A Data Driven Neural Network Approach to Optimal Asset Allocation for Target Based Defined Contribution Pension Plans

Yuying Li* Peter Forsyth†

June 6, 2018

Abstract

A data driven Neural Network (NN) optimization framework is proposed to determine optimal asset allocation during the accumulation phase of a defined contribution pension scheme. In contrast to parametric model based solutions computed by a partial differential equation approach, the proposed computational framework can scale to high dimensional multi-asset problems. More importantly, the proposed approach can determine the optimal NN control directly from market returns, without assuming a particular parametric model for the return process. We validate the proposed NN learning solution by comparing the NN control to the optimal control determined by solution of the Hamilton-Jacobi-Bellman (HJB) equation. The HJB equation solution is based on a double exponential jump model calibrated to the historical market data. The NN control achieves nearly optimal performance. An alternative data driven approach (without the need of a parametric model) is based on using the historic bootstrap resampling data sets. Robustness is checked by training with a blocksize different from the test data. In both two and three asset cases, we compare performance of the NN controls directly learned from the market return sample paths and demonstrate that they always significantly outperform constant proportion strategies.

Keywords: DC plan asset allocation, data-driven, neural network, target based objective

JEL classification: C61, D81, G11, G22

1 Introduction

Throughout the Western world, it is clear that there is a major paradigm shift away from defined benefit (DB) pension plans to defined contribution (DC) plans. Both the public and private sectors are no longer willing to take on the risk of DB plans.

In a typical employee sponsored DC plan, employee and employer contribute to a (usually) tax advantaged account. Often, the employee is presented with a list of possible investment funds, and then required to select an asset allocation from this list. If one considers that the DC fund will be managed by the employee for 30+ years of employment, it is clear that asset allocation will be of crucial importance in order to have a reasonable level of salary replacement during the retirement (decumulation) phase.

In this article, we formulate the multi-period asset allocation strategy during the accumulation phase of a DC pension fund as an optimal stochastic control problem. We use the target based objective function advocated in (Menoncin and Vigna, 2017).

*David R. Cheriton School of Computer Science, University of Waterloo, Waterloo ON, Canada N2L 3G1, yuying@uwaterloo.ca, +1 519 888 4567 ext. 37825.

†David R. Cheriton School of Computer Science, University of Waterloo, Waterloo ON, Canada N2L 3G1, paforsyt@uwaterloo.ca, +1 519 888 4567 ext. 34415.

It is well known (Zhou and Li, 2000; Li and Ng, 2000) that target based objective functions are equivalent to pre-commitment mean variance criteria. Previous work on pre-commitment mean variance asset allocation in the DC pension plan context has been based on (i) postulating a parametric stochastic process for the portfolio components, and (ii) solution of the optimal control problem via a Hamilton Jacobi Bellman (HJB) Partial Differential Equation (PDE) (Vigna, 2014; He and Liang, 2013; Yao et al., 2013; Guan and Liang, 2015).

We should also mention that there is a strand of literature which focuses on time-consistent mean-variance formulations (see Wu et al. (2015) for example). However, target based objective functions (since they can be solved using dynamic programming) are clearly time consistent for a fixed target value. Hence, since we are focusing on target based objective functions, the fact that the equivalent mean-variance problem is time inconsistent is not particularly relevant.

We should also point out that previous work on determining optimal controls for DC plan asset allocation has been mostly restricted to developing closed form solutions of the HJB equation (Vigna, 2014; He and Liang, 2013; Yao et al., 2013; Guan and Liang, 2015; Wu et al., 2015). However, this often requires making unrealistic assumptions, e.g. continuous rebalancing, infinite leverage, trading continues even if insolvent. In fact, one might question previous economic conclusions based on closed-form solutions. For example in (Lioui, 2013), the authors favour time-consistent mean variance strategies compared to pre-commitment policies, since the latter uses large leverage ratios. This problem can, of course, be eliminated by imposing realistic leverage ratio constraints.

In order to allow for practical considerations for DC pension plan asset allocation, such as no-shorting and no-leverage, and discrete rebalancing, numerical methods for solution of the associated HJB PDE have been developed in (Dang and Forsyth, 2014; Forsyth and Labahn, 2018).

In contrast to these previous approaches, we propose to use a data-driven method in this work. In other words, we operate directly on the observed historical data, and by-pass the error prone step of calibrating a parametric model to historical data.

