Draft version October 19, 2024
Typeset using LaTeX default style in AASTeX631

Optimal Asset-Liability Management for Insurers

XU YUNPENG, Zhang Jinggong[1]

[1]*Nanyang Technological University, Singapore*

Abstract

Insurers' asset portfolios are crucial to company-growth and liability management. In previous works, under the framework of time-consistent planning, such portfolio optimization problems are usually done by assuming Markovian processes for asset movements, formulating the problem, finding a closed-form solution of the optimal control to the problem under simplistic assumptions, constructing a neural network approximation of the solution under realistic assumptions, and comparing the difference between the neural network output and the output of closed-form solution given some testing data. In this work, we assume the asset portfolio consists of stock and bonds, the liability consists of claim amounts, and they are modeled using VARIMA time series. Optimal asset allocation and dividend payout rates (optimal control laws) of an insurer will be found using a feedforward neural network surrogate, without a closed-form solution. We will then prove that the surrogate serves as a valid approximation of the optimal control law without empirically comparing it to a closed-form solution.

Keywords: Time-Consistent Planning, Dynamic Programming, Neural Network Surrogate

1. INTRODUCTION

Insurers need investment porfolio optimisations in order to manage solvency issues by increasing the portolio value, and to improve company-level P/L figures by distributing dividends. It is important for an insurer to be able to find the optimal portfolio trading strategy and dividend payout strategies.

In this work, superscript denotes the label, and subscript denotes the index set of time. A lower-case term denotes a constant term, unless been specified as a control variable, and an upper-case term (excluding T, B) denotes a random variable, process, or a set. A lower-case bolded term denotes a vector, and an upper-case bolded term denotes a matrix. Vectors are assumed to be vertical, and terms listed horizontally inside brackets constitute a tuple. A subscript with a square bracket denotes the elements in the vector or tuple. Right superscript of a term in brackets denotes the exponent.

For this research, We plan to find the optimal strategy for an insurer that maximizes the utility of periodic dividend payouts and the terminal portfolio value.

2. PROBLEM SETUP

Let $(\Omega, \mathcal{F}, \mathbb{F}, \mathbb{P})$ be a discrete time filtered probability space with a finite discrete index set $N = \{0 < ... < T\}$ as time points when the controls are executed. Assume there exist m number of assets, in which one is risk free, the market is complete and free of arbitrage. Let S_t^1 denote the price of the risk-free asset at time $t \in N$. Let L_t be the dollar claim amount to be paid at t. Let S_t^i, $i = 2, 3, ..., m$ be the risky assets' price at time t. Assume the vector of all risky assets, the risk-free asset, as well as the liability follows a VARIMA(p,d,q) process, where matrix parameters $\mathbf{\Phi}, \mathbf{\Psi}, \mathbf{\Sigma}$ establish the dependence between bond, stock prices and claim amounts, and the $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ introduces the randomness. We assume non-negative asset prices and liabilities, therefore, define

$$\mathbf{x}_t := (S_t^1, S_t^2, \ldots, S_t^m, L_t)^\intercal \text{ where } (\Delta)^d \ln \mathbf{x}_t = \boldsymbol{\mu} + \sum_{j=1}^{p} \mathbf{\Phi}^j \cdot ((\Delta)^d \ln \mathbf{x}_{t-j+1} - \boldsymbol{\mu}) + \sum_{j=1}^{q} \mathbf{\Psi}^j \cdot \boldsymbol{\epsilon}_{t-j} + \boldsymbol{\epsilon}_t.$$

With $\mathcal{F}_0 = \sigma(\{S_0^i \forall i, L_0\})$ and $\mathcal{F}_t = \sigma(\{(\Delta)^d \ln S_t^i \forall i, (\Delta)^d \ln L_t, \epsilon_t, ..., S_0^i \forall i, L_0\})$, the above vector process above is $\mathcal{F}_t$ adapted.

