Efficient Machine Learning Methods for Risk Management of Large Variable Annuity Portfolios *

Wei Xu[1] , Yuehuan Chen[2] , Conrad Coleman[3] and Thomas F. Coleman[3] †

Abstract

Variable annuity (VA) embedded guarantees have rapidly grown in popularity around the world in recent years. Valuation of VAs has been studied extensively in past decades. However, most of these studies focus on a single contract. These methods cannot be extended to valuate a large variable annuity portfolio due to the computational complexity. In this paper, we propose an efficient moment matching machine learning method to compute the annual dollar deltas, VaRs and CVaRs for a large variable annuity portfolio whose contracts are over a period of 25 years. There are two stages for our method. First, we select a small number of contracts and propose a moment matching Monte Carlo method based on the Johnson curve, rather than the well known nested simulations, to compute the annual dollar deltas, VaRs and CVaRs for each selected contract. Then, these computed results are used as a training set for well known machine learning methods, such as regression tree , neural network and so on. Afterwards, the annual dollar deltas, VaRs and CVaRs for the entire portfolio *are predicted* through the trained machine learning method. Compared to other existing methods (Bauer et al., Gan, Gan and Lin, 2008, 2013, 2015), our method is very efficient and accurate, especially for the first 10 years from the initial time. Finally, our test results support our claims.

Key words: Variable Annuity, Johnson Curve, Machine Learning, Dollar Delta, VaR

1 Introduction

Variable annuities (VAs) were introduce first in 1970s in United States (Sloane, 1970). They are deferred annuities that are fund-linked during the deferment period. Beginning in the 1990s, certain guarantees were included in these policies by insurers, such as guaranteed minimal death benefit (GMDB), guaranteed minimal accumulation benefit

*This work was supported by the Global Risk Institute, Toronto, Canada, www.globalriskinstitute.org; Cayuga Research Inc. www.cayugaresearch.com

†1 Department of Mathematics, Tongji University, Shanghai; China, 200092;Waterloo Research Institute in Insurance, Securities, and Quantitative Finance, University of Waterloo. [2,3] Cayuga Research Inc., Waterloo, On, Canada; [4] Waterloo Research Institute in Insurance, Securities, and Quantitative Finance, University of Waterloo. [1] wdxu@tongji.edu.cn; [2] y225chen@cayugaresearch.com;[3] conrad@cayugaresearch.com; [4] tfcoleman@uwaterloo.ca.

(GMAB), guaranteed minimal income benefit (GMIB) and guaranteed minimal withdrawal benefit (GMWB), as riders. The GMDB guarantees that the beneficiary of a VA holder receives the greater of the sub-account value of the total purchase payments, upon the death of the VA holder. The GMAB and GMIB provide accumulation and income protection for a fixed number of year contingent on survival of the VA holder, respectively. The most popular riders among these is the GMWB, which guarantees a specified amount for withdrawals during the life of the contract as long as both the amount that is withdrawn within the policy year and the total amount that is withdrawn over the term of the policy stay within certain limits. For details about these guarantees, the reader can refer to (Bauer et al., Belanger et al., Huang et al., Ng and Li, 2008, 2009, 2014, 2013).

Due to new innovation guarantee schemes, VAs have grown rapidly in popularity around world in past decades. In 2013 and 2014, the sales of VAs in US are 145 and 140 billion dollars, respectively, according to LIMRA [1]. As a result, almost every insurance company is managing very large VA portfolios. The embedded guarantees in VAs are posing a significant financial risk to insures. Thus, risk management of such policies is a crucial issue to these VA providers (Bauer et al., 2012).

The academic literature on valuing and hedging guarantees in VA contracts is extensive. Armstrong (Amrstrong, 2001) studied the guarantee reset features in segregated funds using a discrete time Markov chain model. Boyle *et. al* (Boyle et al., 2001) proposed a Monte Carlo method to value the reset options embedded in some segregated funds. Milevsky and Promislow (Milevsky and Promislow, 2001) used risk-neutral option pricing theory to price GMDB in a VA contract. Coleman *et. al* (Coleman et al., 2006) used local risk minimization to study the discrete hedging of the guarantees embedded in a VA contract with both equity risk and interest rate risk. They concluded that hedging with standard options is better than hedging with the underlying asset. Boyle and Tian (Boyle and Tian, 2008) analyzed the design of general equity-indexed annuity from the investor's perspective and proposed a generalization of the conventional design. Lin et al.(Lin et al., 2009) used the Esscher transform to determine an equivalent martingale measure for the fair valuation of a VA contract under a regime-switching model in the incomplete market setting. Xu and Wang (Xu and Wang, 2009) proposed a model based on a two-dimensional partial differential equation to price the GMWB rider. Gao and Ulm (Gan and Ulm, 2012) studied the valuation of the GMDB rider using a utility-based approach. Yang and Dai (Yang and Dai, 2013) proposed a tree model to price the GMDB rider embedded in deferred life annuity contracts.

Unfortunately, existing valuation/hedging methods used in the risk management of an individual VA contract cannot be feasibly extended to a large VA portfolio. There are two reasons for this. One reason is the complexity of the payoff function does not lead to closed-form formulas to evaluate the liability of guarantees. The other reason is when

[1] http://www.limra.com/uploadedFiles/limra.com/LIMRA_Root/Posts/PR/Data_Bank/_PDF/2005-2014-Annuities-Estimates-Alternative.pdf

the contract number is large, valuation and sensitivity calculation is computationally challenging. In fact, determining how to hedge the risk of a large VA portfolio and determining the corresponding required capital poses a significant computational challenge to insurance companies (Bauer et al., 2008). In practice, insurance companies typically follow a market-to-model approach and rely heavily on simulations. Nested simulations (NS) are used to determine the probability distribution of loss from mismatching in order to calculate required capital (Reynolds and Man, 2008). There are two levels for the NS procedure. The first level, called the outer loop, projects the VA liabilities onto real world scenarios. At each node of an outer loop, the second level, the inner loop, projects the liability of a VA contract onto a large number of simulated risk-neutral paths. However, a significant concern with NS is its heavy computational cost. For example, if we use 1000 paths for 30 years at each annual node of a 30-year VA contract over 1000 scenarios, then we end with a computational problem with $1000 \times 30 \times 1000 = 30$ million scenarios. If the portfolio consists of 100,000 contracts, it means 3×10^{13} scenarios. This represents a lot of computing!

Effective ways to reduce the computational time are to reduce either the number of outer loop scenarios or the number of inner loop paths. However, because the Monte Carlo method heavily depends on the number of simulations: too few outer loop scenarios or inner loop paths would result in inaccuracy in computation. One approach to reduce the heavy computing demands in nested simulations is to replicate portfolios. Daul and Vidal (Daul and Vidal, 2009) studied the quadratic programming method to replicate cash flows of life insurance liabilities in general. Dembo and Rosen (Dembo and Rosen, 1999) presented a portfolio replication framework that minimizes the sum of absolute differences instead of the sum of squared differences. However, both methods are still quite expensive when used to replicate a large portfolio.

Another approach is to regress the liability value on some key economic factors (Cathcart and Morrison, 2009). Then, the least square Monte Carlo method is employed to approximate the future liability at each time step. This approach can significantly reduce the number of inner paths in nested simulations, but effectively determining the basis functions in the least square Monte Carlo (Longstaff and Schwartz, 2001) is not trivial in real applications.

Gan (Gan, 2013) proposed a method based on k-prototypes data clustering combined with the ordinary kriging method to price guarantees for a large portfolio of VAs. The speedup of this method is significant, but it is only developed for one level of simulations, rather than nested simulations. In 2015, Gan and Lin (Gan and Lin, 2015) extended the data clustering method to compute annual dollar deltas of a large VA portfolio, which does require nested simulations. In their method, k representative contracts are selected whose dollar deltas are computed by nested simulations. Then, annual dollar deltas for the reminder of the other VA contracts in the portfolio is determined by universal kriging for function data (UKFD) method. In (Gan and Lin, 2015), only 5 outer loop scenarios are considered in the nested simulations. However, significant computational time is required when 1,000 outer loop scenarios are considered to compute the annual dollar delta of a large VA portfolio because annual dollar deltas computing for k representative contracts is still expensive through the nested simulations (even though k is relatively small compared to the size of the portfolio).

In this paper, we first propose a moment matching scenario selection method based on the Johnson curve (Johnson, Xu and Yin, 1949, 2014). In this approach, a small number of well selected real world scenarios are generated with a near-unity confidence level to calculate annual dollar deltas, VaRs and CVaRs for an individual VA contract. Compared to nested simulations, our moment matching method is about 150 times faster. Without loss of generality, we assume the outer loop scenarios follows a two regimes switching lognormal model. In fact, our moment matching method can also be applicable to a jump diffusion model (Merton, 1976) and those from the generalized autoregressive heteroskedasticity (GARCH) family (Das and Sundaram, Duan et al., Duan et al., Heston and Nandi, Mazzoni, 1999, 1999, 2006, 2000, 2010). Then, we employ classical machine learning methods, such as neural network, regression tree and random forest, to predict the annual dollar deltas, VaRs and CVaRs for a large VA portfolio. The main advantages of our moment matching machine learning method are efficiency and accuracy using a small number of representative contracts. Our approach is about 40 times faster than the UKFD method (Gan and Lin, 2015) and about 150 times faster than nested simulation.

The rest of this paper is organized as follows. The moment matching method is proposed in Section 2. Then, some classical machine learning methods are introduced in Section 3. The framework for our moment matching machine learning (MMML) approach is proposed in subsection 3.2. After that, we compare our methods with nested simulations and UKFD method (Gan and Lin, 2015) on various sizes of VA portfolios in Section 4. The numerical results demonstrate the efficiency and accuracy of our approach. Finally, we conclude with some remarks in Section 5.

2 Efficient valuation of VA

Nested simulation is an important tool for risk management. In particular, it is used to determine the probability distribution of loss from mismatching (Fox, 2013), which is widely used to compute Greeks, VaR and CVaR (Fox, Reynolds and Man, 2013, 2008). In nested simulations, there are usually two loops of simulations, the outer loop and inner loop (Fox, Reynolds and Man, 2013, 2008). The outer loop projects the VA liabilities along real world scenarios, for example using two regimes switching lognormal model (Hardy, 2001). Then, at each node of an outer loop scenario, the liabilities are estimated along a large number of risk neutral simulations: this is the inner loop. Direct nested simulation is not the only computational challenge that insurers face. Many other computations pose a similar challenges, particular those which are stochastic involve large portfolios or require frequent updates. In this section, we propose a moment matching technique based on the Johnson curve (Johnson, 1949) to replace the outer loop scenario simulations so that the real world scenarios are well selected and have a near-unity confidence level. Without loss of generality, we assume the outer real world scenarios are generated by a two regimes switching lognormal model (Hardy, 2001) while the inner loop is simulated by the geometric Brownian motion (Hull, 2006). In fact, our method can also be applied to other real world models: this will be discussed in Subsection 2.4.

2.1 Generating the Real World Scenario

Assume the value of the hedge fund linked to a VA contract is modeled as a two regimes switching lognormal model, i.e., the investment return in the real world is modeled by the two regimes switching lognormal model (Hardy, 2001). Hence, we assume the hedge fund return lies in one of two regimes. Consider a time interval $[0,\ T)$, where T is the expiry of a VA. We divide $[0,\ T)$ into N subintervals evenly, then the hedge fund value R_n represents the hedge fund value at time $n\Delta t$, where $\Delta t = T/N$ and $n = 0, 1 \cdots, N$. Let ρ_n denote the regime applying in the interval $[n\Delta t,\ (n+1)\Delta t)$, $\rho_n = 1, 2$ and R_n be the hedge fund value at time t, then

$$\log \frac{R_{n+1}}{R_n} \mid \rho_n \sim N(\mu_{\rho_n},\ \sigma^2_{\rho_n}).$$

In other words, given the regime ρ_n, the hedge fund return $\log \frac{R_{n+1}}{R_n}$ is a normal distribution with expectation μ_{ρ_t} and variance $\sigma^2_{\rho_t}$. The transition probabilities of moving regimes are defined as

$$p_{ij} = \Pr[\rho_{n+1} = j \mid \rho_n = i], \qquad i = 1, 2; j = 1, 2.$$

Thus, a two regimes independent lognormal model is determined by six parameters $\Theta = \{\mu_1, \mu_2, \sigma_1, \sigma_2, \rho_{12}, \rho_{21}\}$.

