences in outcomes at the end of the month between the daily RL ALM approach and the weekly traditional ALM approach.

7.2.1 SAMPLE CASE EVALUATION

For testing purposes, we generated a new set of data not seen by the RL agent during the training phase. We generated 1 000 test scenarios; hence, the data was also 3-dimensional but with dimensions of 30 × 1 000 × 3. For illustrative purposes, we selected a scenario from the test data, which is graphically shown in Figure 19.

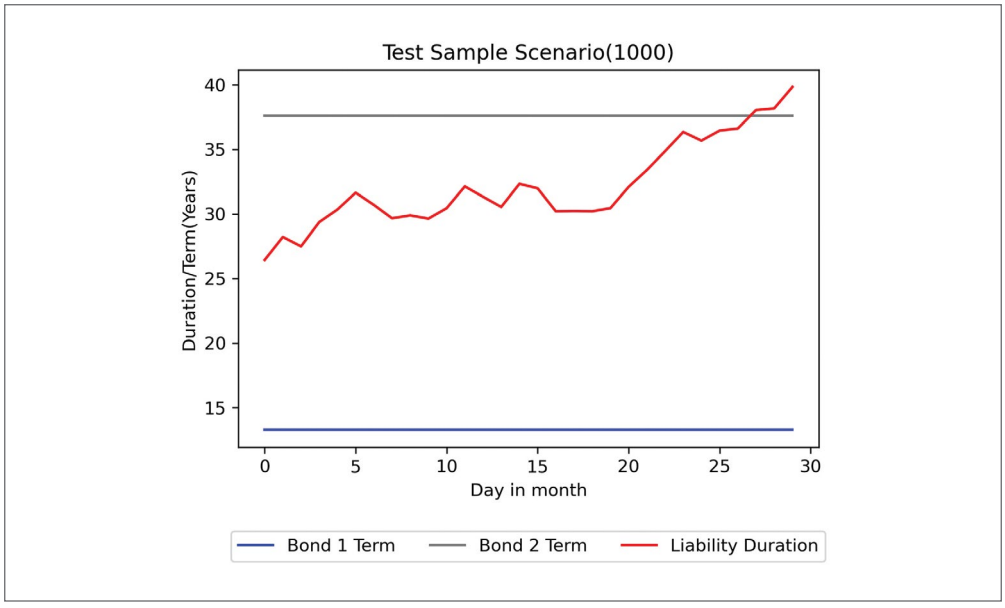

FIGURE 19 Monthly Monte Carlo simulation test sample path

In the above scenario, the liability duration gradually reduced from 23.4 years on Day 1 to 16.8 years on Day 23. From there onwards, the liability duration gradually increases to 19.1 years on day 30. The term for Bond 1 is 10.2 years and that for Bond 2 is 33.1 years. We then applied the traditional ALM approach on weekly rebalancing frequency, followed by the RL ALM on a daily rebalancing regime. We determined their respective asset allocations for Bond 1 and Bond 2. We also determined the daily theoretical asset allocations based on applying the traditional ALM approach on a daily basis. The theoretical asset allocation would represent the benchmark asset allocation if the analytical asset allocation had been possible daily.

The asset allocations under the theoretical, traditional and RL regimes are shown in Figures 20 and 21.

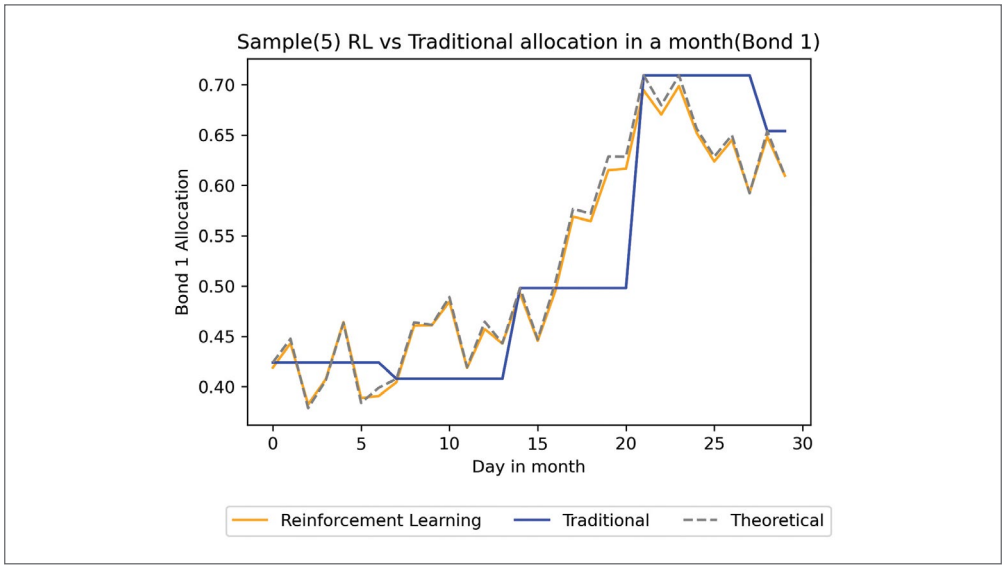

FIGURE 20 Benchmark ALM asset allocation comparisons (Bond 1)