In this paper, we propose a data driven machine learning approach for multi-period optimal asset allocation strategy during the accumulation phase of a DC pension fund. We remark that machine learning approaches have recently been suggested for a variety of insurance related problems (Gan and Lin, 2015; Gan, 2013; Hejazi and Jackson, 2016). In (Gan and Lin, 2015; Gan, 2013), clustering and a functional data approach have been considered for efficient valuation of portfolios of variable annuities. In (Hejazi and Jackson, 2016), Neural Network approaches are also used for fast computation of portfolios of variable annuities. In our proposed method, we represent the control function of the multi-period allocations as a Neural Network (NN) based on a feedback function of feature variables. Our formulation can handle constraints (i.e. discrete rebalancing, no-shorting, no-leverage) in a straightforward manner. Note that the numerical approaches for solving the HJB PDE in (Dang and Forsyth, 2014; Forsyth and Labahn, 2018) can also handle these types of constraints, but numerical PDE methods are restricted to a small (≤ 3) number of factors. Our NN formulation, in principle, can be applied to problems having a larger number of factors.

We validate our approach by considering a *synthetic* market where the assets follow a known parametric stochastic process. In this case, we can compute the provably optimal asset allocation strategy by solving a Hamilton Jacobi Bellman (HJB) equation. Our data driven solution takes as input only samples from the parametric process. Remarkably, our data driven parsimonious NN approach produces results very close to the HJB solution.

We then test the data driven NN solution of the control problem on bootstrap resampling of the historical data, and compare with constant weight strategies. The data driven NN controls are superior in all cases to the constant weight strategies.

2 Technical Objectives

The objective of this paper is to propose a computational framework, which produces the optimal control for asset allocation in a long term DC pension plan portfolio. The advantages of our approach are (i) based solely on sampling price paths (no parametric model required), (ii) allows realistic constraints on the asset allocation policy (e.g. no leverage, no shorting, discrete rebalancing) and (iii) can potentially be extended to high dimensional cases (many assets, taxation effects, transaction costs).

More precisely, we make the following contributions:

- We take a machine learning approach and determine the control $\vec{p}(\cdot,t)$, where $\vec{p}(\cdot,t)_k$ represents the fraction of the total investor wealth invested in the k^{th} asset at time t. The control is in feedback form, i.e., a function of current state and time. The proposed control model is a Neural Network (NN) with an input layer, a hidden layer, and an output layer. The control model is learned by solving a global optimization model defined by a set of sample paths.

- Using simulated samples from a parametric model estimated from monthly market data, we validate the proposed approach by comparing the performance of the optimal NN control with that of the optimal control determined from the solution of the HJB equation, in two examples with different asset pairs. The HJB solution represents ground truth in this situation.

  We demonstrate that, using an NN of a single hidden layer with only 3 hidden nodes, the training performance of the NN control is on par with that from the HJB equation. In addition, the test results from bootstrap resampling data sets demonstrate that the NN optimal controls significantly outperform a typical constant proportion strategy, yielding a higher median wealth, and a significantly smaller risk, as measured by probability of shortfall and standard deviation, but achieving approximately the same expected wealth.

- Using bootstrap resampling market data sets with varying expected blocksizes, in the two-asset case, we learn the NN optimal controls directly from the market return samples. Performance is tested in out-of-sample mode using returns simulated with the parametric model. The NN control similarly has a higher median wealth, with a lower risk, compared to a constant proportion strategy.

- We also consider 3-asset examples and learn the optimal NN strategy directly from a bootstrap resampling market return data set. Performance is tested in out-of-sample mode using resampling return data sets with blocksizes different from the training set blocksize. We observe similar superior training and testing performance from the NN control, compared to a constant proportion strategy. The NN control produces a higher median terminal wealth with lower risk.

3 Embedded Formulation for Dynamically Optimal Long Term Investment

Consider an investment problem in M risky and riskless assets whose prices $\vec{S}_t$ follow a Markov process. Let the initial time $t_0 = 0$ and consider a set $\mathcal{T}$ of rebalancing times

$$\mathcal{T} \equiv \{t_0 = 0 < t_1 < \ldots < t_N = T\}. \tag{1}$$

The fraction of total wealth allocated to each asset is adjusted at times t_n, $n = 0,\ldots,N-1$, with the investment horizon $t_N = T$.

Assume that, at time t, a fund holds wealth of amount $W_m(t)$ in asset m, $m = 1,\ldots,M$. The total value of the portfolio at t is then

$$W(t) = \sum_{m=1}^{M} W_m(t)\ . \tag{2}$$

Let $t_n^+ = t_n + \varepsilon$, $t_n^- = t_n - \varepsilon$, $\varepsilon \to 0^+$. Assume that $W(t_0^-) = 0$, i.e., the initial value of the portfolio is zero, and let $\{q(t_n)\}$ represent an *a priori* specified cash injection schedule.