It follows that the unconditional moments of the hedge fund value at any time $n\Delta t$ can be evaluated as

$$\mathrm{E}[(R_n)^k] = \exp\left(kn\mu_2 + \frac{k^2}{2} n\sigma_2^2\right) \times \sum_{i=0}^{n} \exp\left(i\left(k(\mu_1 - \mu_2) + \frac{k^2}{2}(\sigma_1^2 - \sigma_2^2)\right)\right) p(i), \tag{2.1}$$

where $p(i)$ is the probability of the total sojourn in regime-1 in interval $[0,\ n\Delta t)$ equal to i. Then, given the first four moments of R_n, the discrete values of the random variable R_n can be generated via the Johnson curve proposed in (Johnson, Xu and Yin, 1949, 2014).

In theory, the Johnson curve can convert any continuous random variable R_n into a standard normally distributed random variable Z based on the first four moments of

R_n. On the other hand, based on the Johnson curve inverse transformation, a standard normally distributed variable Z can be converted into a random variable R_n, i.e.,

$$R_n = c + d \cdot g^{-1}\left(\frac{Z-a}{b}\right), \tag{2.2}$$

where parameters $a,\ b,\ c,\ d$ and function g are determined to satisfy the first four moments of R_n. Details of determination are given in Appendix A.

In order to generate the discrete values of R_n, we first construct m pairs $\{(z_i,\ q_i)\}$ for $i = 1, 2, \cdots, m$, where z_i is the discrete values of the standard normal distribution and q_i is the probability $\Pr(Z_{i-1} \le z_i \le Z_i),\ i = 1, 2, \cdots, m+1,\ Z_0 = -\infty,\ Z_m = +\infty$ and $Z_i = N^{-1}(\sum_{j=1}^{i} q_j)$. There are four criteria for sampling $\{(z_i,\ q_i)\}$ (Xu et al., 2013).

1. The expectation of $\{z_i\}$ is zero.
2. The variance of $\{z_i\}$ is one.
3. The kurtosis of $\{z_i\}$ is close to three.
4. The absolute values of z_1 and z_m should be as large as possible.

Based on these four criteria, Xu et al (Xu et al., 2013) proposed a sequence,$\{q_i\}$,

$$q_i = \tilde{q}_i / \sum_{i=1}^{m} \tilde{q}_i \qquad \text{and} \qquad \tilde{q}_i = (i-0.5)^{0.6}/m,\ i = 1, 2, \cdots, m/2.$$

The corresponding sequence, $\{z_i\}$, is the solution of the following nonlinear least squares problems

$$\min_{z_i} [\sum_{i=1}^{m} q_i z_i^4 - 3]^2 \tag{2.3}$$

$$\begin{aligned} \text{s.t.}\quad & \textstyle\sum_{i=1}^{m} q_i z_i = 0 \\ & \textstyle\sum_{i=1}^{m} q_i z_i^2 = 1 \\ & Z_{i-1} \le z_i \le Z_i, \end{aligned}$$

where $Z_i = N^{-1}(\sum_{j=1}^{i} q_j)$, for $i = 1, 2, \cdots, m-1$, $Z_0 = -\infty$ and $Z_m = \infty$.

Table 1 lists the pairs $\{(z_i,\ q_i)\}$ for $m = 30$ and $m = 50$ computed by Xu's strategy. Due to the symmetry of the $\{(z_i,\ q_i)\}$, we only record first 15 and 25 values, respectively, while the other half can be mirrored. It shows that when $m = 30$, the confidence level of $\{(z_i, q_i)\}$ is more than 99% for the standard normal distribution while the confidence level is 99.8% for $m = 50$. Then, using (2.2), discrete values R_n^i can be generated from discrete values $\{z_i\}$ whose corresponding probability $\Pr(\tilde{R}_n^{i-1} \le R_n^i \le \tilde{R}_n^i)$ is q_i where $\tilde{R}_n^i$ is transformed value of Z_i via the Johnson curve. In other words, discrete value R_n^i represents all scenarios in the interval of $[\tilde{R}_n^{i-1},\ \tilde{R}_n^i]$ whose corresponding probability is q_i. Now, we are ready to generate m real world scenarios at time $n\Delta t$ through discrete pairs $\{(z_i,\ q_i)\}$ and Johnson curve inverse transformation.

Conference Copy

i	$m = 30$		$m = 50$	
	z_i	q_i	z_i	q_i
1	2.8821	0.0069	3.137	0.003
2	2.2081	0.0134	2.5112	0.006
3	1.8896	0.0182	2.2265	0.008
4	1.6486	0.0222	2.0156	0.0098
5	1.4491	0.0259	1.8445	0.0114
6	1.2749	0.0292	1.6984	0.0129
7	1.1175	0.0323	1.5693	0.0143
8	0.9717	0.0351	1.4525	0.0155
9	0.8341	0.0379	1.345	0.0167
10	0.7020	0.0405	1.2446	0.0179
11	0.5737	0.0430	1.1499	0.019
12	0.4474	0.0454	1.0597	0.0201
13	0.3216	0.0478	0.9732	0.0211
14	0.1948	0.0500	0.8897	0.0221
15	0.0655	0.0522	0.8085	0.0231
16			0.7292	0.024
17			0.6514	0.0249
18			0.5746	0.0258
19			0.4986	0.0267
20			0.4229	0.0276
21			0.3474	0.0284
22			0.2716	0.0292
23			0.1953	0.03
24			0.1181	0.0308
25			0.0396	0.0316

Table 1: List of $\{z_i, q_i\}$ from Xu's strategy with $m = 30$ and $m = 50$.

Algorithm 1 *(Real World Scenario Generation.) Assume hedge fund value follows a two regime switching lognormal model in (Hardy, 2001) with* $\Theta = \{\mu_1, \mu_2, \sigma_1, \sigma_2, \rho_{12}, \rho_{21}\}$. *All possible values of hedge fund,* R_n^i, $n = 1, 2, \cdots, N$, $i = 1, \cdots, m$, *can be computed as follows with initial value* $R_0 = 1$.

1. *Following the sampling strategy in (Xu et al., 2013), to generate* m *pairs* $\{(z_i,\ q_i)\}$ *for standard normal distribution.*

2. **for** $n = 1, 2, \cdots, N$

   (a) *Compute the expectation, variance, skew and kurtosis of* R_n *from (2.1).*

   (b) *Determine the parameters* a, b, c, d *and function* g *via the algorithm in (Hill et al., 1976) based on the computed first four moments from (a).*

   (c) *Generate pairs* $\{(R_n^i, q_i)\}$ *from (2.2) with discrete values* $\{z_i\}$.

   end

The outputs of Algorithm 1 are $m \times N$ possible scenarios of the hedge fund in the period N years. For example, we consider 1000 real scenarios for a 30-year VA in the nested simulation method. There are 30,000 possible scenarios for the hedge fund. However, due to our moment matching technique, we just need 900 possible values with confidence level more than 99% as $m = 30$, that is we only need consider 30: that is, scenarios for each year to cover 99% possible scenarios.

Another remark regarding to Algorithm 1 is that our scenario generation is path independent. For each time $n\Delta t$, we generate possible scenarios of the hedge fund at that time, independent of the path from initial time to scenarios at time $n\Delta t$. When some specified percentile VaR is required at a certain time horizon, all computations are only carried on possible scenarios with corresponding percentage at that time, not necessarily all paths from the initial time. In the next subsection, considering given the hedge fund scenario R_n^i, how can the liability of a VA contract be evaluated by the inner simulation.

2.2 VA Liability valuation

Variable annuities are equity-linked annuity products with some generated minimum benefits. Among these benefits, the guaranteed minimum withdrawal benefit (GMWB) is the most popular one. In this subsection, given the initial account value A_0, we present how to price the liability of a VA contract with the hedge fund scenario R_n^i at time $n\Delta t$. First, assume that the policyholders withdraw C_n at time $n\Delta t$. Then, the account value A_n^i at time $n\Delta t$ with the hedge fund value R_n^i can be computed in following process. At beginning, the account value is A_0; at the end of year one, i.e., $n = 1$, $A_1^i = \max\{A_0 \cdot R_1^i - C_1,\ 0\}$ for $R_0 = 1$. Now, consider the case for $n \geq 2$. Since the initial value for the fund is one, the average yield of the hedge fund from initial time to $n\Delta t$ with R_n^i can be evaluated as $\sqrt[n]{R_n^i}$. The future values of the withdraws before

$n\Delta t$ can be evaluated as

$$C_1 \cdot (R_n^i)^{\frac{n-1}{n}} + C_2 \cdot (R_n^i)^{\frac{n-2}{n}} + \cdots + C_{n-1} \cdot (R_n^i)^{\frac{1}{n}}.$$

Thus, the account value A_n^i at $n\Delta t$ can be computed as

$$A_n^i = \max\{A_0 R_n^i - \sum_{j=0}^{n-1} C_{n-j} (R_n^i)^{\frac{j}{n}},\ 0\}.$$

After we obtain the account values of a VA contract under each real scenario at time $n\Delta t$, the inner simulation is employed to price the VA with account value A_n^i at time $n\Delta t$. In other words, at time $n\Delta t$, we need to compute the value of guarantees using risk neutral paths along the outer scenario i. For example, at time $n\Delta t$, with account value A_n^i, the liability of a VA contract is valuated through the risk neutral paths (i.e., geometric Brownian motion) from $(n+1)\Delta t$ to T. For $k \equiv n,\ (n+1),\ (n+2), \cdots,\ (N-1)$, the evolution of the state variables between k^+ and $(k+1)^-$ are considered. Assume the risk neutral path is simulated as a geometric Brownian motion, i.e.,

$$S_0 = 1, \qquad S_n = S_{n-1} \exp\left(\left[r - \frac{1}{2}\sigma^2\right] + \sigma Z\right),$$

for $n = 1, 2, \cdots, N$ where r is the interest rate, σ is the volatility of the underlying hedge fund, and Z is a standard normal random variable. Then the account value on this risk neutral path, starting at A_n^i is evaluated as

$$\tilde{A}_{k+1}^- = \tilde{A}_k^+ \frac{S_{k+1}}{S_k},$$

where $k = n, \cdots, N-1$ and $\tilde{A}_n^+ = A_n^i$. The liability of the guaranteed minimum benefits can then be evaluated by the Monte Carlo method in (Bauer et al., 2008) as an inner simulation. The algorithm to evaluate a VA contract on real scenarios R_n^i, $n = 1, \cdots, N$ and $i = 1, \cdots, m$ can be described as follows. Details considering the Monte Carlo(MC) method can be obtained in (Bauer et al., 2008).

Algorithm 2 *(Valuate a VA contract) Given real world scenarios R_n^i, $n = 1, \cdots, N$ and $i = 1, \cdots, m$ and initial account value A_0.*

1. *Generate a risk neutral path for geometric Brownian motion with $S_0 = 1$.*

$$S_n = S_{n-1} \exp\left(\left[r - \frac{1}{2}\sigma^2\right] + \sigma Z\right),\ \textit{for } n = 1, \cdots, N.$$

2. **for** $n = 1, \cdots, N$
    for $i = 1, \cdots, m$
        2.1 $A_n^i = \max\{A_0 R_n^i - \cdot \sum_{j=0}^{n-1} C_{n-j} (R_n^i)^{\frac{j}{n}},\ 0\}$.
        2.2 Evaluate the liability of the VA contract, V_n^i, with the initial account

value A_n^i over $N-n$ years horizon by the MC method in (Bauer et al., 2008).

end

end

3. Return V_n^i for all real world scenarios.

2.3 Computing dollar delta, VaR and CVaR

Dollar delta measures the sensitivity of the liability of VA with respect to the change of hedge fund values. As shown in Algorithm 2, the liability of VA is a function of account value A and time t, i.e., $V(A,t)$. In other words, given time $n\Delta t$ and account value A_n^i, the value of VA is $V_n^i \equiv V(A_n^i,\ n\Delta t)$. Thus, the corresponding dollar delta at $n\Delta t$ with account value A_n^i can be evaluated by the finite difference method as

$$\delta_n^i \equiv \delta(A_n^i,\ n\Delta t) \approx \frac{V(A_n^i+\epsilon,\ n\Delta t) - V(A_n^i,\ n\Delta t)}{\epsilon},$$

where $V(A_n^i,\ n\Delta t)$ and $V(A_n^i+\epsilon,\ n\Delta t)$ are computed via Algorithm 2. The expectation of the dollar delta for the VA contract at time $n\Delta t$ can be evaluated as

$$\bar{\delta}_n = \sum_{i=1}^{m} q_i \delta_n^i,$$

where q_i is the corresponding probability in pairs $\{(R_n^i,\ q_i)\}$. For a portfolio with a large number of VA contracts, its dollar delta at time $n\Delta t$, is the summation of $\bar{\delta}_n$ for all VA contracts.