From these figures, we can see that the RL AML approach allocated is in close proximity to the theoretical level at most time points. As expected, the allocation towards the shorter duration bond (Bond 1) increases until Day 23, which coincides with the decrease in duration noted in this period in the sample graph. As expected, the allocation towards the longer duration bond (Bond 2) increases from day 23 until Day 30, which coincides with the increase in liability duration for this period.

A notable observation is that the traditional ALM allocation pattern is step-wise. It is constant except on the days on which there is weekly rebalancing. On the rebalance days, the allocation is close to the allocations for both the RL ALM and the theoretical ALM. However, in between the rebalancing time points, there are deviations of the traditional allocations from the RL and theoretical allocations.

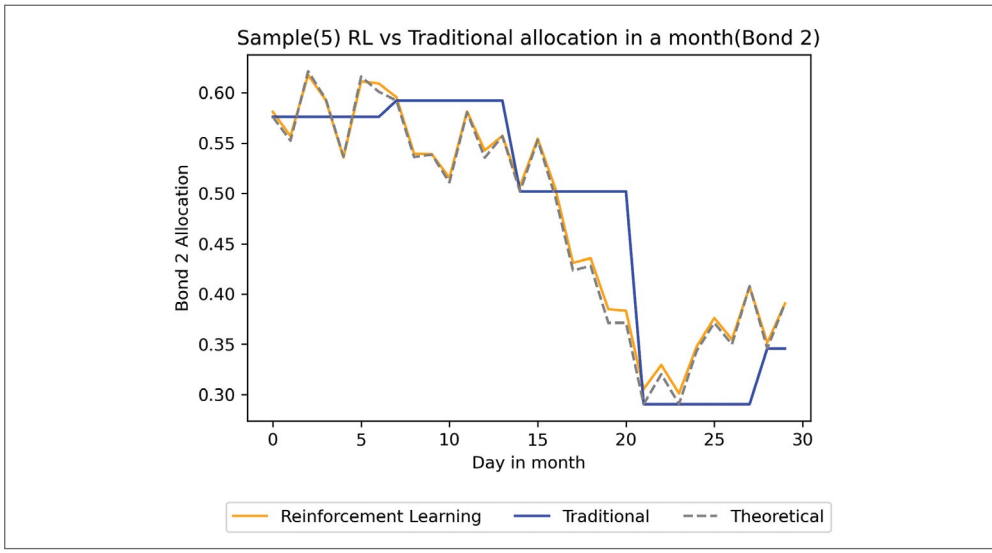

FIGURE 21 Benchmark ALM asset allocation comparisons (Bond 2)

Although there is weekly rebalancing in the traditional regime, there is still a risk that there are sudden changes in interest rates at a time when this traditional weekly allocation deviates significantly from the theoretical allocations. If this happens, there could be significant mismatches between the movements of the value of liabilities relative to that of the assets portfolio. This risk will be quantified in the next section.

The analysis done above was based on an illustration of one test scenario only. In the next section, we aggregate the relative performance of RL against the traditional methods across all the 1 000 test data scenarios.

7.2.2 AGGREGATE EVALUATION

For each of the 1 000 scenarios in the test data, we determine the asset allocations at Day 30 for the RL ALM, traditional ALM with weekly rebalancing and the theoretical ALM. Based on these, we determined the respective asset duration for each approach for each of the 1 000 scenarios.

We used the theoretical asset duration as the benchmark, which is the same as the actual liability duration on Day 30. The closer a result is to this level, the better the outcome of the respective method. Therefore, to assess the RL ALM performance, we calculated the difference of its asset duration from the theoretical one on Day 30. We repeated this for each of the 1 000 scenarios and plotted a histogram of the duration differences in Figure 22.

From Figure 22, we can see that the duration differences are centred symmetrically and bell-shaped around 0. The mean difference is 0.055 years, which shows that, on average, the outcome of the RL ALM is very close to what one would practically require for sufficient risk management. The standard deviation of these differences is 0.28. This means that 95%