On the construction of the insurer's asset portfolio, let the units of stock i in the portfolio at $t, t \in N$ be h_t^i (assume

no short allowed). Define the portfolio value C_t at time t as below

$$C_t = \sum_{i=1}^{m} h_{t-1}^i \cdot S_t^i - L_t.$$

The previous number of shares is evaluated at the current asset price to arrive at the portfolio value, hence the insurer would experience the random change of asset price and claim amounts from $(\Delta)^d \ln \mathbf{x}_t$.

At each time point t, let the dividend payout rate be u_t^0, capped at a maximum dividend payout ratio d; let the constant premium rate received per time t be p, using the formulation of asset rebalance from the classical Agent's Problem, we construct the rebalance as below

$$\sum_{i=1}^{m} h_t^i \cdot S_t^i = C_t + p - u_t^0 \cdot C_t.$$

For the ease of calculations, we define the proportion of portfolio value of each stock in the portfolio as

$$u_t^i := \frac{h_t^i \cdot S_t^i}{C_t + p - u_t^0 \cdot C_t}, i = 2, 3, ..., m,$$

and the work thereafter treats the above dividend payout rate and proportion as the control variables, i.e. the admissible trading strategy in the below set

$$\mathcal{A}_t := \left\{ (\mathbf{u}_t)_{t=1,...,T-1} : \forall i \in \{1, 2, ..., m\}, u_t^i \in [0, 1 + (d-1)\mathbf{1}_{i=0}], \sum_{i=1}^{m} u_t^i = 1 \right\}$$

Deducting the claims occured at $t+1$, L_{t+1}, we are led to the following dynamics of portfolio value

$$C_{t+1} = \sum_{i=1}^{m} u_t^i \cdot R_{t+1}^i \cdot (C_t + p - u_t^0 \cdot C_t) - L_{t+1}$$

where $R_{t+1}^i = S_{t+1}^i / S_t^i$ is the return rate.

3. VALUE FUNCTION

The aim of the insurer is to maximise the utility from periodic dividend payouts as well as discounted terminal portfolio value at the discount rate of β. Assuming the insurer faces the utility function of the below form

$$U(\cdot) = \frac{1}{\gamma}(\cdot)^\gamma, 0 < \gamma < 1,$$

the second derivative of U is negative and therefore exhibits a risk-averse appetite toward dividend and terminal portfolio value.

Define $\mathbb{Y} = ([0, \infty))^{(m+1)\cdot(d+1)} \times (\mathbb{R})^{(m+1)\cdot(p+q)} \times \mathbb{R}$ as the state space of Y_t, where Y_t is the stochastic state variable, and can be realized as y,

$$y := ((B)^0 \ln \mathbf{x}, ..., (B)^d \ln \mathbf{x}, (B)^0(\Delta)^d \ln \mathbf{x}, ..., (B)^{p-1}(\Delta)^d \ln \mathbf{x}, (B)^0\boldsymbol{\epsilon}, ..., (B)^{q-1}\boldsymbol{\epsilon}, C).$$

The Markov decision problem that the insurer wishes to maximize at each t is formulated as below

$$\mathbb{E}\left(\sum_{s=t}^{T-1} U(u_s^0 \cdot C_s) + \beta^{T-t} \cdot U(C_T) \middle| Y_t = y \right), t \in N.$$

It follows that the value function is of the below form

$$V_t(y) = \sup_{(\mathbf{u}_s)_{s=t,...,T-1} \in \mathcal{A}_t} \mathbb{E}\left(\sum_{s=t}^{T-1} U(u_s^0 \cdot C_s) + \beta^{T-t} \cdot U(C_T) \middle| Y_t = y \right), t \in N.$$