Another application of our method is to compute VaR for the portfolio. VaR of a portfolio is defined as

$$\Pr(r_h < \mathrm{VaR}) = p,$$

where r_h is the liability of the portfolio over h years horizon, that is there is $1-p$ probability that the liability over h year horizon is less than the value of VaR. In other words, VaR is the value of r_h at the $1-p$ percentile. For a VA portfolio, its liability at time $n\Delta t$ highly depends on the linked hedge fund scenario. Thus, based on the discrete pair $\{(R_n^i,\ q_i)\}$, the specified value R_n^i with the probability $1-p$ can be determined. Then, the liability of the portfolio under this scenario at time $n\Delta t$ can be computed via the inner simulation, which is the approximation of corresponding $1-p$ percentile VaR at time $n\Delta t$. Another important risk measurement for the portfolio is conditional VaR (CVaR) or expected shortfall, which is derived by taking a weighted average between the VaR and loss exceeding the VaR, i.e.,

$$\mathrm{CVaR}_\alpha = \frac{1}{\alpha}\int_0^\alpha \mathrm{VaR}_\gamma(X)d\gamma,$$

where X is the payoff of a portfolio at some future time and $0 < \alpha < 1$. Thus, based on the definition, it can be estimated by scenarios in the tail less than $1 - p$ percentile.

For extremely high probability VaR, say 99.5% VaR, and CVaR, we can increase the number of samples, m, to obtain enough sample values at the tail of the distribution. For example, when $m = 100$, there are three sample points at the 1% tail while there are six samples when $m = 200$. As the VaR and CVaR only consider the scenarios at the tail, its computational cost is still low even for a large m. As mentioned in Subsection 2.1, each point R_n^i does not represent a single scenario, it represents all scenarios between $[\tilde{R}_n^{i-1},\ \tilde{R}_n^i]$, whose corresponding probability is q_i. In other words, each point R_n^i in the tail is selected to best represent all scenarios in the interval. Thus, in our numerical experiments, the moment matching method works well to estimate 95% and 99% VaRs and CVaRs of the portfolio although only 3 or 6 R_n^i's are selected from the tail.

2.4 Other Real Scenario Models

In previous subsections, we focus on the two regimes switching lognormal model for the evolution of hedge fund values, R_n. The advantage of this model is that there are analytic formulae for computing the first four moments of R_n. In fact, our method is also applicable to many other finance models with this property, such as the jump diffusion model (Merton, 1976) and those from the generalized autoregressive heteroskedasticity (GARCH) family (Das and Sundaram, Duan et al., Duan et al., Heston and Nandi, Mazzoni, 1999, 1999, 2006, 2000, 2010). As for the models without analytic formulae for the first four moments, we can first simulate a number of time series of hedge fund values evolution via the Monte Carlo method. Then, the estimates of these moments can be obtained from these series and used as inputs for Johnson curve inverse transformation in Algorithm 1. In this case, the Monte Carlo method only generates time series of the hedge fund values, so it takes relatively little computational time.

3 Machine Learning Methods

In the previous section, we proposed an efficient moment matching method to price a single VA contract, which is at least 100 times faster than the nested simulation based on our numerical experiments in Section 4. However, the moment matching method is still too expensive to price and manage the risk of a large VA portfolio. We observe that every VA contract is unique in terms of gender, age, time to maturity, guarantee type and fund type. Therefore, in this section, we combine machine learning methods with the moment matching method. Machine learning methods are usually divided into two phases, the training phase and the predicting phase. In the training phase, a predictive model is built to learn the pattern of the input features and their corresponding outputs. In the predicting phase, the predictive model is employed to predict the outputs of the unseen data only given the input features. In our moment matching machine learning (MMML) approach, we first select a relative small number of VA contracts, and compute all the risk indicators accurately by the moment-matching method. Second, the "machine is trained" with a standard machine learning method, such as neural network or regression tree. Afterwards, the risk indicators of the rest of the contracts are estimated via the trained machine.

3.1 Feature Selection

Before the machine learning methods are introduced, we first define the specific input features for VA contracts. A VA contract typically includes the following attributes: guarantee type, gender, age, account value, withdrawal rate, and the number of years to maturity. All these attributes are crucial in the evaluation of dollar deltas and risk management. The account premium and the market value of the contract are linearly related given the other attributes fixed. Without losing generality, we assume all contacts have a unit premium. Thus, the account value is not necessary to be an input feature. In other words, the number of input features is reduced to five, rather than six. Given a VA contract, x, its corresponding input features can be defined as,

$$feature_1(x) = \begin{cases} 0 & \text{if } x \text{ contains GMDB;} \\ 1 & \text{if } x \text{ contains GMDB and GMWB ,} \end{cases}$$

$$feature_2(x) = \begin{cases} 0 & \text{if } x \text{ is male;} \\ 1 & \text{if } x \text{ is female,} \end{cases}$$

$$feature_3(x) = Z(\text{age of } x),$$

$$feature_4(x) = Z(\text{withdrawal rate of } x),$$

$$feature_5(x) = Z(\text{maturity of } x),$$

where $Z(y) = \dfrac{y - \bar{y}}{std(y)}$, $\bar{y}$ is the mean of the random variable y, and $std(y)$ is the standard deviation of y.

3.2 Training the predictive model

Given a training set of VA contracts expiring within 25 years with the input features defined in subsection 3.1 and corresponding annual dollar deltas, we can employ popular machine learning methods to build up a predictive model. Here, we employ three popular machine learning methods, Regression Tree (Olshen et al., 1984), Neural Network(Braspenning et al., 1995), and Random Forest(Breiman, 2001) to estimate the dollar deltas of a VA portfolio.

A neural network is an effective two-stage model for regression and classification. It can be visualized as a network whose inter-connected nodes mimic the processing element of a distributed information processing structure. Each node of the network contains an activation function that can introduce non-linearity to model complicated patterns. In our case, a feed forward network is considered with sigmoid activation function and multiple hidden layers. The weights of nodes are updated by back-propagation with adjusting learning rates. Weight initialization is crucial in this problem because the back-propagation algorithm can easily converge to local minimums. We follow (Bishop, 1995) to generate the initial weights from a spherically symmetric Gaussian distribution with the variance d^{-1}, where d corresponds to the input dimension. The choice of the variance ensures the summation $a = \sum_{i=1}^{d} w_i$ to be at order unity so that the activation of a is determined by the nonlinear part of the activation function. In each training epoch, the learning rate will be updated accordingly. The modification of learning rate depends on the relative changes of errors and weights. The training process terminates when the maximum number of epochs exceeds a given bound or the error reaches a predefined small value. Since it is hard to find the optimal number of nodes in each hidden layer when there is more than one hidden layer, we use a heuristic technique. The optimal number of hidden nodes, s, is determined given one hidden layer. We then try r number of hidden layers by setting each hidden layer to have $floor(\frac{s}{r})$ number of hidden nodes. As the performance difference from adding additional hidden layer is not very significant(Lopes and Ribeiro, 2015), only three hidden layers are used in this paper.

In addition to neural networks, tree-based regression models can also be used, such as regression tree and random forest method. Tree-based models are independent of data form, and thus usually outperforms other models for mixed-type data set. In the training process, model parameters such as the number of hidden nodes, tree depth, and the number of sample trees in the forest need to be tuned for optimal outcomes. We perform a 10-fold cross-validation to find the optimal values of the aforementioned parameters by minimizing the corresponding mean square errors. The parameters chosen for all three machine learning methods in this paper are listed in Table 2. Although we just adopt these three machine learning methods in this paper, other machines learning methods can also be employed to determine the predictive models.

3.3 Portfolio dollar deltas prediction

We combine the moment matching method proposed in Section 2 and machine learning methods introduced in subsection 3.2, to give a general framework for moment matching machine learning (MMML) approach to evaluate annual dollar delta for a large VA portfolio. The framework to evaluate VaRs and CVaRs can be presented similarly.

Algorithm 3 *(Computing annual dollar deltas for a large VA portfolio by MMML approach)*
Assume that a large VA portfolio is based on the same underlying hedge fund, and all contracts have the five features defined in subsection 3.1.

1. *Randomly select M contracts.*

	# of hidden layers	# of hidden nodes	
Neural Network	1	56	
	2	28	
	3	18	
	min leaf size	# of features selected each split	
Regression Tree	3	3 (predefined)	
	# of sample trees	# of features selected each split	bootstrapping fraction for growing each tree
Random Forest	259	3 (predefined)	1 (predefined)

Table 2: Parameters chosen for neural network, regression tree and random forest methods.

2. *Compute the annual dollar deltas for these M contracts by the moment matching method, Algorithm 2.*

3. *Train the machine by one of machine learning methods in Table 2 with computed results of M contracts obtained from Step 2.*

4. *Predict the dollar delta for the rest contracts based on the trained machine learning method from Step 3.*

5. *Return the expected dollar delta for the whole portfolio.*

4 Numerical Results

In this section, we compare our proposed MMML approach with nested simulations (NS) (Bauer et al., 2008) and unified kriging for function data (UKFD) method (Gan and Lin, 2015). First, the test data set and performance measures are described in subsection 4.1. Then, the computed results are presented in subsection 4.2. The UKFD (Gan and Lin, 2015) and NS method are implemented in C#. To achieve the best performance of the UKFD method, we adopt the C# implementation from the authors of (Gan and Lin, 2015). In our MMML approach, the moment matching method is combined with neural network (MMNN), regression tree (MMTR) and random frost (MMRF) implemented in MATLAB with Statistical and the Neural Network toolboxes. All parameters for these machine learning methods are chosen as in Table 2. All experiments are carried on a machine with Intel Core Duo 3.4 GHz CPU, 16GB RAM and 500GB hard driver, running MATLAB 2014b under Windows 7 Professional.

4.1 Data Set and model setting

The test data were generated similar to (Gan, Gan and Lin, 2013, 2015). Attributes of VA contracts are outlined in Table 3. In this paper, we restrict the data set to contracts with GMDB and GMWB riders only. In fact, contracts with GMAB and GMIB riders can also be handled by our approach with minor modifications in step 2.2 of Algorithm 2 since the Monte Carlo simulation for VA pricing in step 2 is also applicable to price all these complex features (Bauer et al., 2008). In Table 3, we assume values of attributes are uniformly distributed, but other distributions can be treated similarly.

Attribute	**Values**	**Distribution**
Guarantee type	{GMDB,GMDB+GMWB}	50%, 50%
Gender	{Male,Female}	50%, 50%
Age	{20,21,22,...,60}	Uniform
Account value	[10000,500000]	Uniform
GMWB withdrawal rate	{0.04,0.05,0.06,0.07,0.08}	Uniform
Maturity	{10,11,12,...,25}	Uniform

Table 3: Description of variable annuity attributes.

The performance of all methods is measured by Mean Absolute Error (MAE), defined as,

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|f_i - y_i|,$$

where f_i and y_i denote the predicted value and the actual value of the ith contract respectively. The accuracy measurement of estimating annual dollar deltas, VaRs, and CVaRs of the VA portfolio is defined as percentage absolute error (PAE),

$$PAE = \frac{|\sum_{i=1}^{n} f_i - \sum_{i=1}^{n} y_i|}{|\sum_{i=1}^{n} y_i|}.$$

In MAE and PAE, the actual value y_i of the ith contract is evaluated by the nested simulation method with 1000 outer loop scenarios.

4.2 Experimental Results

First, we consider an individual VA contract to demonstrate the effectiveness of our proposed moment matching method. The attributes of the selected contract are described in Table 4.