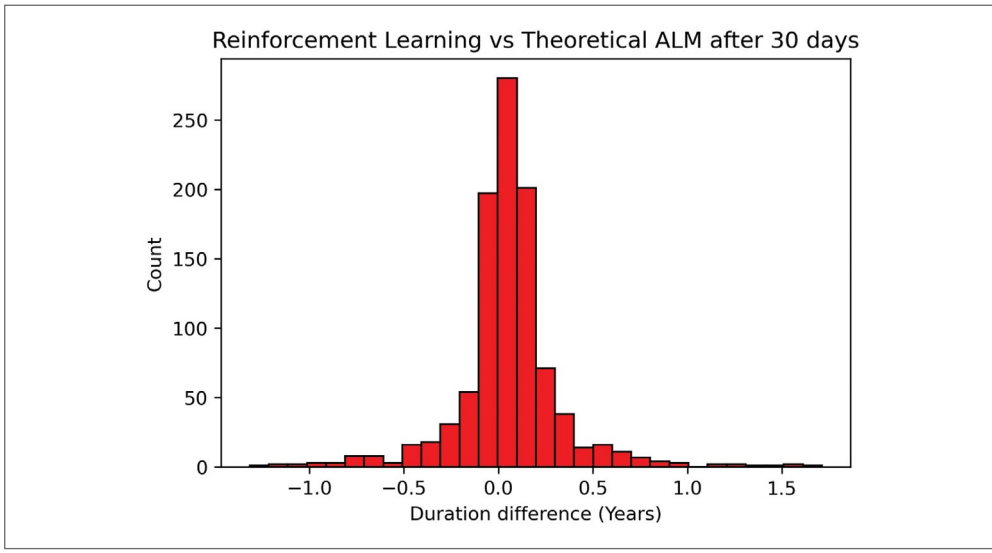

FIGURE 22 Daily reinforcement learning aggregate duration differential at Day 30

of the differences are not more than 0.56 years (approximately two standard deviations) years from the theoretical level.

Therefore, in order to assess the traditional ALM performance, we calculated the difference in its asset duration from the theoretical at Day 30. We repeated this for each of the one-thousand (1 000) scenarios and plotted a histogram of the duration differences in Figure 23. From this figure, we can see that the duration differences are still centred

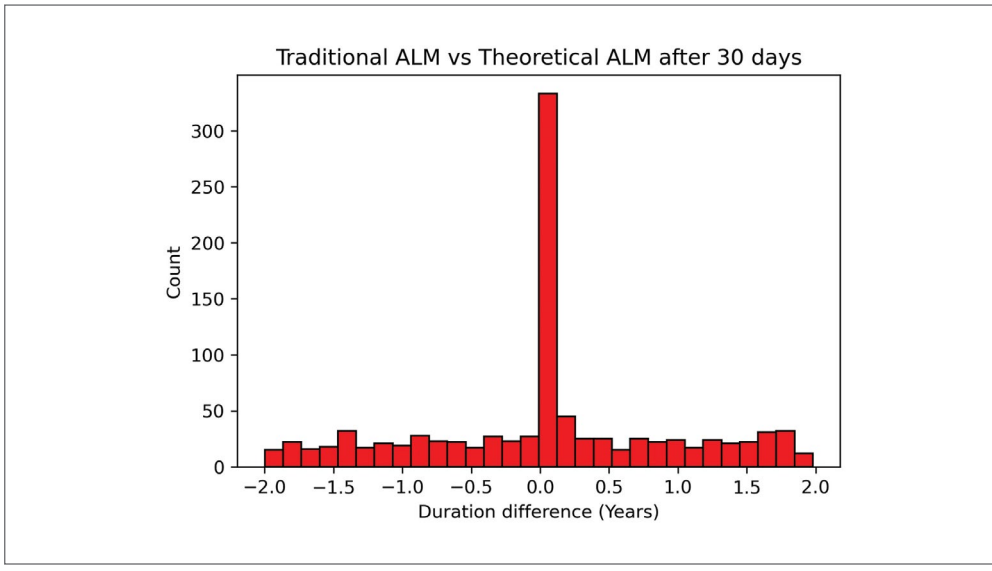

FIGURE 23 Weekly traditional aggregate duration differential at Day 30

around 0 but are no longer bell-shaped. There is a higher proportion of differences which are significantly different from 0 at the end of the month.

The mean difference is around 0.03, but the standard deviation is much higher at 0.92. This means that 95% of the differences are not more than 1.84 years (approximately two standard deviations) from the theoretically correct level. This threshold level for the traditional ALM approach is over three times that of the RL ALM (0.56 years). This means that with all else being the same, the traditional approach can differ from the theoretical duration by three times as much as the RL approach.

The modified duration of the bond gives the impact of a 100-basis point (1%) on the price of a bond. This is given by the formula shown in Equation 16 (Bierwag & Fooladi, 2006; Wu, 2022):

$$MD = \frac{D}{1 + \frac{YTM}{n}}, \tag{16}$$

where:

— MD is the modified duration. D is the Macaulay duration, that is, the usual duration we have been referring to for the rest of the paper (as defined in Equation 2) in years.
— YTM is the yield-to-maturity of the bond, that is, the nominal annual interest rate.
— n is the frequency of interest compounding per year. In our case $n = 1$ because the bonds are zero-coupon bonds. In this case of $n = 1$, the YTM is also the effective annual interest rate.