Using the terminal condition at $t = T$, we have

$$V_T(y) = \frac{1}{\gamma} \cdot (C)^\gamma,$$

therefore by constructing the dynamic programming equation, we have

$$V_t(y) = \sup_{\mathbf{u}_t} \left\{ \frac{1}{\gamma} \cdot (u_t^0 \cdot C)^\gamma + \beta \cdot \mathbb{E}\left(V_{t+1}\left(\sum_{i=1}^{m} u_t^i \cdot \tilde{R}^i \cdot (C + p - u_t^0 \cdot C) - \tilde{L}\right) \middle| Y_t = y \right) \right\}, t \leq T-1 \tag{1}$$

where

$$(\tilde{R}^1, \tilde{R}^2, \ldots, \tilde{R}^m)^\intercal = \left((B)^{-1}\mathbf{x}_{[1:m]} \oslash \mathbf{x}_{[1:m]}\right), \tilde{L} = (B)^{-1}\mathbf{x}_{[m+1]}$$

$$(B)^{-1}\mathbf{x} = \exp\left(\sum_{k=0}^{d} \binom{d}{k} (-1)^k (B)^k \ln \mathbf{x} + (B)^{-1}(\Delta)^d \ln \mathbf{x} \right)$$

$$(B)^{-1}(\Delta)^d \ln \mathbf{x} = \boldsymbol{\mu} + \sum_{j=1}^{p} \boldsymbol{\Phi}^j \cdot (B)^{t-j+1}((\Delta)^d \ln \mathbf{x} - \boldsymbol{\mu}) + \sum_{j=1}^{q} \boldsymbol{\Psi}^j \cdot (B)^{t-j}\boldsymbol{\epsilon} + \boldsymbol{\epsilon}$$

B^{-1} here denotes the forward shift operator, as the opposite operation to B^1. Let $\hat{\mathbf{u}}_t$ be the optimal control law, for this problem, it is extremely time consuming to find the closed-form solution of $\hat{\mathbf{u}}_t$ and V_t. And we are led to the neural network surrogate of them.

4. SURROGATE

In this section, the above dynamic programming problem will be solved using a neural network surrogate. The general idea is adopted from [Tao Chen, Mike Ludkovski, and Moritz Voß (19 Dec 2023): On parametric optimal execution and machine learning surrogates, Quantitative Finance, DOI: 10.1080/14697688.2023.2282657]. The aim is to find the neural network that serves as approximations of the mappings of $V_{t+1} : N \times \mathbb{Y} \to \mathbb{R}$, $\forall t \leq T-2$. With the approximated mapping of V_{t+1}, for any sample path of $Y_t = y$, one standing at time t can replace the true V_{t+1} in Equation 1 by the approximated mapping, input y into it, and find the argmax of Equation 1 to find the value of the optimal controls for this sample path at t .

Assuming the future indeed follows the VARIMA model specifications, the first step is to generate n number of bounded training samples of $y \in \mathbb{Y}$, each is denoted as y^j, so that we have the training samples $Y^n \subseteq \mathbb{Y}$ of the state variables,

$$Y^n = (y^1, y^2, ..., y^n),$$

Those bounds are of careful choices that should cover almost all possible values in the next T time points and cannot be impractical (e.g. though liability in $(B)^0 \ln \mathbf{x}$ can go close ∞, if T is chosen to be 5 months, then the bounds must be reasonably large numbers $\ll \infty$). Here we choose the bounds for $y[1 : d+1+p+q]$ by simulating T time steps forward and observe the maximum and minimum values of each state variable. Based on the values of simulations of $y[1 : d+1+p+q]$, the lower bound of C can be determined by multiplying the initial portfolio value $C_0 + \sum_{s=0}^{T} v^s \cdot p$ by the least simulated return rate $\min_{\forall i, \forall t}\{S_t^i / S_t^i\}$ for $T-1$ times, then minus off the largest simulated claim amount $\max_{\forall i, \forall t}\{L_t\}$ $T-1$ times. The upper bounds of C can be determined similarly. Each element in training sample y^j is generated using sobol sequence sampling given that we obtained the bounds of state variables in $\mathbb{Y}$. Sobol sequence sampling provides a more uniform coverage than random sampling, and hence would prevent clustering of training samples in some regions of the input space $\mathbb{Y}$.