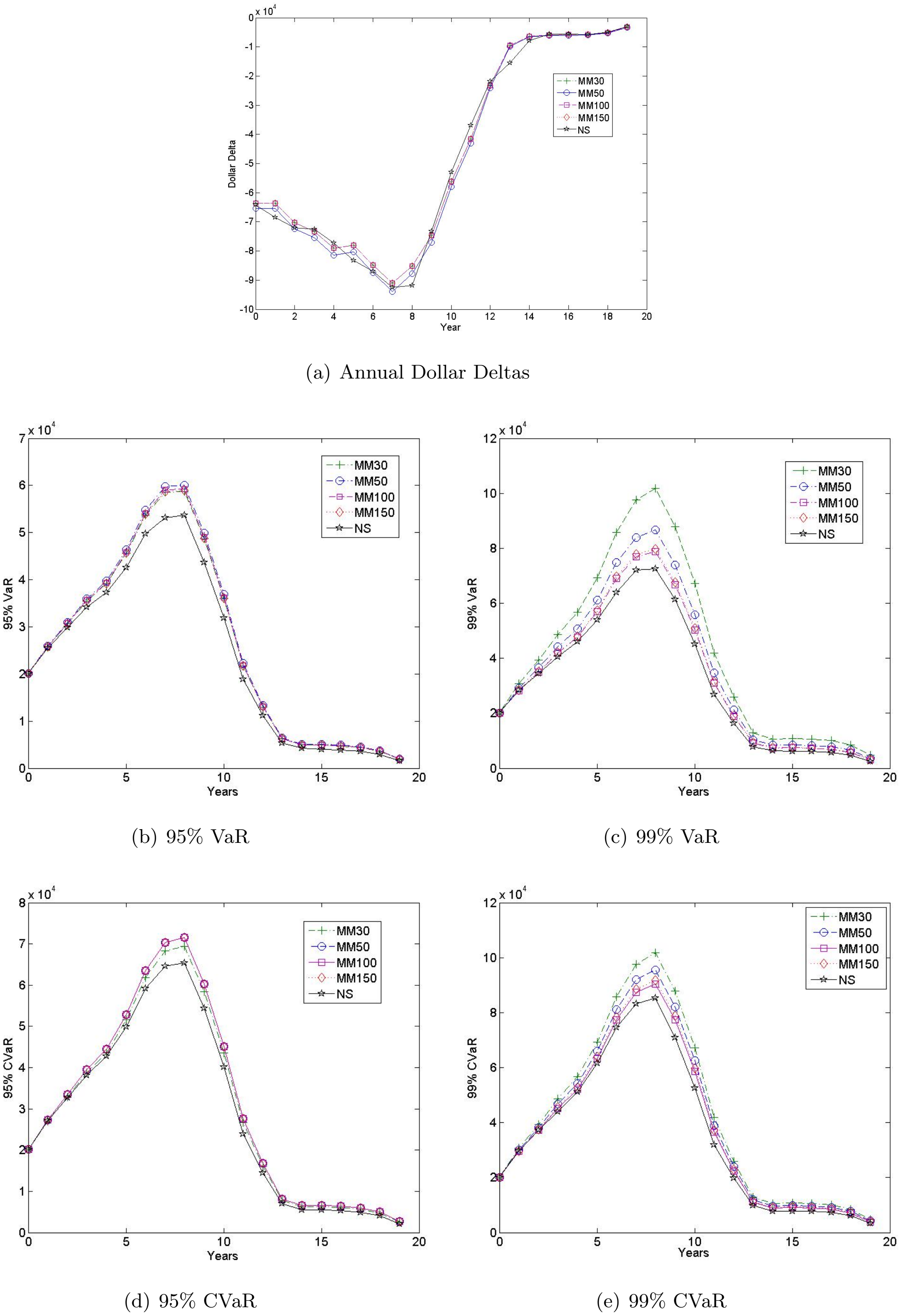
Attribute	Guarantee Type	Gender	Age	Account Value	Withdrawal Rate	Maturity
Value	GMWB+GMDB	Male	41	272,934.2503	8%	19

Table 4: Description of a selected contract.

Assume that the hedge fund follows the two regimes switching lognormal model with $\Theta = \{\mu_1, \mu_2, \sigma_1, \sigma_2, \rho_{12}, \rho_{21}\} = \{0.0126, -0.0185, 0.0350, 0.0748, 0.0398, 0.3798\}$ given in (Hardy, 2001). We first generate m scenarios ($m = 30, 50, 100, 150$) by Algorithm 1 for this model, then the corresponding annual dollar deltas, VaRs and CVaRs are calculated by Algorithm 2. All these computed risk indicators are plotted in Figure 1 along with the results from NS with 10,000 outer loop scenarios and 10,000 inner paths as benchmarks. The results illustrate that when we choose more than 50 selected scenarios in the moment matching method, the computed dollar deltas are very close to benchmarks. The situation for VaR and CVaR is different. When more scenarios are selected in the moment matching method, the computed results get closer to the benchmarks. The reason for this phenomenon is that when the number of scenarios is large, there are more sample scenarios in the tail of the distribution, which results in good approximations of benchmarks. On the other hand, since only the sample scenarios in the tail are considered for VaR and CVaR calculation, the computational time for the moment matching method does not increase as the total number of selected scenarios increases. The computational times record in Table 5 support this claim. Figure 2 shows the plot of the root mean square errors (RMSEs) between our moment matching method and the benchmark (NS with 10,000 outer loop scenarios). It shows that for dollar deltas, a small number of selected scenarios is enough to provide accurate approximations. For VaR and CVaR, the moment matching method requires at least 100 selected scenarios so that there are enough sample scenarios in the tail to provide accurate approximations. Table 5 lists computational times for our moment matching method and NS. It illustrates that our moment matching method is 80 times faster than NS in VaR and CVaR evaluation while it is about 150 times faster in annual dollar delta calculation on a single VA contract.

The second experiment is to compute annual dollar deltas, VaRs and CVaRs for a VA portfolio with 10,000 contracts, whose total account value is 2,564,636,436 dollars. We use the results from NS with 1,000 outer loop scenarios and 10,000 inner paths as benchmarks, which takes about 97.5 hours on a single CPU. In addition, we also include the UKFD method proposed by Gan and Lin (Gan and Lin, 2015) for comparison. As shown in (Gan and Lin, 2015), k representative contracts are determined by k-prototypes clustering. Then, kriging steps are employed to approximate annual dollar deltas for the rest contracts in the portfolio. In our experiments, we take $k = 500,\ 1,000$ and 2,000. In comparison, we randomly select k contracts to set up a training set in our MMML approach.

Figure 3 plots the annual dollar deltas for the 5 different methods, moment matching neutral network (MMNN), moment matching regression tree (MMTR), moment matching random forest (MMRF), UFKD and nested simulations (NS), over 25 years as $k = 500$. The computed results from MMNN, MMTR and MMRF are close to the benchmark except for the MMNN method on the 23rd and 24th year, but it can be improved when the number of training contracts increases. The result of UKFD method is far from the benchmark because it does not have enough representative contacts. Table 6 shows the annual PAE of MMTR, MMNN, MMRF, and UKFD. As years proceed, an increasing



(a) Annual Dollar Deltas

(b) 95% VaR

(c) 99% VaR

(d) 95% CVaR

(e) 99% CVaR

Figure 1: Computed dollar deltas, VaRs and CVaRs from our proposed moment matching method (MM) and nested simulations (NS) with 10,000 outer loop scenarios on an individual VA contract expiring in 19 years, where MM50 stands for the moment matching method with 50 selected scenarios.

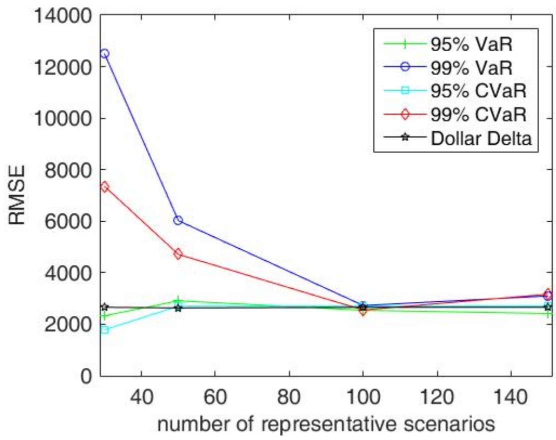


Figure 2: Root mean square errors between computed results from MM and NS for annual dollar deltas, VaRs and CVaRs.

number of contracts expire whose corresponding dollar deltas are zeros. Based on the definition of PAE in (4.1), the value on denominator decreases which leads to increasing in $PAEs$ although the absolute errors between computed results and benchmarks are still small. Even with a small training set, our three MMML approaches are still much more accurate than the UKFD method. Figure 7 plots the computed 95% VaR/CVaR and 99% VaR/CVaR for all five methods. It illustrates that the MMNN method provides the best approximation out of four comparison methods.

Next, we consider the case for $k = 1000$. Figure 4 plots the computed results and benchmarks of annual dollar deltas. The performance of UKFD is improved, but its computed dollar deltas still are not as accurate as those from our MMML approach. The PAEs of UKFD are reduced significantly except first 4 years. It implies that the UKFD method depends on the number of representative contracts. Figure 8 plots 95% VaR/CVaR and 99% VaR/CVaR for all five methods. The MMTR, MMNN and UKFD methods provide good approximations to the benchmarks while MMRF does not improve the results even though k increases to 1,000. Finally, we consider $k = 2,000$. In this case, the computed dollar deltas in Figure 5 from UKFD almost match the benchmarks, but the computed results for the first 4 years are still quite different from the benchmarks although we use 2,000 representative contracts out of a 10,000-contract portfolio. In other words, the UKFD method can generate a large bias on delta hedging for the first 4 or 5 years even with a large number of representative contracts. On the other hand, although the number of representative contracts is doubled , the PAEs of UKFD for the first 4 years are still big compared to other methods as results listed in Table 8. Figure 9 plots corresponding VaRs and CVaRs. It turns out that MMTR and MMRF can predict the dollar deltas very well from now to expiry. Neither of them requires a large training set. MMRF method underestimates the VaRs and CVaRs in a long term, but it still can

Criteria	Method	Computational time (second)
VaR	MM30	1.4756
	MM50	2.0468
	MM100	3.4036
	MM150	3.6192
	NS	258.4800
CVaR	MM30	1.5683
	MM50	2.0665
	MM100	3.4629
	MM150	3.7901
	NS	259.7648
Dollar delta	MM30	2.3656
	MM50	3.7568
	MM100	6.8336
	MM150	9.6692
	NS	594.9000

Table 5: Computational times for VaR, CVaR, dollar delta calculation by MM and NS methods where MM50 stands for the moment matching method with 50 selected scenarios.

return good VaRs and CVaRs approximations in the first 10 years. For a small training set, the MMNN method has errors in predicting dollar deltas, but it can predict the VaRs and CVaRs well. As for the UKFD method, it performs well when the number of representative contacts is larger than 1,000 for a 10,000-contract portfolio. However, its predictions within the first 4 or 5 years are not accurate for dollar delta, VaR or CVaR calculation even with 2,000 representative contracts. Figure 6 plots the MAEs for all three MMML approaches for various sized training set in dollar delta computing. It turns out that the MMTR method generates the minimal absolute errors out of three methods compared to the benchmarks.

Table 9 and 10 record the computational times for all five methods in annual dollar deltas and CVaR computation. Our MMML approach is about 20 times faster than UKFD method while it is about 100 times faster than NS. There are three stages in MMML approach and UKFD method, pricing, training and prediction. In pricing, the NS is employed for UKFD method to evaluate dollar deltas and CVaRs for representative contracts while moment matching method is used for our MMML approaches. It demonstrates that the moment matching method yields a significant saving in pricing, which dominates the total computational time. In the training and prediction stage, the MMTR and MMFR methods are much more efficient than UKFD and MMNN method. In summary, the MMTR method not only provides accurate computed results in dollar deltas, VaRs and CVaRs, but also is most efficient out of five methods.