Given an allocation control sequence $\vec{\rho}_0, \ldots, \vec{\rho}_{N-1}$, the dependence of the terminal wealth $W(T)$ on the control is as follows,

$$
\begin{array}{ll}
\text{for} & n = 0,1,\ldots,N-1 \\
& W(t_n^+) = W(t_n^-) + q(t_n) \\
& W(t_{n+1}^-) = \vec{\rho}_n^T \vec{R}(t_n) W(t_n^+) \\
& \qquad = \left(\vec{\rho}_n^T \vec{R}(t_n) \right) (W(t_n^-) + q(t_n)), \\
\text{end} &
\end{array}
\tag{3}
$$

where $\vec{R}(t_n)$ is the vector of returns on assets in (t_n^+, t_{n+1}^-).

Using the target based objective function advocated in (Menoncin and Vigna, 2017), a standard formulation for the multi-period allocation problem is the following constrained quadratic optimization, with controls $\vec{\rho}_0(\cdot), \ldots, \vec{\rho}_{N-1}(\cdot)$, where $\vec{\rho}_n(\cdot)$ depends only on the wealth $W(t_n)$,

$$
\begin{array}{ll}
\min\limits_{\{\vec{\rho}_0(\cdot),\ldots,\vec{\rho}_{N-1}(\cdot)\}} & \mathbf{E}\left[(W(T) - W^*)^2 \right] \\
\text{subject to} & 0 \le \vec{\rho}_n(\cdot) \le 1, \quad n = 0,1,\ldots,N-1 \ , \\
& \mathbf{1}^T \vec{\rho}_n(\cdot) = 1 \ , n = 0,1,\ldots,N-1 \ ,
\end{array}
\tag{4}
$$

where W^* is a given target parameter.

The following Proposition, proven in Zhou and Li (2000), establishes an additional appealing property of the quadratic target formulation (4).

Proposition 1 (Dynamic mean variance efficiency)**.** *Problem (4) is multi-period mean variance optimal, in the pre-commitment sense.*

Instead of the quadratic objective in Problem (4), in this paper, we consider

$$
\begin{array}{ll}
\min\limits_{\{\vec{\rho}_0(\cdot),\ldots,\vec{\rho}_{N-1}(\cdot)\}} & g(W(T)) \equiv \mathbf{E}\left[(\min(W(T) - W^*, 0))^2 \right] \\
\text{subject to} & 0 \le \vec{\rho}_n(\cdot) \le 1, \quad n = 0,1,\ldots,N-1 \ , \\
& \mathbf{1}^T \vec{\rho}_n(\cdot) = 1 \ , n = 0,1,\ldots,N-1 \ .
\end{array}
\tag{5}
$$

Remark 1 (Relation of Problem (4) and Problem (5))**.** *In Dang and Forsyth (2016), it is shown that a solution of Problem (5) is a solution of Problem (4) when the set of admissible controls is enlarged to include withdrawing cash. Since the control set for Problem (5) is larger than the control set for Problem (4), then the optimal value of the objective function in Problem (5) can never be larger than the optimal objective function of Problem (4).*

Here we give an intuitive interpretation of the objective function in Problem (5). In the context of an investor saving for retirement, we can imagine that W^* is a target value of (real) wealth at the retirement date $t = T$. This objective function penalizes the expected quadratic shortfall with respect to the target W^*. Note that we do not penalize final wealth which exceeds W^*. In other words, we measure risk only in terms of undershooting the target wealth W^*. This is similar in spirit to the use of an upside wealth constraint suggested in (Donnelly et al., 2015). For a discussion concerning target based strategies for DC pension plans, we refer the reader to (Vigna, 2014; Menoncin and Vigna, 2017).

Remark 2 (Time consistency)**.** *There is considerable confusion in the literature about pre-commitment mean-variance strategies. These strategies are commonly criticized for being time inconsistent (Basak and Chabakauri, 2010; Bjork et al., 2014). However, the pre-commitment optimal policy can be found by solving problem (5), which can be determined by dynamic programming, and hence is time consistent when viewed as minimizing expected quadratic shortfall with respect to a fixed target W^*. Consequently, when determining the time consistent optimal strategy for problem (5), we obtain, as a by-product, the optimal mean variance pre-commitment solution. Further insight has been provided in (Vigna, 2017) and (Vigna, 2014; Menoncin and Vigna, 2017). As noted in (Cong and Oosterlee, 2016), the pre-commitment strategy can be seen as a strategy consistent with a fixed investment target, but not with a risk aversion attitude. Conversely, a time-consistent strategy has a consistent risk aversion attitude, but is not consistent with respect to an investment target. We contend that consistency with a target is more natural for DC pension investment strategies.*