From the above formulation, we can deduce that the estimated average net portfolio (asset less liabilities) from traditional ALM with a weekly rebalancing at the end of 30 days has a 1.71% change in value from a 1% change in interest rates. On the other hand, the estimated average net portfolio (assets less liabilities) from RL ALM with a daily rebalancing has a 0.53% change in value from a 1% change in interest rates.

From these calculations, we can deduce that the RL ALM's net portfolio will be much more securely hedged (approximately three times better) compared to the traditional ALM's net portfolio, with all else being the same. This is a particularly important consideration in a situation where interest rates are increasing. If the asset duration is much higher than the liability duration, then the asset portfolio value will fall by a bigger margin compared to the liability value, representing bigger exposure to interest rate risk.

8. DISCUSSION, CONCLUSION AND RECOMMENDATIONS

8.1 Results discussion

From Section 7.1, the first finding was that, in general, DRL could successfully replicate the performance of Redington's immunisation theory. We calculated confidence intervals for the differences between immunisation theory results and DRL at test time points.

The results first demonstrated that DRL ALM can achieve duration-matching outcomes within 1% of the theoretical immunisation theory at a 95% confidence level. The RL ALM was able to achieve the same level of performance despite not relying on interest rate assumptions compared to the traditional method. In Section 7.1, we also confirmed that as interest rate conditions deviate from precise theoretical assumptions (which is associated with more volatile liability duration), DRL was able to maintain its duration matching capabilities. This illustrated the increased robustness of RL to market conditions compared to the traditional methods. Furthermore, RL did not require external restrictions on many of the optimisation parameters, such as the range of values for the weights. We noticed that the RL agent was able to automatically learn, based on the reward function, that the appropriate weights needed to range between zero (0) and one (1) because the context did not allow short-selling. On the other hand, in immunisation theory, we needed to explicitly express this restriction or make a manual adjustment afterwards. Therefore, RL demonstrated the ability to perform as well as immunisation theory by just expressing the objective with fewer assumptions and restrictions.

From Section 7.2, the investigations showed that DRL ALM could be implemented more frequently with daily rebalancing of the asset portfolio allocation for duration matching as the liability continuously changes. This resulted in significantly better interest rate hedged portfolios than the traditional ALM approach with weekly rebalancing, which is the most frequent that can be feasibly achieved with traditional strategies in practice in most cases. In our case, we estimated that RL net portfolio values are approximately three times less sensitive to interest rate movements compared to the traditional ALM on a weekly regime. This is a significant outperformance, especially considering the large financial sums that large institutions manage. This ability to be implemented on a more frequent basis is because DRL requires less human intervention compared to the traditional approach. This is partly because the RL approach relies less on theoretical assumptions. In addition, RL ALM is performed within scalable computational frameworks, which unlock faster and more automated processing by leveraging powerful open-source libraries such as TensorFlow. This enables one to practically apply RL regularly at an enterprise level with high-velocity and voluminous data. RL also achieves more consistent hedging within the month since its allocations track better to the theoretical level. This means that with DRL there are fewer times when there are significant mismatches between assets and liabilities.

8.2 Conclusion

We applied a combination of RL and deep learning to actuarial ALM. This combination is called DRL.

Before this paper, RL had been applied in the actuarial and quantitative finance domains to mostly carry out trading, portfolio allocation, and derivatives hedging (Hariom et al., 2020; Kolm & Ritter, 2020; Dixon & Halperin, 2019). Although the recent application of deep learning and RL for ALM was successful in banking, it was based on approaches

customised specifically for retail banking (Krabichler & Teichmann, 2023; Englisch, et al., 2023). There had been no well-documented literature and applications of RL, which considered both sides of a risk-taker's balance sheet in all spheres of risk management theory. This paper fills this gap both in the literature and application by providing an avenue for the improvement of actuarial and quantitative finance practice through a solution that replicates and potentially improves upon Redington's immunisation theory.

For testing purposes, we used Monte Carlo methods to simulate data similar to that encountered in the real world by risk-taking institutions. For each simulation, we had stochastic variation in the liability duration and the asset portfolio's maturity terms. We assumed that the assets were comprised of two zero-coupon bonds. The approach can, as we discussed, be validly extended to coupon-bearing bonds (Jarrow, 2004; Jarrow & Turnbull, 2000). We used 10 000 simulations for training the RL ALM agent. The data sets were generated from separate Monte Carlo simulation runs and were thus unseen by models during training, which provided a robust testing regime.

For the purpose of comparison, we set up the Redington immunisation model, which is the foundation theory for most traditional approaches to ALM. It is based on solving simultaneous equations to determine the appropriate asset weights required to achieve matching of the asset and liability duration. We noted the restrictive theoretical assumptions as well as several practical problems, including the resource-intensive requirements for careful monitoring and professional adjustments.