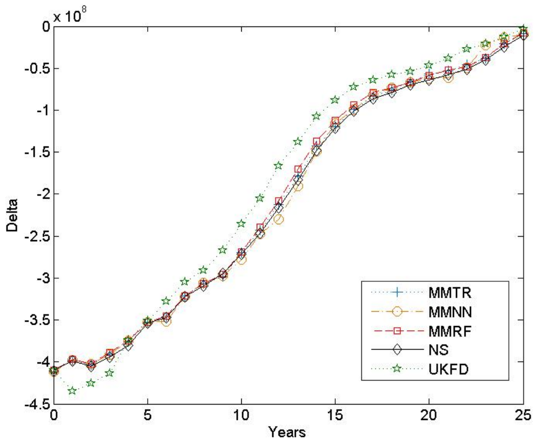


Figure 3: Annual dollar deltas of the MMNN, MMTR, MMRF and UKFD methods for 500 representative contracts with NS as benchmarks.

The third experiment tests a 200,000-contract portfolio. Due to the size of the portfolio, the NS is not applicable since it takes about 1.5 years to obtain all annual dollar deltas, VaRs and CVaRs. We compare the results from our MMML approach with the UKFD method. Figure 9 plots the computed dollar deltas from our MMNN, MMTR, MMFR and UKFD methods with $k = 500$, 1000 and 2000. When $k = 500$, MMNN does not return reasonable deltas in some years, but it can be improved when k increases. Just as we mentioned in the second experiment, the computed results from the UKFD method get close to the ones from our MMML approaches when k increases, but there is still a big difference in the first 4 years. The results from the MMTR and MMRF methods do not vary no matter the value of k. Figure 10 and 11 plot the corresponding annual VaRs and CVaRs for all four methods. Although MMNN does not perform well in dollar deltas estimation when $k = 500$, it still works well in annual VaRs and CVaRs computation. All three MMML approaches have similar VaR and CVaR approximations in the first 10 years, which is different from the ones of the UKFD method. Table 12 and 13 record the corresponding computational times for all four methods. It illustrates that the MMTR method is most efficient out of four methods, which is about 30-40 times faster than UKFD method.

In summary, our proposed MMML approach can provide an accurate and efficient way to estimate portfolio values and manage the risk for large VA portfolios compared to existing NS and UKFD methods.

Year	Regression Tree	Neural Network	Random Forest	UKFD
0	0.1894%	0.3542%	0.0576%	0.267%
1	0.2027%	0.2262%	0.3149%	9.1572%
2	0.5131%	0.5654%	0.6973%	5.0594%
3	0.8848%	0.5594%	1.4063%	5.0505%
4	0.9795%	1.3675%	1.3508%	1.6666%
5	0.3762%	0.2524%	0.3429%	0.8986%
6	0.0853%	1.3825%	0.5994%	5.3949%
7	0.4069%	0.0045%	0.4469%	5.3643%
8	0.8652%	0.9983%	0.7535%	5.9727%
9	1.3634%	0.9457%	0.23%	9.5643%
10	0.7996%	2.256%	1.1722%	13.4087%
11	1.4941%	0.4817%	2.7506%	16.724%
12	1.6406%	6.485%	3.8103%	23.2268%
13	2.3008%	3.9031%	6.9412%	24.4708%
14	3.6153%	0.5843%	7.1795%	27.3001%
15	1.4892%	3.9736%	7.2476%	27.3459%
16	1.8818%	1.998%	7.1721%	28.4808%
17	3.7222%	6.1835%	8.3905%	26.4609%
18	4.0433%	6.6456%	7.1087%	27.0288%
19	5.7283%	3.958%	4.3254%	22.5858%
20	12.4065%	3.4934%	8.6322%	27.874%
21	8.5508%	5.5816%	9.5307%	34.205%
22	11.0662%	3.7869%	6.5343%	46.3626%
23	5.473%	44.7494%	8.1402%	49.4967%
24	16.919%	46.8886%	17.0695%	48.5616%
25	14.22%	10.7284%	23.5181%	65.9987%

Table 6: The percentage absolute errors (PAEs) of annual dollar deltas from MMTR, MMNN, MMRF and UKFD methods on a 500-contract training set.

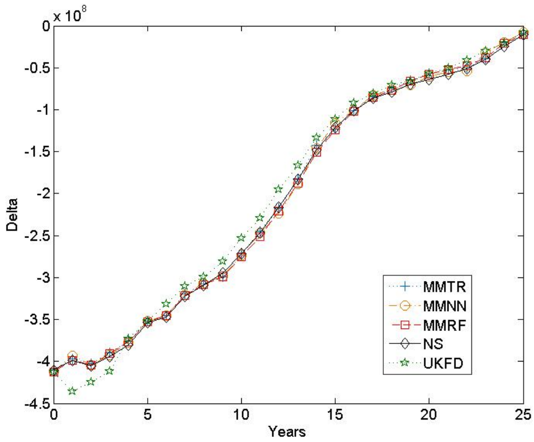


Figure 4: Annual dollar deltas of the MMNN, MMTR, MMRF and UKFD methods for 1000 representative contracts with NS as benchmarks.

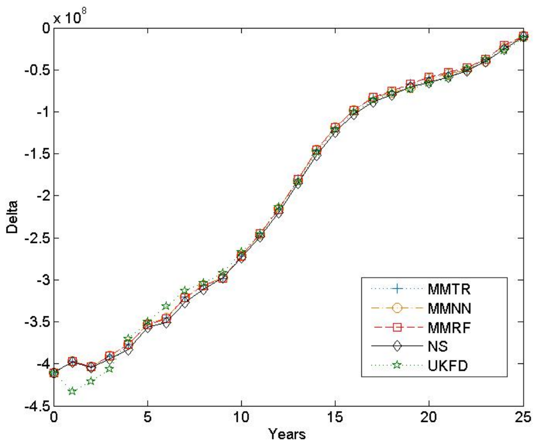


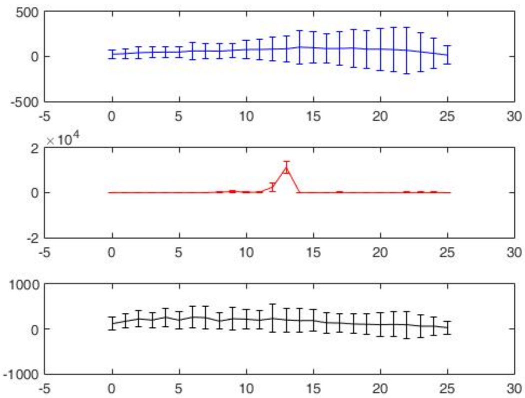
Figure 5: Annual dollar deltas of the MMNN, MMTR, MMRF and UKFD methods for 2000 representative contracts with NS as benchmarks.

Year	Regression Tree	Neural Network	Random Forest	UKFD
0	0.0343%	0.0922%	0.6299%	0.5942%
1	0.3476%	0.2519%	0.0207%	9.4125%
2	0.5217%	0.5256%	0.7372%	4.7347%
3	0.9894%	0.814%	1.603%	4.4602%
4	0.9102%	0.892%	1.0992%	1.9012%
5	0.2642%	0.257%	0.3999%	0.5806%
6	0.4088%	0.1107%	0.3553%	4.3365%
7	0.5477%	0.6506%	0.4274%	3.84%
8	0.7022%	0.6881%	0.8955%	3.0826%
9	0.9418%	1.9621%	1.3633%	4.783%
10	0.5156%	0.0688%	0.452%	6.8489%
11	0.8084%	1.5212%	2.3083%	6.9859%
12	0.7188%	0.7672%	1.1765%	9.91%
13	1.4561%	1.3289%	0.0242%	9.0427%
14	2.3999%	1.8473%	1.2017%	10.2724%
15	2.9056%	2.3533%	1.9557%	8.2605%
16	3.4876%	2.6239%	3.8713%	8.8059%
17	4.6011%	5.5381%	6.4621%	7.2023%
18	5.1858%	3.9882%	6.986%	11.065%
19	2.8422%	1.9423%	5.8571%	6.4923%
20	6.5922%	8.2988%	5.9235%	11.7178%
21	8.0779%	10.9336%	7.6597%	13.9204%
22	7.6528%	6.9267%	4.2044%	20.2639%
23	10.1504%	26.4989%	7.408%	24.8633%
24	15.3669%	15.1472%	21.9137%	17.4535%
25	12.5709%	14.5144%	10.955%	24.7977%

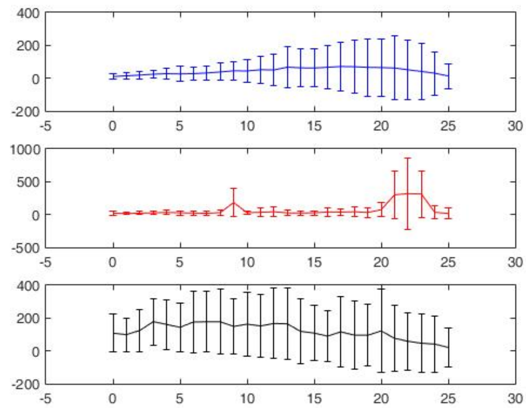
Table 7: The percentage absolute errors (PAEs) of annual dollar deltas from MMTR, MMNN, MMRF and UKFD methods on a 1000-contract training set.

Year	Regression Tree	Neural Network	Random Forest	UKFD
0	0.0186%	0.0087%	0.0131%	0.3357%
1	0.3004%	0.5255%	0.2851%	9.0308%
2	0.4831%	0.4014%	0.5067%	3.9919%
3	0.8505%	0.6493%	0.7896%	3.4913%
4	0.9238%	0.9143%	0.7844%	2.7799%
5	0.296%	0.7245%	0.4236%	1.7385%
6	0.3485%	0.3864%	0.2463%	5.2595%
7	0.5753%	0.463%	0.9047%	4.6294%
8	0.7176%	0.8567%	0.6631%	3.5834%
9	1.0309%	1.1318%	1.5019%	4.5112%
10	0.0083%	0.0784%	0.174%	4.4499%
11	0.7116%	1.1615%	0.9578%	2.3204%
12	0.2913%	1.1545%	0.0117%	2.578%
13	1.4322%	5.6479%	1.231%	0.4003%
14	2.3589%	1.7808%	1.3794%	1.1354%
15	2.2941%	6.4604%	0.9963%	0.6111%
16	2.6876%	1.4015%	3.8365%	1.3251%
17	3.7288%	3.8309%	5.4481%	1.4626%
18	3.3553%	3.3477%	4.8107%	4.7105%
19	3.9778%	2.9328%	4.9479%	0.5775%
20	7.5902%	6.2331%	9.9149%	0.7869%
21	6.3186%	6.4172%	8.288%	0.2569%
22	4.2712%	3.4927%	5.6675%	6.5739%
23	6.7332%	2.3018%	9.1386%	8.6174%
24	13.1917%	8.5905%	15.1178%	2.7841%
25	12.3421%	19.6912%	13.2043%	25.5092%

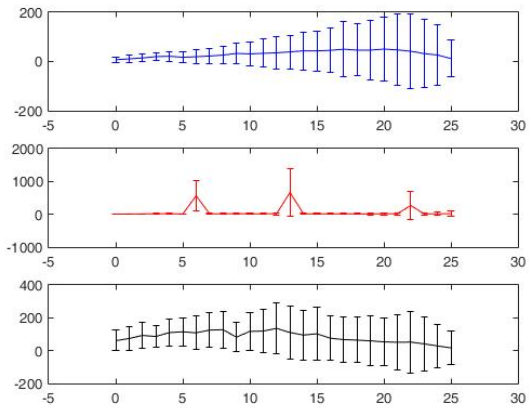
Table 8: The percentage absolute errors (PAEs) of annual dollar deltas from MMTR, MMNN, MMRF and UKFD methods on a 2000-contract training set..