We note that Problem (5) does not uniquely specify the optimal controls. Suppose that one of the assets is a risk free bond with interest rate r. Let

$$Q(t_\ell) = \sum_{j=\ell+1}^{j=N-1} e^{-r(t_j - t_\ell)} q(t_j) \tag{6}$$

be the discounted future contributions as of time t_ℓ. If

$$(W(t_n^- + q(t_n)) > W^* e^{-r(T-t_n)} - Q(t_n), \tag{7}$$

then an optimal strategy is to (i) invest $\left(W^* e^{-r(T-t_n)} - Q(t_n)\right)$ in the risk-free asset; and (ii) invest the remainder $\left(W^* e^{-r(T-t_i)} - Q(t_i)\right)$ in any long positions in the stock and bond. This strategy remains optimal since, when equation (7) holds at time t_i^-, then $\mathbf{E}[(\min(W(T) - W^*, 0))^2] = 0$ (Dang and Forsyth, 2016). As is common in the literature, we refer to the amount $W(t_n^-) + q(t_n) - \left(W^* e^{-r(T-t_n)} - Q(t_n)\right)$ as free or surplus cash (Bauerle and Grether, 2015). In the following sections, we describe a tie-breaking strategy which ensures that, if a risk-free asset is available, we invest the surplus in the risk-free asset.

4 A Data Driven Approach Solving a Single Optimization Problem

Problem (5) is an N stage stochastic optimization problem, which suffers from the curse of dimensionality. When the number of assets $M \leq 3$, the optimal control can be determined by solving a Hamilton-Jacobi-Bellman (HJB) equation (Dang and Forsyth, 2014; Forsyth and Labahn, 2018). Unfortunately, for a larger M, a sample based approximation is necessary and some approximate dynamic programming algorithm needs to be used.

Assume that the price $\vec{S}(t_n)$ follows a Markov process and the objective function $g(W(T))$ is regarded as a cost/reward function for the control policy. The asset allocation problem (5) can be solved approximately by reinforcement learning (RL)/backward dynamic programming (DP) methods, for which samples from the Markov process are specified (Bertsekas and Tsitsiklis, 1996). This gives rise to a finite horizon and continuous space optimization problem, which can be solved with an approximate DP/RL approach, via either forward or backward iterations, using a policy iteration or value iteration method. Bellman's principle is the crucial foundation for both types of methods since it converts the optimal selection of a sequence of decisions to a sequence of selections of decisions. In particular, value iteration algorithms search for optimal value functions, which are then used to compute optimal policies. Policy iteration methods, on the other hand, iteratively improve controls and the value function of the current control is determined and used to compute new policies.

Instead of following the typical approach to solve (5) sequentially using Bellman's equation, we propose to solve this multi-stage optimization problem directly from sample paths by seeking a global function $\vec{p}(F(t_n)) \in \Re^M$ of the state $F(t_n)$, for $n = 0,1\ldots,N-1$, where $F(t_n)$ is a feature vector, representing the information needed to determine the control at t_n, including minimally the current wealth at time t_n and time to go $T-t_n$. The objective is to minimize a performance measure $g(W(T))$ based on the terminal wealth at T. In other words, instead of selecting N control functions in (5), which are multiple functions of current wealths, e.g., in (4) and (5), we seek a single function $\vec{p}(\cdot)$ of the feature vector $F(t_n)$, with time explicitly included as a feature variable of time-to-go $T-t_n$. By using a global model $\vec{p}(\cdot)$ to represent controls at different time t_n through a time-to-go feature variable, we leverage continuity of the control with respect to time, which enhances learning efficiency. In addition, solving a global control model in a single optimization, instead of using the Bellman's equation, avoids error propagation, which is unavoidable in the iterative time stepping process. Indeed our computational results, based on both synthetic and real data, will validate the proposed approach.

Let $\vec{p}(F(t))$ be the vector of controls at time t, which depends on the feature vector $F(t)$. At each rebalancing time t_n, the control function $\vec{p}(F(t_n)) \in \Re^M$ yields fractions of the total wealth to be allocated to M assets. In other words, we let $\vec{\rho}_n(W(t_n)) = \vec{p}(F(t_n))$ and the amount invested in asset m is

$$W_m(t_n^+) \quad = \quad \vec{p}(F(t_n))_m W(t_n^+)\ , \tag{8}$$

so that the the vector of asset wealths at t_n^+, is

$$\vec{p}(F(t_n))W(t_n^+) = \vec{p}(F(t_n))(W(t_n^-)+q(t_n))\ . \tag{9}$$

Hence the wealth amount vector, after rebalancing at t_n, is $\vec{p}(F(t_n))W(t_n^+)$.