We then developed the RL framework for implementation. The RL approach required the development and implementation of object-oriented orientated programming (OOP) or class-based programming in Python programming. First, we defined an autonomous agent as a class that can create RL objects with specific attributes, modules, and functions. We then defined the critical RL components, such as the agent, the environment, the states of the environment (simulation data), the actions of the agent (asset allocations), and the reward function (sum square error). In addition, the core functionality of RL in its data processing and computations was implemented using the TensorFlow library in Python. We used TensorFlow because of its enhanced abilities to manage large datasets and demanding computational jobs due to its unique computation graph structure (Lang, 2022; Pang et al., 2020; Geron, 2019). Within this TensorFlow framework, we incorporated a deep learning model, the long-short-term memory recurrent neural network (LSTM-RNN) (Smagulova & James, 2019; Staudemeyer & Morris, 2019). The LSTM-RNN enabled in-depth learning for the RL agent to be able to learn and discern time-dependent trends in the data in the formulation of its policy.

We compared the performance of the DRL ALM to both Redington immunisation theory and a weekly traditional ALM as a benchmark of the current best possible implementation. The results first demonstrated that DRL ALM can achieve duration-matching outcomes within 1% of the theoretical immunisation theory at a 95% confidence level. Second, compared to a benchmark weekly rebalancing traditional ALM regime, high-frequency

DRL ALM achieved superior outcomes that were three times less sensitive to interest rate changes. DRL ALM also demonstrated capacity for increased automation, speed, flexibility, and multi-objective optimisation in ALM, thereby furthering the potential for reducing the negative impact of human limitations and improving risk management outcomes. The findings and principles presented in this study apply to various institutional risk-takers, including insurers, banks, pension funds, and asset managers.

Overall, DRL was not only able to perform similar tasks as traditional methods in general but also improved many aspects of ALM. DRL demonstrated the potential to improve ALM by providing an avenue for more frequent and automated asset re-allocations for better interest rate hedging, less reliance on theoretical assumptions and restrictions, high frequency, increased robustness in varying market conditions, and increased flexibility. We demonstrated that, combining deep learning and RL, which complement each other, indeed gives a DRL solution with both strong perceptive and strategic capabilities. This combined power yields significant outperformance for ALM problems.

8.3 Recommendations for future research

There are several areas of potential additional research that can be pursued to build on the research done in this paper.

First, we discovered that more complex problems, especially those with multi-objective optimisation, required more simulations and longer training times. We, however, had the limitation of using a personal computer with limited computing power. One area of easy improvement would be to generate more simulations, such as in the range of millions, and train the DRL agent for longer. This would involve running the RL on an environment with higher compute servers, such as those with parallel computing and graphical processing units (GPUs) or tensor processing units (TPUs). In addition, it would be worth investigating the degree to which the performance improves, which would be the expectation.

Second, another potential research area is to test how the trained DRL agent would perform on real-world data. This would involve deploying the trained reinforcement agent on a real-world investment management or asset management floor with all the real-world challenges. Some development would be needed to integrate the RL agent object and its associated Python program into the enterprise systems of the institution deploying the agent. In this deployment, a feedback loop that feeds data to the RL from the institution's asset and liability data systems and simultaneously sends the agent's actions would need to be established.

Third, another area of future research would be to incorporate additional restrictions or parameters such as regulatory capital regimes such as Solvency II and Solvency Assessment and Management (SAM), liquidity requirements, or other risk assessment measures such as value-at-risk. It would be valuable to assess how easily the current solution will extend to incorporate these additional objectives and how it performs in balancing the different objectives.

Fourth, we would also recommend assessing the feasibility and impact on the performance of including different data types in the states, such as asset returns, asset volatility, detailed risk profiles of liabilities (such as ages, health status, gender, etc.), and even non-numeric data. These non-numeric data inputs could include some market sentiment data or some internal institutional data. Deep learning models are well known for their ability to cope well with non-numerical data, such as in natural language processing, and it would be interesting to investigate whether this strength can be leveraged in DRL.

Fifth, in this paper, the asset portfolio was based on zero-coupon bonds. This is not a very restrictive assumption, as the findings are applicable when hedging is carried out with coupon-bearing bonds. However, it would be valuable to carry out research on the effect of incorporating more complicated asset types such as equities, convertible bonds, derivatives, etc. In addition, it would be interesting to investigate the impact on the choices of the RL solution.

Sixth, a promising avenue of research would be the improvement of this application with human-in-the-loop RL. Human-in-the-loop RL allows humans to assist the agent in the training processes, thereby enabling the agents to learn faster, adapt faster, and perform better. Human-in-the-loop RL has been successfully applied to improve solutions to difficult problems such as autonomous vehicles (Wu et al., 2023). In the DRL ALM application, this holds the promise of allowing experienced professionals to correct or aid the agent in real time for better outcomes.