(a) Training with 500 contracts



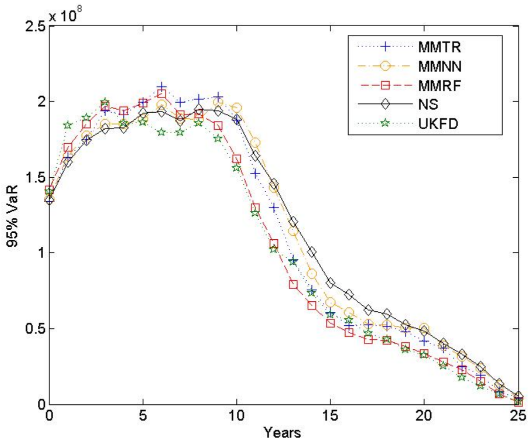
(b) Training with 1000 contracts



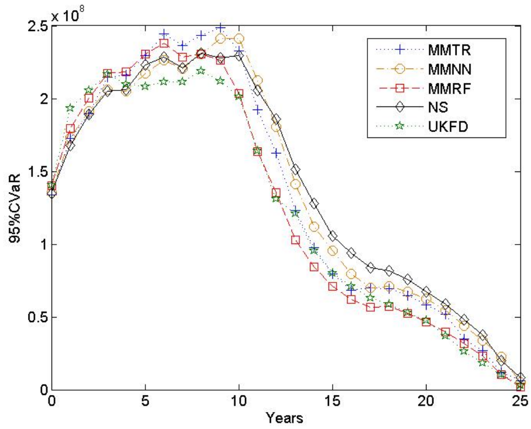
(c) Training with 2000 contracts

Figure 6: The mean absolute error (MAE) $\pm$ the standard deviation of the absolute error for three MMML approaches (MMTR (top), MMNN (middle), MMRF (bottom)) with various sizes of training sets.

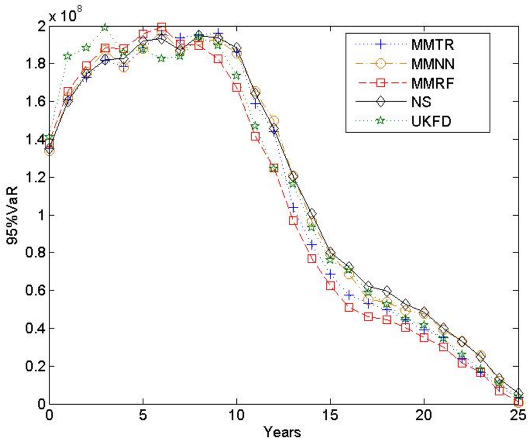
Conference Copy



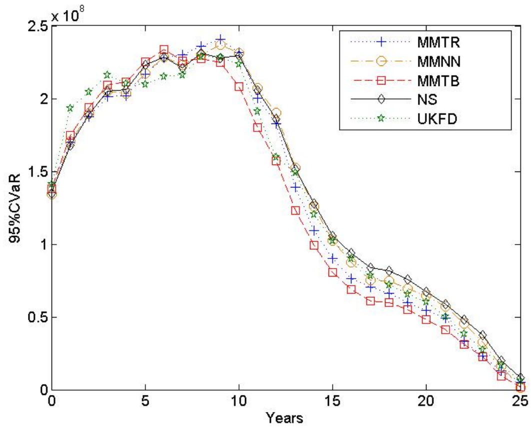
(a) 95% CVaR with $k = 500$



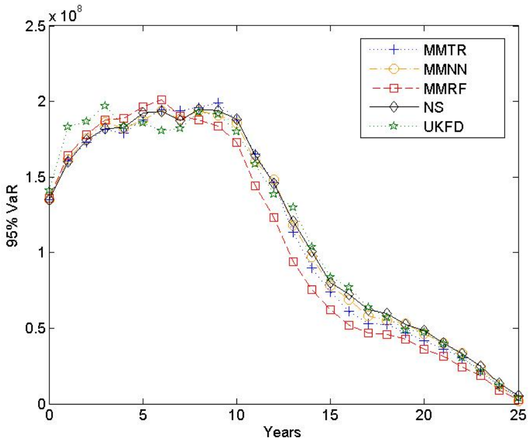
(b) 95% CVaR with $k = 500$



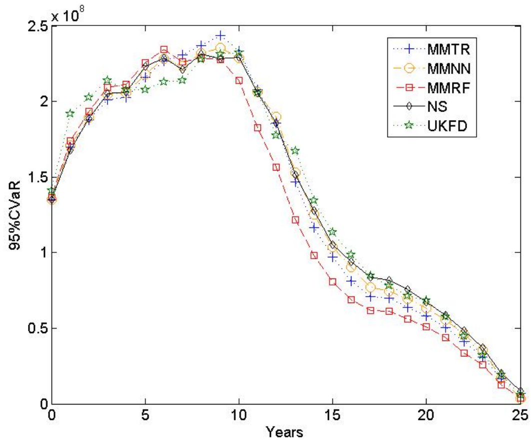
(c) 95% VaR with $k = 1000$



(d) 95% CVaR with $k = 1000$

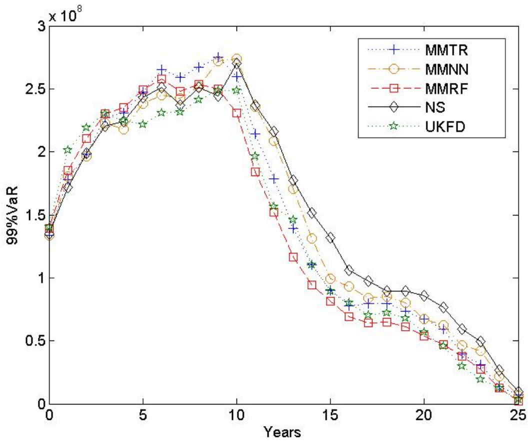


(e) 95% VaR with $k = 2000$

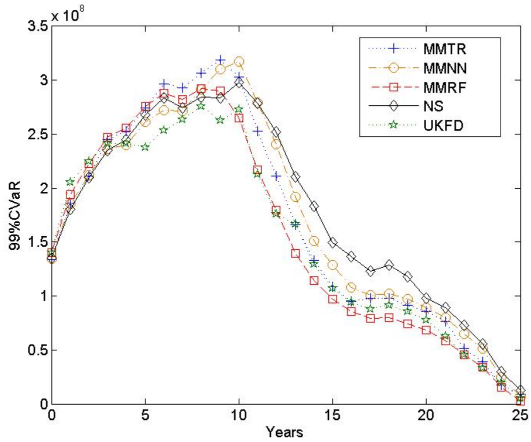


(f) 95% CVaR with $k = 2000$

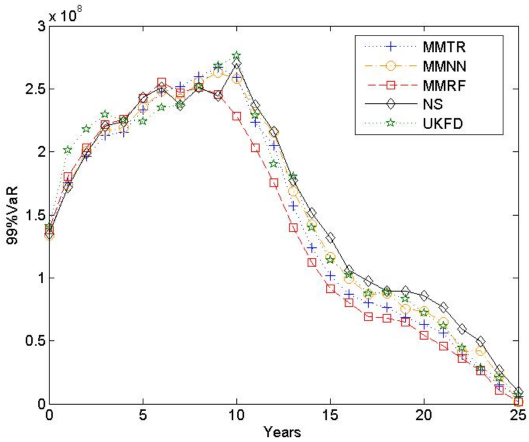
Figure 7: Computed 95% VaRs, and 95% CVaRs from MMTR, MMNN, MMRF and UKFD methods with benchmarks from NS on a 10,000-contract portfolio over 25 years.



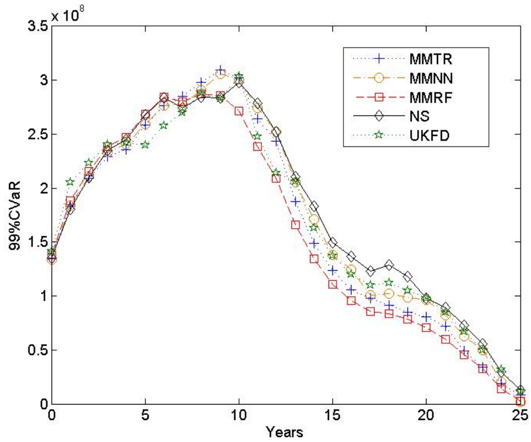
(a) 99% VaR with $k = 500$



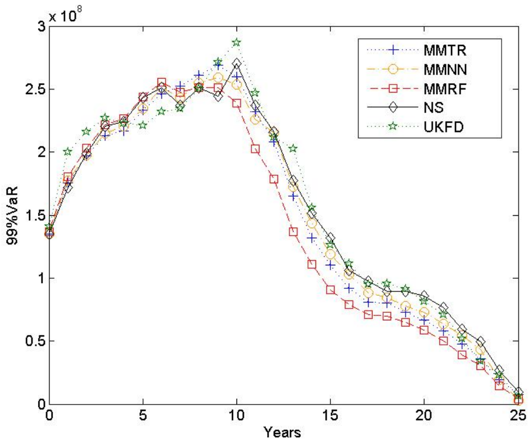
(b) 99% CVaR with $k = 500$



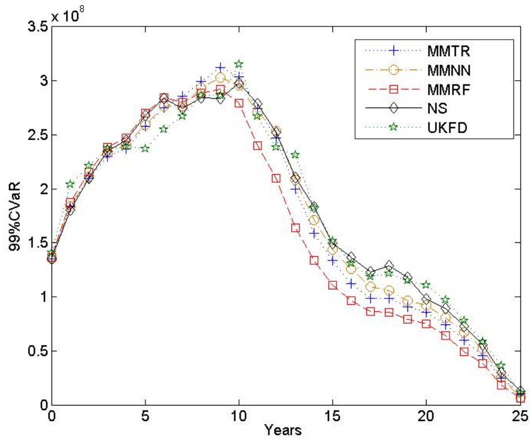
(c) 99% VaR with $k = 1000$



(d) 99% CVaR with $k = 1000$

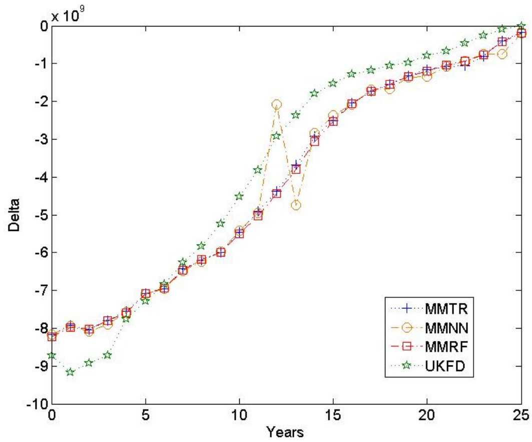


(e) 99% VaR with $k = 2000$

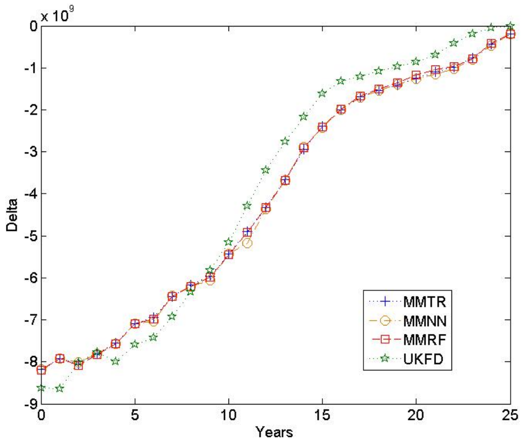


(f) 99% CVaR with $k = 2000$

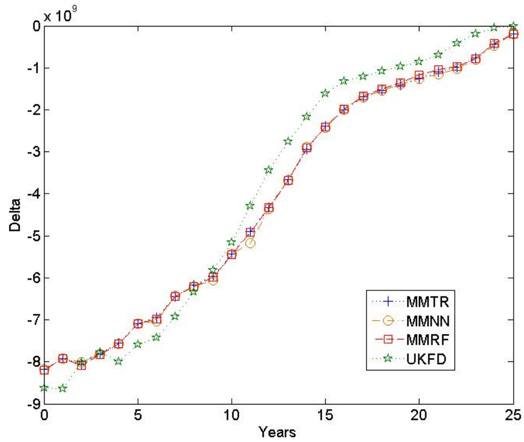
Figure 8: Computed 99% VaRs, and 99% CVaRs from MMTR, MMNN, MMRF and UKFD methods with benchmarks from NS on a 10,000-contract portfolio over 25 years.