Given a control sequence $\vec{p}(F(t_0)),\ldots,\vec{p}(F(t_{N-1}))$, the dependence of the terminal wealth $W(T)$ on the control $\vec{p}(F(t_0)),\ldots,\vec{p}(F(t_{N-1}))$ now becomes,

$$\begin{array}{ll} \text{for} & n = 0,1,\ldots,N-1 \\ & W(t_n^+) = W(t_n^-)+q(t_n) \\ & W(t_{n+1}^-) = \vec{p}(F(t_n))^T \vec{R}(t_n) W(t_n^+) \\ & \qquad = \left(\vec{p}(F(t_n))^T \vec{R}(t_n)\right)(W(t_n^-)+q(t_n)) \\ \text{end} & \end{array} \tag{10}$$

where $\vec{R}(t_n)$ is the (stochastic) vector return on assets in (t_n^+, t_{n+1}^-). The final wealth $W(T) = W(t_N^-)$. For $t \in (t_n^+, t_{n+1}^-)$, $W(t)$ follows a stochastic process determined by $\vec{S}_t$ and hence the feature vector $F(t)$ is also stochastic.

Corresponding to (5), we have the following optimal investment problem seeking a global control function,

$$\begin{array}{rl} \min\limits_{\vec{p}(\cdot)} & g(W(T)) \\ \text{subject to} & 0 \le \vec{p}(F(t_n)) \le 1, \quad n = 0,1,\ldots,N-1, \\ & \mathbf{1}^T \vec{p}(F(t_n)) = 1\ , \end{array} \tag{11}$$

where the bound constraints specify no shorting and no leverage respectively, which would be common in practice.

Solving the multi-dimensional stochastic dynamic programming problem (11) remains computationally challenging when the number of asset M is large, particularly when we are interested in solving a long

horizon investment problem, e.g., $N = 30$ years. Note that the feature vector $F(t)$ takes on continuous values, and that the control $\vec{p}(F(t))$ depends on the feature $F(t)$. Hence, in order to use stochastic dynamic programming, we need to sample from the continuous state space $F(t)$. Of course, practical problems also necessitate honouring the constraints in Problem (11). In the learning context, $F(t)$ could be any features, representing relevant information available at t, which are used to train the model.

When solving the multi-asset problem (11) with $M > 3$, a sample based approach is likely the only viable computationally feasible approach. The idea is then to learn the optimal control function $\vec{p}(\cdot)$ based on the available sample paths, which come from either simulations of a parametric model or more interestingly, return samples observed directly from the market. In the latter case, we are then learning the optimal control directly from the market, bypassing an intermediate parametric modeling step, which has been the common practice in financial modeling.

Specifically, assume that a set of L return sample paths $\{\vec{R}^{(j)}(t_n), n = 1, \cdots, N, j = 1, \cdots, L\}$ are given. Let $\vec{p}^{(j)}(F(t_n))$ denote the allocation at time t_n along the j^{th} path. This yields the sample optimization problem below, arising from equation (11) based on samples,

$$\begin{aligned} \min_{\{\vec{p}(F^{(j)}(t_0)),\vec{p}(F^{(j)}(t_1)),\ldots,\vec{p}(F^{(j)}(t_{N-1}))\}} \quad & \frac{1}{2}\bar{g}(W^{(1)}(T),\ldots,W^{(L)}(T)) \\ \text{subject to} \quad & 0 \le \vec{p}(F^{(j)}(t_n)) \le 1, \quad n = 0,1,\ldots,N-1,\ j = 1,\ldots,L, \\ & \mathbf{1}^T \vec{p}(F^{(j)}(t_n)) = 1, \quad n = 0,1,\ldots,N-1,\ j = 1,\ldots,L\,, \end{aligned} \tag{12}$$

where $W^{(j)}(T)$ depends on the control as shown in (10). Here the objective function $\bar{g}(\cdot)$ is the objective function in (5) augmented with a small regularization,

$$\bar{g}(W^{(1)}(T),\ldots,W^{(L)}(T)) = \frac{1}{L}\sum_{j=1}^{L}\left(\min(W^{(j)}(T) - W^*, 0)\right)^2 + \frac{\lambda}{L}\sum_{j=1}^{L} W^{(j)}(T)$$

with $\lambda > 0$ a small constant, e.g., $\lambda = 10^{-6}$. The regularization is introduced to resolve the ambiguity of the objective function by forcing investment of surplus cash into the risk-free asset.

Unfortunately, the stochastic optimization problem (12) solves for the controls at the rebalancing time along each path. It has $O(MNL)$ variables, along with $O(MNL)$ equality and inequality constraints, which becomes computationally challenging to solve in practice since L is typically very large. Furthermore, the solution only provides the control values $\vec{p}(F^{(j)}(t_0)),\ldots,\vec{p}(F^{(j)}(t_{N-1}))$ along each path. It does not immediately supply controls at different paths.