The second step is to define the one-step-forward transition of y given a control u and random noise $\boldsymbol{\epsilon}$ that acts similarly to the forward shift operator $(B)^{-1}$:

$$(\tilde{B})^{-1}\big(y, \boldsymbol{\epsilon}, \mathbf{u}\big) := \big((B)^{-1}\mathbf{x}, ..., (B)^{d-2}\mathbf{x}, (B)^{-1}(\Delta)^d\mathbf{x}, ..., (B)^{p-2}(\Delta)^d\mathbf{x}, \boldsymbol{\epsilon}, ..., (B)^{q-2}\boldsymbol{\epsilon},$$
$$\sum_{i=1}^{m} u^i \cdot \tilde{R}^i \cdot (C + p - u^0 \cdot C) - \tilde{L}\big)$$

The next step is to find the expectation component of the optimal value function numerically, we apply a quantization

approach to do so. Given a level of quantization, let $\boldsymbol{w}$ denotes the weights vector of occurring $\boldsymbol{e}$ as a quantization of $\boldsymbol{\epsilon}$, we define $\mathring{V}_t(y)$ such that $\mathring{V}_t \simeq V_t$,

$$\mathring{V}_t(y) = \sup_{\forall \mathbf{u}_t} \left\{ \frac{1}{\gamma} \cdot (u_t^0 \cdot C)^\gamma + \beta \cdot \boldsymbol{w}^\intercal \cdot \mathring{V}_{t+1}\big((\tilde{B})^{-1}(y, \boldsymbol{e_t}, \mathbf{u}_t)_{[-1]}\big) \right\}, \text{ whenever } Y_t = y$$

$$\mathring{V}_T(y) = \frac{1}{\gamma} \cdot (C)^\gamma = V_T(y)$$

The final step is to find the approximation of $\mathring{V}_t(y)$ using backward iteration. Denote the neural network surrogate of the closed-form solution of V_t as $\mathring{V}^{\theta_t}$ where θ_t is the initial neural network weights and biases. Then, define

$$v_t(y, \mathbf{u}) = \begin{cases} \frac{1}{\gamma} \cdot (\mathbf{u}_{[1]} \cdot C)^\gamma + \beta \cdot \boldsymbol{w}^\intercal \cdot \mathring{V}_{t+1}\big((\tilde{B})^{-1}(y, \boldsymbol{e_t}, \mathbf{u}_{[2:m+1]})\big), t = T-1, \\ \frac{1}{\gamma} \cdot (\mathbf{u}_{[1]} \cdot C)^\gamma + \beta \cdot \boldsymbol{w}^\intercal \cdot \mathring{V}^{\hat{\theta}_{t+1}}\big((\tilde{B})^{-1}(y, \boldsymbol{e_t}, \mathbf{u}_{[2:m+1]})\big), t < T-1, \end{cases}$$

where $\hat{\theta}$ will be introduced shortly. Given the closed-from expression of V_T, from $T-1$ to 1, for each y^j, we find the largest value of the function $v_t(y^j, \mathbf{u})$ using scipy.optimize, denoted as v_t^j. Then we train the optimal weights and biases $\hat{\theta}_t$ using those n number of samples of v_t^j corresponding to y^j such that the MSE is minimised, so the mapping of $\mathring{V}_t$ can be approximated and used in the next backward iteration.

Here, before the training of $\mathring{V}^{\theta_t}$, we allow it to be a function of y and $\mathring{V}^{\hat{\theta}_{t+1}}$; whereas after the training, the θ_t that minimises the MSE, i.e. the $\hat{\theta}_t$ is found, thus $\hat{\theta}_t$ is only a function of y. The entire procedure is summarised as below,

Algorithm 1 Construct surrogate

Generate Y^n that fills n locations in $\mathbb{Y}$.
Initialise `V_train` as an empty $(T-1) \times n$ space.
for $t = T-1, T-2, \ldots, 1$ **do**
  for $j = 1, 2, \ldots, n$ **do**
    Optimise $v_t(y^j, u)$ and find $\hat{v}_t^j$ using scipy.optimize.
    `V_train[t][j]` $= v_t^j$
  end for
  Use `V_train[t]` and Y^n to fit a mapping $\mathring{V}^{\hat{\theta}_t}$ as an approximation of V_t
end for