(a) Annual dollar delta with $k = 500$

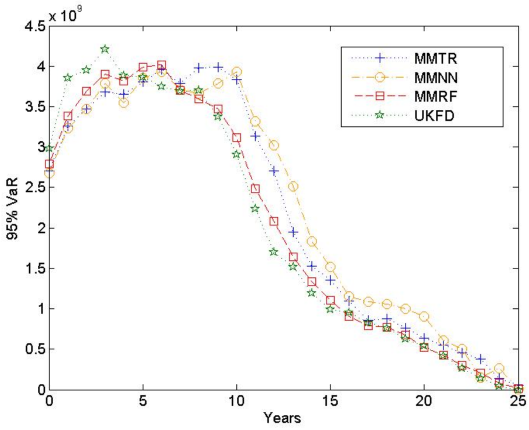


(b) Annual dollar delta with $k = 1000$

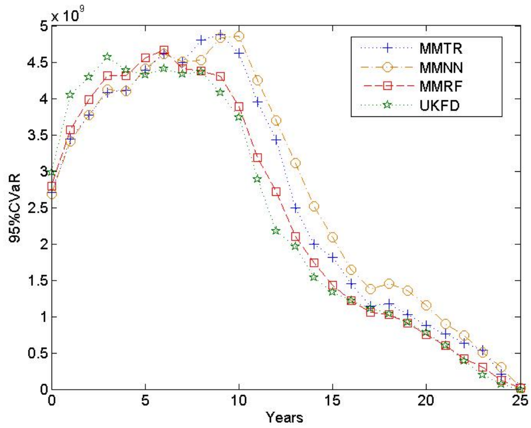


(c) Annual dollar delta with $k = 2000$

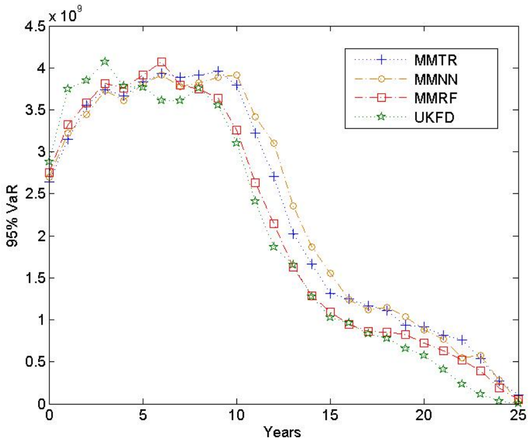
Figure 9: Computed annual dollar delta from MMTR, MMNN, MMRF and UKFD methods on a 200,000-contract portfolio over 25 years..



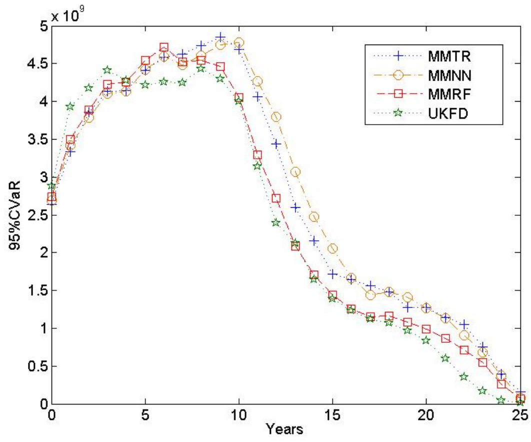
(a) 95% VaR with $k = 500$



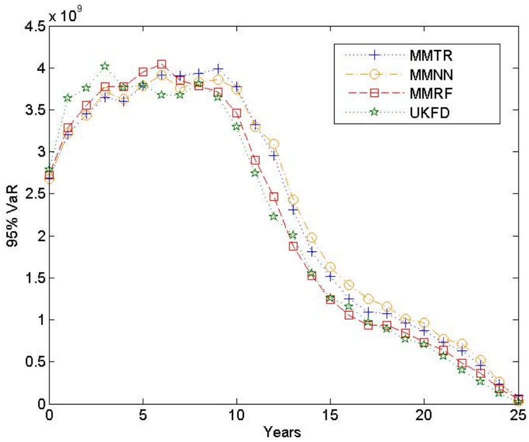
(b) 95% CVaR with $k = 500$



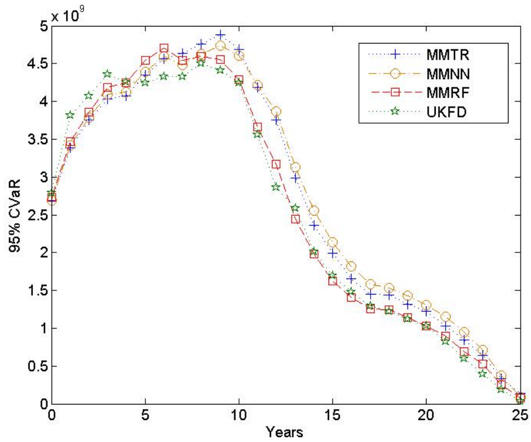
(c) 95% VaR with $k = 1000$



(d) 95% CVaR with $k = 1000$

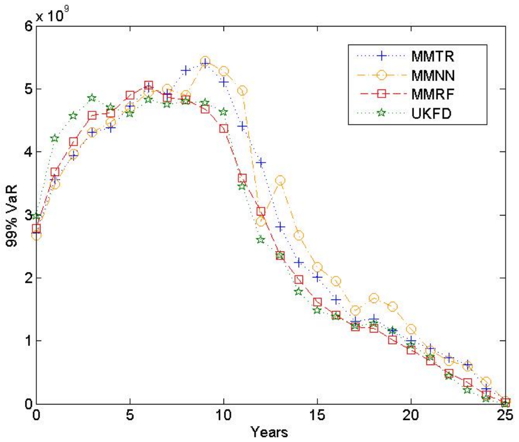


(e) 95% VaR with $k = 2000$

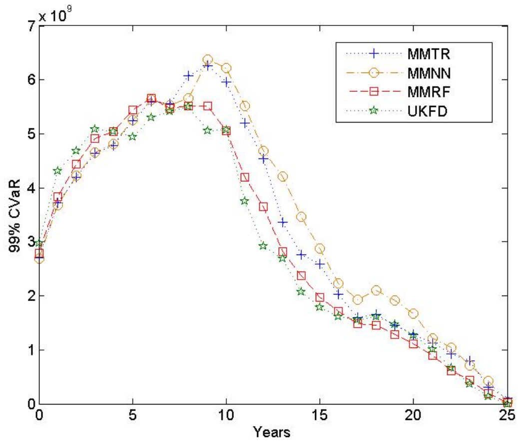


(f) 95% CVaR with $k = 2000$

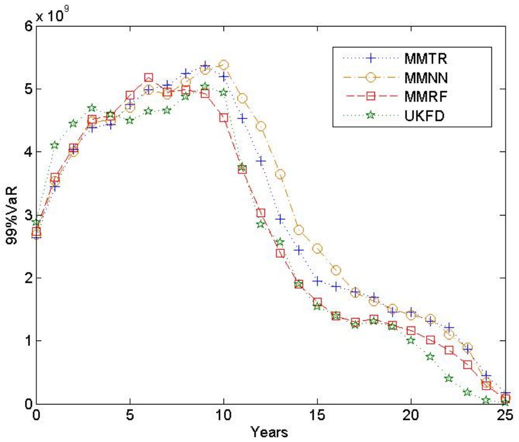
Figure 10: Computed 95% VaRs, and 95% CVaRs from MMTR, MMNN, MMRF and UKFD methods on a 200,000-contract portfolio over 25 years with various sized training sets.



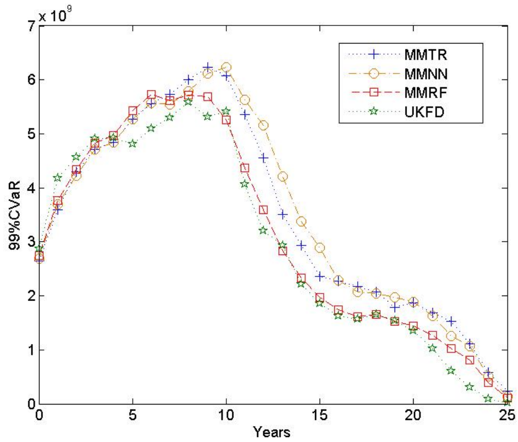
(a) 99% VaR with $k = 500$



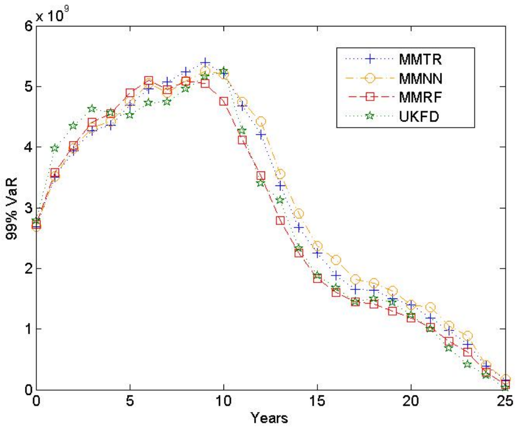
(b) 99% CVaR with $k = 500$



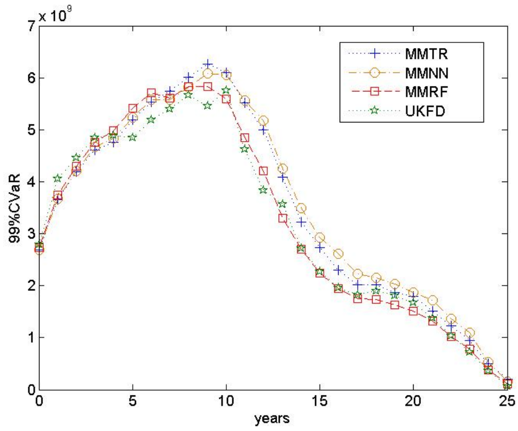
(c) 99% VaR with $k = 1000$



(d) 99% CVaR with $k = 1000$



(e) 99% VaR with $k = 2000$



(f) 99% CVaR with $k = 2000$

Figure 11: Computed 95% VaRs, and 95% CVaRs from MMTR, MMNN, MMRF and UKFD methods on a 10,000-contract portfolio over 25 years with various sized training sets.

Training size	Method	Pricing	Prediction	Total time (second)
500	MMTR	953.4	0.62	954.02
	MMNN		273.52	1226.92
	MMRF		23.41	976.81
	UKFD	15516.25	142.84	15659.09
1000	MMTR	2111.03	0.83	2112.13
	MMTR		427.21	2538.51
	MMRF		32.34	2143.64
	UKFD	33222.85	303.7	33526.55
2000	MMTR	3983.15	5.21	3988.36
	MMNN		1401.38	5384.53
	MMRF		73.07	4056.22
	UKFD	69786.06	658.54	70444.6
	Nested Monte Carlo with 1000 outer loop			350491.0

Table 9: Computational times of the MMNN, MMTR, MMRF, UKFD and NS on annual dollar deltas computational on a 10,000 contract portfolio with various sizes of training sets.

5 Conclusions

Variable annuities are very popular in the market, but when the VA portfolio is large, it is time-consuming to manage the risk of the portfolio through nested simulations. In this paper, we propose a moment matching machine learning (MMML) approach to compute dollar deltas, VaRs and CVaRs for the large portfolio. There are two main contributions in our paper.

First, we propose a moment matching method to compute annual dollar deltas, VaRs, and CVaRs for a single VA contract. Compared to nested simulations, our method is much more efficient. The outer scenarios are well selected to match the first four moments of some specified stochastic models, such as the regime switching model (Hardy, 2001), jump diffusion model (Merton, 1976), GARCH family (Das and Sundaram, Duan et al., Duan et al., Heston and Nandi, Mazzoni, 1999, 1999, 2006, 2000, 2010) and so on. Due to these selected scenarios, the moment matching method can compute the annual dollar deltas, VaRs and CVaRs as accurate as the nested simulations, but only takes less than 1% computational time as nested simulations requires.

The second contribution is that we combined the moment matching method with some classical machine learning methods to manage the risk of a large VA portfolio. We first select a relative small number of VA contracts, and compute all the risk indicators accurately by the moment-matching method. Second, the “machine is trained” with a

Training size	Method	Pricing	Prediction	Total time(second)
500	MMTR	936.69	0.53	937.22
	MMNN		45.25	981.94
	MMRF		5.33	942.02
	UKFD	7366.23	71.41	7437.64
1000	MMTR	1922.04	1.92	1923.96
	MMNN		169.02	2091.06
	MMRF		21.05	1943.09
	UKFD	16778.43	162.3	16940.73
2000	MMTR	3725.4	86.71	3812.11
	MMNN		781.06	4506.46
	MMRF		81.63	3807.03
	UKFD	34210.5	392.6	34603.1
	Nested Monte Carlo with 1000 outer loop			182371.0

Table 10: Computational times of the MMNN, MMTR, MMRF, UKFD and NS on CVaR computational on a 10,000 contract portfolio with various sizes of training sets.