Given a finite set of sample paths, approximations are necessary to computationally solve this multi-stage optimization problems. Finding a good approximation model for the control, suitable for effective learning, is the key for computing a solution efficiently and accurately. In the RL or DP approaches, a recursive procedure is followed based on dynamic programming principle to simplify the computation of the optimal control. We propose to directly solve a global scenario optimization Problem (12) by deploying a novel approximation model $\vec{p}(F(t))$, using a neural network to represent controls at any time t, $0 \le t \le T$, and any feature state $F(t)$.

In the proposed approximate model structure, the control is a function of the feature vector which has time to go $T - t$ as a variate. Specifically, let $F(t_n) \in \Re^d$ denote values of feature variables at t_n. In the simplest case, the feature vector variables can be the wealth at t_n and time to go, i.e., $F(t_n) = \{W(t_n), T - t_n\}$. More generally, the feature variables can include additional market information, e.g., market implied volatility or historical realized volatilities. Additionally features can include individual investor's information, e.g., taxes, which personalizes the allocation solution. We note that neural networks have been used to represent policies and/or value functions in RL/DP, see, e.g., (Vinyals et al., 2017).

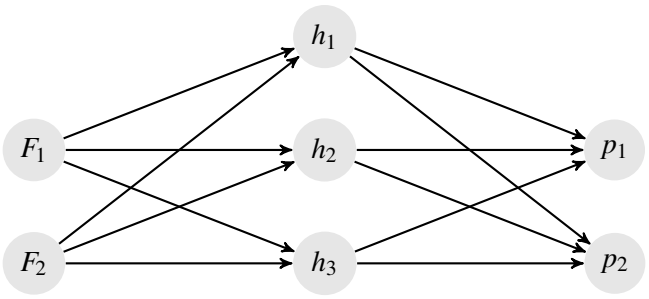


FIGURE 1: *A 2-layer NN representing control functions*

Specifically, we represent the control $\vec{p}(F(t))$, for any t, as the output of an NN and feature $F(t)$ as inputs. At the rebalance time t_n, the state variable $F(t_n)$ includes minimally the current wealth $W(t_n)$ and time-to-target-horizon $T-t_n$. Thus, in the training, the single NN model outputs the control $\vec{p}(F(t_n))$ at t_n, $n=0,\cdots,N-1$, for any feature state $F(t_n)$. In particular, we consider a parsimonious two layer NN, depicted in Figure 1, to represent the control function. We remark that shallow learning is also found to outperform deep learning for asset pricing in (Gu et al., 2018). We also note that good results are obtained in (Hejazi and Jackson, 2016) with an NN containing only one hidden layer (shallow learning). Suppose that input features $F(t) \in \Re^d$ and there are a total of M assets. Assume that there are l nodes in the hidden layer of NN. This NN is represented by the weights for the input layer and the weights for the output layer.

Assume that $h \in \Re^l$ is the output of the hidden layer. Let the matrix $z \in \Re^{dl}$ be the weights from the inputs $F(t_n) \in \Re^d$ to the hidden nodes $h \in \Re^l$. Using the sigmoid activation function,

$$\sigma(u) = \frac{1}{1+e^u}$$

we have

$$h_j(F(t_n)) = \sigma(F_i(t_n)\, z_{ij}), \tag{13}$$

where here (and in the following) we use the summation convention, i.e. summation over repeated indices is implied. For example

$$F_i\, z_{ij} \equiv \sum_{i=1}^{i=d} F_i z_{ij}\ ,\ j=1,\ldots,l,$$

where F_i is the i^{th} component of the feature vector F.

At the output layer, we use the logistic sigmoid. Let the matrix $x \in \Re^{lM}$ be the weights for the output layer. For the m^{th} asset, $1 \le m \le M$, the holding is given by the m^{th} component of the output $\vec{p}$, i.e.,

$$\vec{p}_m(F(t_n)) = \frac{e^{x_{\ell m} h_\ell(F(t_n))}}{\sum_i e^{x_{\ell i} h_\ell(F(t_n))}}, \qquad 1 \le m \le M. \tag{14}$$

Using this representation, the controls automatically satisfy

$$0 \le \vec{p}(F(t_n)) \le 1, \qquad \vec{p}(F(t_n))^T \mathbf{1} = 1\ .$$

If there are more stringent upper bounds on the assets, they can be similarly incorporated into the representation through appropriate scalings.