The construction of the above neural network is documented in (https://github.com/yyunpeng/NNsurrogate2). It adopts the idea of [Tao Chen, Mike Ludkovski, and Moritz Voß (19 Dec 2023): On parametric optimal execution and machine learning surrogates, Quantitative Finance, DOI: 10.1080/14697688.2023.2282657]. However, here we do not have the approximation of the optimal control as a function of state variables, and the value function will be directly used to find the value of optimal control, not as functions, for each sample path. In this way, there is one less round of approximation, and for multiple output neurons, the MSE tends to be high for training than single output neuron in the case of approximating value function alone.

5. EXPERIMENT

In this section, we demonstrate a numerical experiment of the dynamics of the insurer's asset portfolio and dividend payout given the surrogates as described in Section 4. The baseline assumptions of the experiment are as follows:

1. The life insurer is evaluating its asset portfolio strategy as of 2023-03-01, time units are in months, and uses the monthly claim and monthly close price of assets in the past 2 years to fit a VARIMA(2,1,0) model.

2. The life insurer policy portfolio includes 1% of monthly provisional counts of deaths in the U.S.(https://data.cdc.gov/NCHS/Monthly-Provisional-Counts-of-Deaths-by-Select-Cau/9dzk-mvmi) based on the date range as specified in 1.. Sum assured is $10,000$ USD for every death and will be paid immediately at the end of the month of death. The sum assured is adjusted by monthly inflation rate (https://www.investing.com/economic-calendar/cpi-69).

3. Assets to be traded are BND (S^1), PFE(S^2) and REM(S^3), corresponding the bonds, stocks and mortgages that typically appear in an insurer's asset portfolio. Data are obtained from yahoo finance library `yfinance` based on the date range as specified in 1..

4. $m = 3$, $T = 5$ in months, constant premium rate $P = 12,000$ USD (i.e. with risk loading of 1.2). d, γ and v are constant throughout, initial capital value $C_0 = 10,000,000$ USD.

5. quantization level is 1,000 to numerically compute the expectation component of the optimal value function. Uniform quantization is adopted, therefore $\boldsymbol{w}^\intercal = (\frac{1}{1000}, ..., \frac{1}{1000})$.

We adopted serveral benchmarking strategies to evaluate the performance of the neural network strategy:

1. The portfolio value is fully allocated to S^1 for every time step.

2. The portfolio value is fully allocated to S^2 for every time step.

3. The portfolio value is fully allocated to S^3 for every time step.

4. The portfolio value is allocated to S^1, S^2, S^3 at the proportion of $\frac{1}{3}, \frac{1}{3}, \frac{1}{3}$ for every time step.

For all the above benchmarking strategies, the portfolio dynamic is the same as in Section 2, only the definition of $u_t^i, i = 1, ..., m$ changes. Dividend payout strategy will be the same as the NN strategy. Testing is conducted by constructing 1000 testing paths of the state variables, and apply $\mathring{\mathbf{u}}_{\mathtt{stk}}^{\hat{\vartheta}_t^1}, \mathring{u}_{\mathtt{dvd}}^{\hat{\vartheta}_t^2}$, and the benchmarking strategies to see which strategy yields the most utility. Below is the boxplot of the utility of terminal portfolio value C_T for each of the strategies during testing, where $\gamma = 0.9, v = 0.8$, and there exist 1,000 (testing) simulated paths for the future 5 months. Unites are in thousands for portfolio value.