Lastly, we noticed some practical challenges that one would need to overcome in applying DRL. The first challenge is that there are no off-the-shelf consolidated packages, libraries, or pre-defined modelling approaches for implementing RL. This means that one typically has to program many of the steps and details of the autonomous reinforcement agent classes and objects, along with their attributes and functions. Furthermore, implementing deep learning in TensorFlow requires a good understanding of unique and specific computational graph formulation and syntax. A good understanding of TensorFlow is also required for managing all the data in high-dimensional matrices (Tensors), which are different from the usual handling of data in most libraries and programming languages.

We, therefore, believe that there are also opportunities for research into simplifying the development of RL solutions, especially when used with TensorFlow. There is also a gap in making the deployment of a DRL agent easier and quicker. Some work has been done in areas such as stable baselines (D'Eramo et al., 2020; Baselines, 2022). It would be valuable to assess how easy it is to apply these approaches.

REFERENCES

Abrate, C, Angius, A, De Francisci Morales, G, Cozzini, S, Iadanza, F, Puma, LL, Pavanelli, S, Perotti, A, Pignataro, S & Ronchiadin, S (2021). Continuous-action reinforcement learning or portfolio allocation of a life insurance company. *In* Y Dong, N Kourtellis, B Hammer & JA Lozano (Eds.). Machine learning and knowledge discovery in databases. *Applied Data Science Track* (pp. 237–252). Springer International Publishing.

Altché, F & de La Fortelle, A (2017). An LSTM network for highway trajectory prediction. 2017. IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), pp. 353–359. https://doi.org/10.1109/ITSC.2017.8317913.

Arulkumaran, K, Deisenroth, MP, Brundage, M & Bharath, AA (2017). Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, **34**(6), 26–38. https://doi.org/10.1109/MSP.2017.2743240.

Barr, C (2023). What's going on with First Republic Bank? *WSJ*. https://www.wsj.com/articles/first-republic-bank-stock-rescue-what-to-know-a22a3d (Accessed on 04/02/2023).

Baselines, S (2022). Welcome to stable baselines docs! RL baselines made easy—stable baselines 2.10.3a0 documentation. https://stable-baselines.readthedocs.io/en/master/ (Accessed on 11/21/2022).

Bierwag, G & Fooladi, I (2006). Duration analysis: An historical perspective. *Journal of Applied Finance*, **16**.

Bondt, WD, Mayoral, RM & Vallelado, E (2013). Behavioral decision-making in finance: An overview and assessment of selected research/la toma de decisión en las finanzas del comportamiento: Estado de la cuestión a partir de los trabajos seleccionados. *Revista Española de Financiación y Contabilidad*, **42**(157), 99–118. http://www.jstor.org/stable/42782820 (Retrieved November 22, 2022).

Bühler, H, Gonon, L, Teichmann, J & Wood, B (2018). *Deep hedging*. https://arxiv.org/abs/1802.03042.

Cheridito, P, Ery, J & Wüthrich, MV (2020). Assessing asset-liability risk with neural networks. *Risks*, **8**(1). https://doi.org/10.3390/risks8010016.

Chiu, C-J, Ho, AY-F & Tsai, L-F (2022). Effects of financial constraints and managerial overconfidence on investment-cash flow sensitivity. *International Review of Economics Finance*, **82**(135–155). https://doi.org/https://doi.org/10.1016/j.iref.2022.06.008.

Daga, A (2023). What happened at Credit Suisse and how did it reach crisis point? Reuters. https://www.reuters.com/business/finance/credit-suisse-how-did-it-getcrisis-point-2023-03-16/ (Accessed on 04/02/2023).

D'Eramo, C, Tateo, D, Bonarini, A, Restelli, M & Peters, J (2020). Mushroomrl: Simplifying reinforcement learning research. https://doi.org/10.48550/ARXIV.2001.01102.

Devraj, A & Meyn, S (2017). Zap Q-learning. https://proceedings.neurips.cc/paper/2017/file/4671aeaf49c792689533b00664a5c3ef-Paper.pdf.

Dixon, MF & Halperin, I (2019). The four horsemen of machine learning in finance. *SSRN*, 26, 18–28. https://ssrn.com/abstract=3453564.

Dong, S, Wang, P & Abbas, K (2021). A survey on deep learning and its applications. *Computer Science Review*, **40**, 100379. https://doi.org/https://doi.org/10.1016/j.cosrev.2021.100379.

Englisch, H, Krabichler, T, Müller, KJ & Schwarz, M (2023). Deep treasury management for banks. *Frontiers in Artificial Intelligence*, **6**. https://doi.org/10.3389/frai.2023.1120297.

Fontoura, A, Haddad, D & Bezerra, E (2019). A deep reinforcement learning approach to assetliability management. https://doi.org/10.1109/BRACIS.2019.00046.