Let $F^{(j)}(t_n)$ be the state variables at $t = t_n$ along sample path j, $j \in \{1,\ldots,L\}$. The approximation to the optimal learning problem (12) becomes

$$\begin{aligned} \min_{z\in\Re^{dl},x\in\Re^{lM}} \quad & \frac{1}{2}\bar{g}(W^{(1)}(T),\ldots,W^{(L)}(T)) \\ \text{subject to} \quad & \vec{p}_m(F^{(j)}(t_n)) = \frac{e^{x_{\ell m}h_\ell(F^{(j)}(t_n))}}{\sum_i e^{x_{\ell i}h_\ell(F^{(j)}(t_n))}}, \quad m = 1,\ldots,M,\ j = 1,\ldots,L,\ n = 0,\ldots,N-1 \\ & h_\ell(F^{(j)}(t_n)) = \sigma(F_i^{(j)}(t_n)\ z_{i\ell}), \quad \ell = 1,\ldots,l,\ j = 1,\ldots,L,\ n = 0,\ldots,N-1, \end{aligned} \tag{15}$$

where we remind the reader that again we use the summation convention. We note that, in the proposed approximation model, the constraints in (15) are explicitly satisfied, further simplifying (5) to an unconstrained problem (15).

In contrast to the large and constrained problem (12), with a dimension of $O(MNL)$ and $O(MNL)$ constraints, we note that (15) is an unconstrained optimization problem with $l(d+M)$ variables, the entries of the weight matrices z and x. Consequently the dimension of the unconstrained learning optimization problem no longer depends on the number of scenarios L or the number of time steps N. Rather it depends only on the NN model structure.

Let $\bar{W}(T)$ be the column scenario wealth vector below

$$\bar{W}(T) = [W^{(1)}(T),\ldots,W^{(L)}(T)]' .$$

Using unconstrained smooth optimization methods to solve (15) requires evaluation of the objective function and its derivative with respect to z and x. Following (10), each objective function evaluation costs $O(l(d+M)NL)$, or $O(L)$ assuming a fixed NN model structure and fixed rebalancing schedule.

For the gradient evaluation, we note that

$$\nabla_{x,z}\bar{g} = \left(\nabla_{\bar{W}}\bar{g}\right)\left(\nabla_{x,z}\bar{W}(T)\right)$$

where here the gradient $\nabla_{x,z}$ is with respect to the weight x and z.

Following (10), we have the iterative computation below for the Jacobian matrices $\nabla_{x,z}\bar{W}(T)$,

$$\begin{aligned} \nabla_{x,z}\bar{W}(t_{n+1}^-) &= \nabla_{x,z}\left(\sum_m R_m(t_n)\vec{p}_m(F(t_n))(\bar{W}(t_n^-)+q(t_n))\right) \\ &= \left(\sum_m \vec{R}_m(t_n)\nabla_{x,z}\vec{p}_m(F(t_n))\right)(\bar{W}(t_n^-)+q(t_n)) + \left(\sum_m \vec{R}_m(t_n)\vec{p}_m(F(t_n))\right)\nabla_{x,z}W(t_n^-) \end{aligned}$$

and

$$\nabla_{x,z}\bar{W}(t_0^-) = \left(\sum_m R_m(t_0)\nabla_{x,z}\vec{p}_m(F(t_0))\right)(\bar{W}(t_0^-)+q(t_0))$$

Further simplifying notations by dropping dependence on the feature $F(t_n)$, the gradient of the control $\vec{p}$ is given below, for $1 \le q \le l$, $1 \le m \le M$,

$$\begin{aligned} \nabla_{x_{qm}}\vec{p}_m &= \overbrace{(1-\vec{p}_m)\vec{p}_m h_q}^{no\ sum} \\ \nabla_{x_{qj}}\vec{p}_m &= -\vec{p}_m\frac{e^{x_{\ell j}h_\ell}}{\sum_k e^{x_{\ell k}h_\ell}}h_q = -\vec{p}_m\vec{p}_j h_q, \quad j \ne m . \end{aligned}$$

Using the definition of the sigmoid function, we also have that, for $1 \le j \le l$, $1 \le q \le d$,

$$\begin{aligned}\nabla_{z_{qj}}\vec{p}_m &= \frac{e^{x_{\ell m}h_\ell}}{\sum_k e^{x_{\ell k}h_\ell}}\left(x_{jm}-\frac{\sum_{k=1}^M e^{x_{\ell k}h_\ell}x_{jk}}{\sum_{k=1}^M e^{x_{\ell k}h_\ell}}\right)\overbrace{\nabla_{z_{qj}}h_j}^{no\ sum}\\ &= -\vec{p}_m\left(x_{jm}-\frac{\sum_{k=1}^M e^{x_{\ell k}h_\ell}x_{jk}}{\sum_k e^{x_{\ell k}h_\ell}}\right)\frac{e^{F_\ell z_{\ell j}}}{(1+e^{F_\ell z_{\ell j}})^2}F_q\ .\end{aligned}$$

The gradient evaluation costs $O(l(d+M)NL)$, and the Hessian computation costs $O(l^2(d+M)^2LN)$, using a finite difference of the gradient. Given the function/gradient/Hessian, solving the trust region subproblem requires $O((l(d+M))^3)$. Since the dimension of the optimization problem $l(d+M)$ is small for the problem considered in this paper, e.g., $l(d+M) = 15$ for three assets, function/gradient/Hessian evaluations become the dominant cost and the usage of a trust region subproblem is a reasonable computational choice.