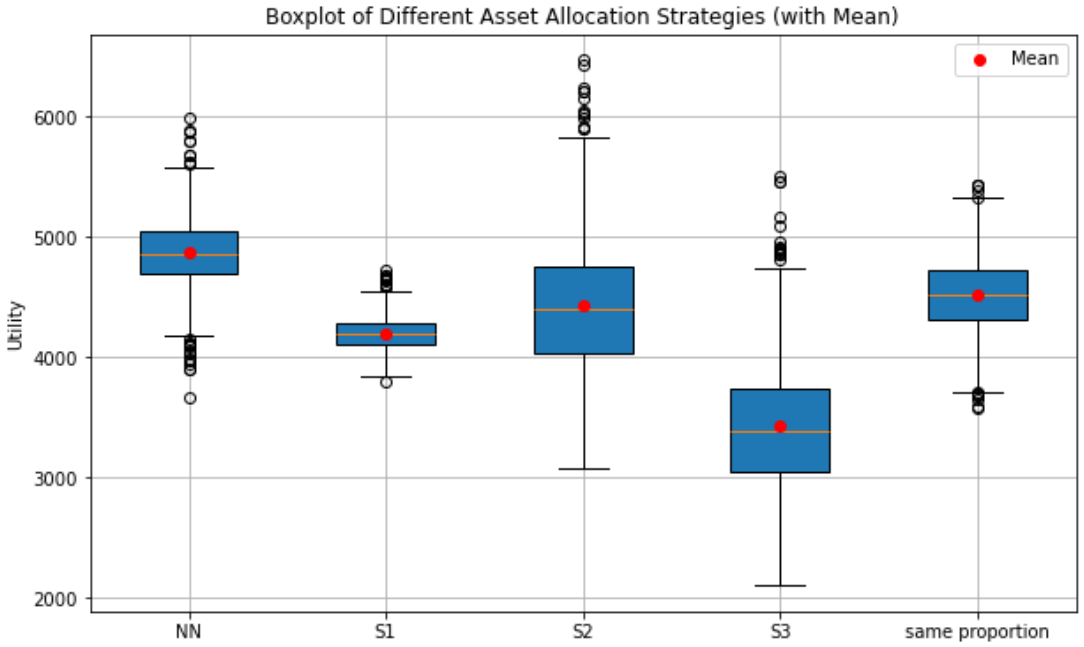

Figure 1.

It can be seen that the average utility of the testing sample is the highest when our neural network surrogate of $\mathring{\mathbf{u}}_{\mathtt{stk}}^{\hat{\vartheta}_t^1}$ is adopted. Furthermore, when looking at the distribution of the utility and portfolio value in the testing sample,

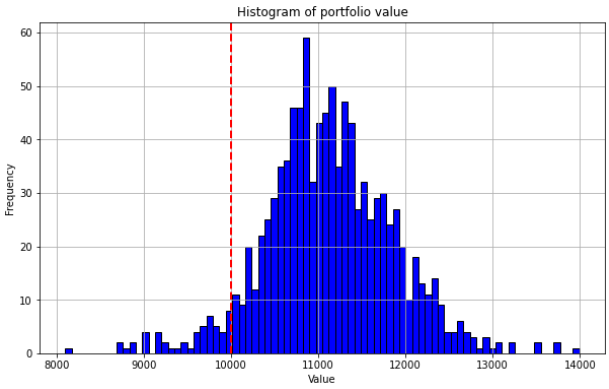

Figure 2.

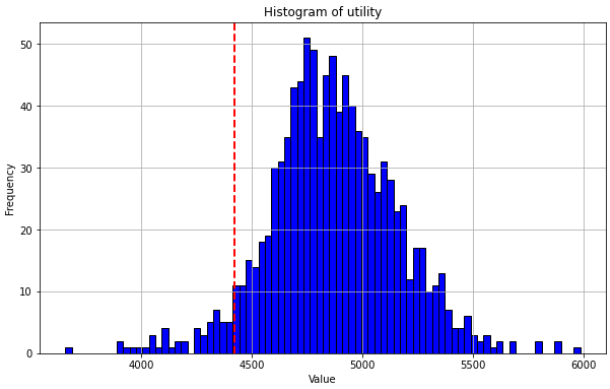

Figure 3.

where the horizontal red dotted line denotes the value and the utility of the initial capital, we observe some normality in distribution.