Fooladi, I & Roberts, G (2000). Risk management with duration analysis. *Managerial Finance*, **26**(18–28). https://doi.org/10.1108/03074350010766558.

Garrett, S (2013). Chapter 9. Term structures and immunization. *In* S Garrett (Ed.). *Introduction to the mathematics of finance*. (Second edition, pp. 177–209). Butterworth-Heinemann. https://doi.org/https://doi.org/10.1016/B978-0-08-098240-3.00009-6.

Geman, H (2023). From Lehman to Silicon Valley Bank and beyond: Why are mistakes repeated in the US banking system? https://www.policycenter.ma/sites/default/files/2023-03/PB 16-23 Helyette.pdf (Accessed on 04/02/2023).

Geron, A (2019). *Hands-on machine learning with Scikit-learn, Keras, and Tensorflow.* 2nd edition. O'Reilly Media, Inc. **40.**

Guignard, F, Amato, F & Kanevski, M (2021). Uncertainty quantification in extreme learning machine: Analytical developments, variance estimates and confidence intervals. *Neurocomputing*, 456, 436–449. https://doi.org/https://doi.org/10.1016/j.neucom.2021.04.027.

Hariom, T, Puri, S & Lookabaugh, B (2020). *Machine learning and data science blueprints for finance*. O'Reilly Media, Inc. https://www.oreilly.com/library/view/machine-learning-and/9781492073048/.

He, T & Droppo, J (2016). Exploiting LSTM structure in deep neural networks for speech recognition. 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 5445–5449. https://doi.org/10.1109/ICASSP.2016.7472718.

Hochreiter, S & Schmidhuber, J (1997). Long short-term memory. *Neural Computation*, **9**(1735–80). https://doi.org/10.1162/neco.1997.9.8.1735.

Hsu, W-N, Zhang, Y, Lee, A & Glass, J (2016). Exploiting depth and highway connections in convolutional recurrent deep neural networks for speech recognition, 395–399. https://doi. org/10.21437/Interspeech.2016-515.

Hua, Y, Zhao, Z, Li, R, Chen, X, Liu, Z & Zhang, H (2019). Deep learning with long short term memory for time series prediction. *IEEE Communications Magazine*, **57**(6), 114–119. https://doi.org/10.1109/MCOM.2019.1800155.

Jarrow, RA (2004). Risky coupon bonds as a portfolio of zero-coupon bonds. *Finance Research Letters*, **1**(2), 100–105. https://doi.org/https://doi.org/10.1016/j.frl.2004.03.003.

Jarrow, RA & Turnbull, SM (2000). The intersection of market and credit risk. *Journal of Banking Finance*, 24(1), 271–299. https://doi.org/https://doi.org/10.1016/S0378-4266(99)00060-6.

Kahlig, J (2022). Duration, convexity and immunisation. https://people.tamu.edu/~kahlig/notes/325/ch11-part-c.pdf (Accessed on 11/20/2022).

Kolm, PN & Ritter, G (2020). Modern perspectives on reinforcement learning in finance. *SSRN*, **1**(18–28). https://ssrn.com/abstract=3449401.

Krabichler, T & Teichmann, J (2023). A case study for unlocking the potential of deep learning in asset-liability-management. *Frontiers in Artificial Intelligence*, **6**. https://doi.org/10.3389/frai.2023.1177702.

Lang, N (2022). An introduction to Tensorflow. Get to know the machine learning, its architecture and the comparison to PyTorch. https://towardsdatascience.com/an-introduction-to-tensorflow-fa5b17051f6b (Accessed on 11/20/2022).

Li, Y (2017). Deep reinforcement learning: An overview. https://doi.org/10.48550/ARXIV.1701.07274.

Li, Y (2022). Introducing deep reinforcement learning. https://medium.com/@yuxili/deeprl-6c8c48b6489b (Accessed on 11/02/2022).

Mallinar, N & Rosset, C (2018). Deep canonically correlated LSTMs. https ://doi . org/10.48550/ARXIV.1801.05407.

Mousavi, S, Schukat, M & Howley, E (2018). Deep reinforcement learning: An overview. Proceedings of SAI Intelligent Systems Conference (IntelliSys) 2016: Volume 2, 426–440.

Naeem, M, Rizvi, STH & Coronato, A (2020). A gentle introduction to reinforcement learning and its application in different fields. *IEEE Access*, **8**, 209320–209344. https://doi.org/10.1109/ACCESS.2020.3038605.

Nieto, A, Juan, AA & Kizys, R (2022). Asset and liability risk management in financial markets. *In* J Pilz, TA Oliveira, K Moder, & CP Kitsos (Eds.), *Mindful topics on risk analysis and design of experiments* (pp. 3–17). Springer International Publishing.

Osiński, B, Jakubowski, A, Ziecina, P, Miloś, P, Galias, C, Homoceanu, S & Michalewski, H (2020). Simulation-based reinforcement learning for real-world autonomous driving. 2020 IEEE International Conference on Robotics and Automation (ICRA), 6411–6418. https://doi.org/10.1109/ICRA40945.2020.9196730.