Training Size	Method	Pricing	Prediction	Total time (second)
500	MMTR	923.99	4.21	928.2
	MMNN		301.12	1225.11
	MMRF		37.5	961.49
	UKFD	15007.3	248.307	15255.607
1000	MMTR	2044.28	5.63	2049.91
	MMNN		443.99	2488.27
	MMRF		57.12	2101.4
	UKFD	31785.83	498.7	32284.53
2000	MMTR	4120.7	19.7	4140.4
	MMNN		1239.5	5360.2
	MMRF		267.3	4388
	UKFD	71323.59	1928.36	73251.95

Table 11: Computational times of the MMNN, MMTR, MMRF, and UKFD on annual dollar delta computation on a 200,000 contract portfolio with various sizes of training sets..

Training Size	Method	Pricing	Prediction	Total time (second)
500	MMTR	916.85	3.27	920.12
	MMNN		68.25	985.1
	MMRF		8.23	925.08
	UKFD	7543.21	236.76	7779.97
1000	MMTR	1917.5	9.02	1926.52
	MMNN		265.3	2182.8
	MMRF		31.2	1948.7
	UKFD	16997.56	479.35	17476.91
2000	MMTR	3703.2	31.3	3734.5
	MMNN		1210.22	4913.42
	MMRF		232.51	3935.71
	UKFD	35801	1983.75	37784.75

Table 12: Computational times of the MMNN, MMTR, MMRF, and UKFD on CVaR computation on a 200,000 contract portfolio with various sizes of training sets..

standard machine learning method, such as neutral network or tree regression. Finally, the risk indicators of the rest of the contracts are estimated via the trained machine. Our MMML approach can easily handle huge portfolios (which cannot be handled via the nested simulation method due to cost). For example, for a portfolio with 200,000 VA contracts, We estimate it will take nested simulation more than 1.5 years to obtain all these indicators, but only about 1.5 hours for the proposed machine learning approach with a 2000-contract training set. Although the portfolio has 200,000 contracts, a 2000-contract training set is large enough for the machine learning approach. In other words, a small training set is enough to produce accurate risk indicator estimates.

In conclusion, our proposed MMML approach appears to be a remarkably efficient alternative to the standard nested simulation methodology to hedge and manage the risk of large portfolios arising in the insurance industry. Further development and testing is needed on a real data set with a larger number of attributes (and perhaps more widely varying). Finally, we expect there will be application of similar ideas to other large complex portfolios of financial instruments.

A Johnson Curve

Johnson curve (Johnson, 1949) can transform an arbitrage continuous random variable X into a standard normally distributed random variable Z in the form of

$$Z = a + b \cdot g\left(\frac{X - c}{d}\right),$$

where a and b are shape parameters, c is a location parameter, d is a scale parameter and $g(\cdot)$ is a function which has forms for the four families of distributions, i.e.,

$$g(u) = \begin{cases} \ln(u) & \text{for the lognormal family,} \\ \ln(u + \sqrt{u^2+1}) & \text{for the unbounded family,} \\ \ln(u/(1-u)) & \text{for the bounded family,} \\ u & \text{for the normal family.} \end{cases}$$

The main advantage of Johnson curve is the flexibility to match any set of values with the mean, variance, skewness and kurtosis for a given distribution. On the other hand, its inverse transformation can transform a standard normally distributed random variable Z into a given distribution X, as

$$X = c + d \cdot g^{-1}\left(\frac{Z-a}{b}\right),$$

where

$$g^{-1}(u) = \begin{cases} e^u & \text{for the lognormal family,} \\ (e^u - e^{-u})/2 & \text{for the unbounded family,} \\ 1/(1+e^{-u}) & \text{for the bounded family,} \\ u & \text{for the normal family.} \end{cases}$$

The algorithm proposed in (Hill et al., 1976) can determine parameters a, b, c and d, and the form of $g(\cdot)$ function given the first four moments of a target distribution. First, a parameter γ is computed to determine the family of the Johnson system

$$\gamma = \omega^4 + 2\omega^3 + 3\omega^2 - 3,$$

where κ_3 and κ_4 are the skewness and kurtosis of X, respectively, and

$$\omega = 0.5\left(8 + 4\kappa_3^2 + 4\sqrt{4\kappa_3^2 + \kappa_3^4}\right)^{1/3} + 0.5\left(8 + 4\kappa_3^2 + 4\sqrt{4\kappa_3^2 + \kappa_3^4}\right)^{-1/3} - 1.$$

If γ is close to κ_4, the lognormal family is chosen. If γ is smaller than κ_4, the unbounded family is chosen while the bounded case is for γ greater than κ_4. Then, all parameters can be determined based on the identified family. For the lognormal case, the parameters can be evaluated as

$$b = (\ln\omega)^{-1/2}, \qquad a = 0.5b\ln(\omega(\omega-1)), \qquad d = \operatorname{sign}(\kappa_3), \quad \text{and} \quad c = -\exp(\frac{0.5b-a}{b}).$$

For the unbounded family, if κ_3 is close to zero, then $a = 0$ and $b = \ln(\zeta)^{-1/2}$, where $\zeta = \sqrt{(2\kappa_4 - 2)^{1/2} - 1}$. Otherwise, the algorithm in (Hill et al., 1976) gives a procedure to compute ζ, a and b. Given ζ, a and b, the values of c and d can be determined as

$$d = (0.5(\zeta-1)(\zeta\cosh(2a/b)+1))^{-1/2} \qquad \text{and} \qquad c = d\sqrt{\zeta}\sinh(a/b).$$

As for the bounded family, there is an iterative algorithm in (Hill et al., 1976) to determine all parameters.

References

[1] M. Amrstrong. The reset decision for segregated fund maturity guarantees. *Insurance: Mathematics and Economics*, 29:257–269, 2001.

[2] D. Bauer, A. Kling, and J. Russ. A universal pricing framework for guaranteed minimum benefits in vairable annuities. *ASTIN Bulletin*, 38:621–651, 2008.

[3] D. Bauer, A. Russ, and D. Singer. On the calculation of the solvency capital requirement based on nested simulations. *ASTIN Bulletin*, 42:453–499, 2012.

[4] A. Belanger, P. Forsyth, and Labahn G. Valuing the guaranteed minimum death benefit clause with partial withdraws. *Applied Mathematical Finance*, 16:451–496, 2009.

[5] Christopher M Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995.

[6] P. Boyle and W. Tian. The design of equity-indexed annuities. *Insurance: Mathematics and Economics*, 43:303–315, 2008.

[7] P. Boyle, A. Kolkiewicz, and K. Tan. Valuation of the reset options embedded in some equity-link insurance products. *North American Actuarial Journal*, 5:1–18, 2001.

[8] Petrus J Braspenning, Frank Thuijsman, and Antonius Jozef Martha Maria Weijters. *Artificial neural networks: an introduction to ANN theory and practice*, volume 931. Springer Science & Business Media, 1995.

[9] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[10] M. Cathcart and S. Morrison. Variable annuity economic capital: the least squares monte carlo approach. *Life Pension*, pages 36–40, 2009.

[11] T.F. Coleman, Y. Li, and M.C. Patron. Pricing american options on jump-diffusion processes using fourier hermite series expansions. *Insurance: Mathematics and Economics*, 38:215–228, 2006.

[12] S. Das and R. Sundaram. Of smiles and smirks: A term structure perspective. *Journal of Financial and Quantitative Analysis*, 34(02):211–239, 1999.

[13] S. Daul and E.G. Vidal. Replication of insurance liabilities. *RiskMetrics J.*, 9, 2009.

[14] R. Dembo and D. Rosen. The practice of portfolio replication: a practical overview of forward and inverse problems. *Ann. Oper. Res.*, 85:267–284, 1999.

[15] J.C. Duan, G. Gauthier, and J.G. Simonato. An analytical approximation for the garch option pricing model. *Journal of Computational Finance*, 2:75–116, 1999.

[16] J.C. Duan, G. Gauthier, C. Sasseville, and J.G. Simonato. Approximating the gjr-garch and egarch option pricing models analytically. *Journal of Computational Finance*, 9:41–69, 2006.

[17] J. Fox. A nested approach to simulation var using mosec. *Insights: Financial Modelling*, pages 1–7, 2013.

[18] G. Gan. Application of data clustering and machine learning in variable annuity valuation. *Insurance: Mathematics and Economics*, 53:795–801, 2013.

[19] G. Gan and X. Lin. Valuation of large variable annuity portfolios under nested simulation: A functional data approach. *Insurance: Mathematics and Economics*, 62:138–150, 2015.

[20] J. Gan and E. Ulm. Optimal consumption and allocation in variable annuities with guaranteed minimum death benefits. *Insurance: Mathematics and Economics*, 51 (3):586–598, 2012.

[21] M. Hardy. A regime-switching model of long-tem stock returns. *North American Actuarial Journal*, 5:41–53, 2001.

[22] S. Heston and S. Nandi. A closed-form garch option pricing model. *Review of Financial Studies*, 13:586–625, 2000.

[23] I. D. Hill, R. Hill, and R. L. Holder. Algorithm as 99: Fitting johnson curves by moments. *Applied Statistics*, pages 180–189, 1976.

[24] H. Huang, M. Milevsky, and T. Salisbury. Optimal initiation of a glwb in a variable annuity: No arbitrage approach. *Insurance: Mathematics and Economics*, 56:102–111, 2014.

[25] J.C. Hull. *Options, futures, and other derivatives*. Pearson Education, 2006.

[26] N.L. Johnson. System of frequency curves generated by methods of translation. *Biometrika*, 36:149–176, 1949.

[27] X.S. Lin, K.S. Tan, and H. Yang. Pricing annuity guarantees under a regime switching model. *North American Actuarial Journal*, 13:316–338, 2009.

[28] F.A. Longstaff and E.S. Schwartz. Valuing american options by simulation: a simple least squares approach. *Review of Financial Study*, 14:113–147, 2001.

[29] Noel Lopes and Bernardete Ribeiro. *Machine Learning for Adaptive Many-Core Machines-A Practical Approach*. Springer, 2015.

[30] T. Mazzoni. Fast analytic option valuation with garch. *Journal of Derivatives*, 18: 18–40, 2010.

[31] R.C. Merton. Hedging guarantees in variable annuities under both equity and interest rate risk. *Journal of Financial Economics*, 3:124–144, 1976.

[32] M. Milevsky and S. Promislow. Mortality derivatives and the option to annuities. *Insurance: Mathematics and Economics*, 9:299–318, 2001.

[33] A. C-Y. Ng and J. S-H Li. Pricing and hedging variable annuity guarantees with multiasset stochastic investment models. *North American Actuarial Journal*, 17: 41–62, 2013.

[34] LBJFR Olshen, Charles J Stone, et al. Classification and regression trees. *Wadsworth International Group*, 93(99):101, 1984.

[35] C. Reynolds and S. Man. Nested stochastic pricing: The time has come. *Product Matters! - Society of Actuaries*, 71:16–20, 2008.

[36] W.R. Sloane. Life insurers, variable annuities and mutual funds: a critical study. *Journal of Risk and Insurance*, 37:99, 1970.

[37] W. Xu and Y. Yin. Pricing american options by willow tree method under jump-diffusion process. *Journal of Derivatives*, 22(1):46–56, 2014.

[38] W. Xu, Z. Hong, and C. Qin. A new sampling strategy willow tree method with application to path-dependent option pricing. *Quantitative Finance*, 13(6):861–872, 2013.

[39] Z. Xu and Q. Wang. Pricing guaranteed minimum withdrawal benefits: a pde approach. *In: International Conference on Management and Service Science*, pages 1–4, 2009.

[40] S. Yang and T.S. Dai. A flexible tree for evaluating guaranteed minimum withdrawal benefits under deferred life annuity contracts with various provisions. *Insurance: Mathematics and Economics*, 52:231–242, 2013.