Pang, B, Nijkamp, E & Wu, YN (2020). Deep learning with Tensorflow: A review. *Journal of Educational and Behavioral Statistics*. **45**(2), 227–248. https://doi.org/10.3102/1076998619872761.

Perla, F, Richman, R, Scognamiglio, S & Wüthrich, MV (2021). Time-series forecasting of mortality rates using deep learning. *Scandinavian Actuarial Journal*, 2021. 7(572–598). https://doi.org/10.1080/03461238.2020.1867232.

Qu, Z, Haghani, P, Weinstein, E & Moreno, P (2017). Syllable-based acoustic modeling with CTCSMBR-LSTM. 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), 173–177. https://doi.org/10.1109/ASRU.2017.8268932.

Rabbani, AG, Grable, JE, O'Neill, B, Lawrence, F & Yao, Z (2021). Financial risk tolerance before and after a stock market shock: Testing the recency bias hypothesis. *Journal of Financial Counseling and Planning*. https://doi.org/10.1891/JFCP-19-0002541.

Redington, FM (1952). Review of the principles of life-office valuations. *Journal of the Institute of Actuaries* (1886–1994), **78**(3), 286–340. http://www.jstor.org/stable/41139015 (Retrieved 13 November 2022).

Richman, R (2022). Mind the gap – Safely incorporating deep learning models into the actuarial toolkit. *British Actuarial Journal*, **27**(e21). https://doi.org/10.1017/S1357321722000162.

Richman, R & Wüthrich, MV (2023). Localglmnet: Interpretable deep learning for tabular data. *Scandinavian Actuarial Journal*. **1**(71–95). https://doi.org/10.1080/03461238.2022.2081816.

Sak, H, Senior, A & Beaufays, F (2014). Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. https://doi.org/10.48550/ARXIV.1402.1128.

Sato, Y (2019). Model-free reinforcement learning for financial portfolios: A brief survey. https://arxiv.org/abs/1904.04973.

Sherstinsky, A (2020). Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena*, **404**, 132306. https://doi.org/https://doi.org/10.1016/j.physd.2019.132306.

Sigaud, O & Stulp, F (2019). Policy search in continuous action domains: An overview. *Neural Networks*, **113**(28–40). https://doi.org/https://doi.org/10.1016/j.neunet.2019.01.011.

Smagulova, K & James, A (2019). A survey on LSTM Memristive Neural Network Architectures and Applications. *The European Physical Journal Special Topics*, **228**. https://doi.org/10.1140/epjst/e2019-900046-x.

Smink, M & van der Meer, RAH (1997). Life insurance asset-liability management: An international survey. The Geneva Papers on Risk and Insurance. *Issues and Practice*, **22**(82), 128–142. http://www.jstor.org/stable/41952310.

Staudemeyer, RC & Morris, ER (2019). Understanding LSTM – A tutorial into long short-term memory recurrent neural networks. https://doi.org/10.48550/ARXIV.1909.09586.

Syed, Z & Bansal, R (2018). Do investors exhibit behavioral biases in investment decision making? A systematic review. *Qualitative Research in Financial Markets*, **10**, 00–00. https://doi.org/10.1108/QRFM-04-2017-0028.

Tensorflow.org (2022). Why TensorFlow. https://www.tensorflow.org/about#:~:text=Robust%20ML%20production%20anywhere&text=Whether%20it's%20on%20servers%2C%20edge,edge%20devices%2C%20use%20TensorFlow%20Lite (Accessed on 11/20/2022).

Ward, D & Zurbruegg, R (2000). Does insurance promote economic growth? Evidence from OECD countries. https://www.jstor.org/stable/253847?seq=1#metadata info tab contents (Accessed: 03.10.2021).

Weil, RL (1973). Macaulay's duration: An appreciation. *The Journal of Business*, **46**(4), 589–592. http://www.jstor.org/stable/2351923 (Retrieved November 14, 2022).

Wu, J (2022). Financial market analysis for duration and modified duration. Proceedings of the 2022 7th International Conference on Financial Innovation and Economic Development (ICFIED 2022), 2637–2641. https://doi.org/https://doi.org/10.2991/aebmr.k.220307.429.

Wu, J, Huang, Z, Hu, Z & Lv, C (2023). Toward human-in-the-loop AI: Enhancing deep reinforcement learning via real-time human guidance for autonomous driving. *Engineering*, **21**, 75–91. https://doi.org/https://doi.org/10.1016/j.eng.2022.05.017.

Wüthrich, M & Merz, M (2023). Statistical foundations of actuarial learning and its applications. Springerlink. https://link.springer.com/chapter/10.1007/978-3-031-12409-9_1 (Accessed on 07/01/2023).