5 Ground truth: a low dimensional parametric return model

Given a parametric model of the underlying stochastic process, for a small number of random factors, we can solve (15) by computing the solution of the associated Hamilton Jacobi Bellman (HJB) equation (Dang and Forsyth, 2014).

We first validate the proposed data driven NN approach (15) for determining the optimal controls for Problem (5) by comparing the solution to that from solving the HJB equation under a parametric model, assuming a portfolio with two assets. Let $S(t)$ and $B(t)$ respectively denote the *amounts* invested in the risky and risk-free assets at time t, $t \in [0,T]$. In practice, we will suppose that $S(t)$ represents the amount invested in a broad stock market index, while $B(t)$ is the amount invested in short term default-free government bonds.

In general, the amounts $S(t)$ and $B(t)$ will depend on the investor's strategy over time, including contributions, withdrawals, and portfolio rebalances, as well as changes in the unit prices of the assets. Suppose for the moment that the investor does not take any action with respect to the controllable factors. We refer to this as the absence of control. This situation applies in between the rebalancing times. In this case, we assume that $S(t)$ follows a jump diffusion process. Recall that $t^- = t-\epsilon, \epsilon \to 0^+$, i.e. t^- is the instant of time before t, and let ξ be a random number representing a jump multiplier. When a jump occurs, $S(t) = \xi S(t^-)$. Allowing discontinuous jumps lets us explore the effects of severe market crashes on the risky asset holding. We assume that ξ follows a double exponential distribution (Kou, 2002; Kou and Wang, 2004). If a jump occurs, p_{up} is the probability of an upward jump, while $1-p_{up}$ is the chance of a downward jump. The density function for $y = \log\xi$ is

$$f(y) = p_{up}\eta_1 e^{-\eta_1 y}\mathbf{1}_{y\geq 0} + (1-p_{up})\eta_2 e^{\eta_2 y}\mathbf{1}_{y<0}. \tag{16}$$

For future reference, note that

$$E[y=\log\xi] = \frac{p_{up}}{\eta_1} - \frac{(1-p_{up})}{\eta_2} \quad , \quad E[\xi] = \frac{p_{up}\eta_1}{\eta_1-1} + \frac{(1-p_{up})\eta_2}{\eta_2+1}\ . \tag{17}$$

In the absence of control, $S(t)$ evolves according to

$$\frac{dS(t)}{S(t^-)} = (\mu - \lambda E[\xi-1])\,dt + \sigma\,d\mathcal{Z} + d\left(\sum_{i=1}^{\pi_t}(\xi_i-1)\right), \tag{18}$$

where μ is the (uncompensated) drift rate, σ is the volatility, $d\mathcal{Z}$ is the increment of a Wiener process, π_t is a Poisson process with positive intensity parameter λ, and ξ_i are i.i.d. positive random variables having distribution (16). Moreover, ξ_i, π_t, and $\mathcal{Z}$ are assumed to all be mutually independent.

μ	σ	λ	p_{up}	η_1	η_2
Real CRSP Cap-Weighted Index					
.08889	.14771	.32222	0.27586	4.4273	5.2613
Real CRSP Equal-Weighted Index					
.11833	.16633	.40000	.33334	3.6912	4.5409

TABLE 1: *Estimated annualized parameters for double exponential jump diffusion model. Cap-weighted and equal-weighted CRSP indexes, deflated by the CPI. Sample period 1926:1 to 2015:12.*

In the absence of control, we assume that the dynamics of the amount B_t invested in the risk-free asset are

$$dB(t) = rB(t)\,dt, \tag{19}$$

where r is the (constant) risk-free rate. This is obviously a simplification of the real bond market. We remind the reader that, ultimately, our NN method is entirely data driven, and will be based on bootstrapped stock and bond indexes.

With this parametric model of stock prices, we can determine the optimal solution to Problem (5) using dynamic programming. This in turn results in a nonlinear Hamilton-Jacobi-Bellman PDE. We use the methods described in (Dang and Forsyth, 2014; Forsyth and Labahn, 2018) to determine the provably optimal solution (to within a tolerance). At each rebalancing date t_n, at each value of $W(t_n^-)$, we check to see if equation (7) holds, which indicates that surplus cash is available. In this case we withdraw the surplus cash from the portfolio, and invest the remainder in the risk-free asset. We also invest the surplus cash in the risk-free asset. This is an optimal strategy, as described in Dang and Forsyth (2016).

6 Data

Our data is from the Center for Research in Security Prices (CRSP) on a monthly basis over the *1926:1-2015:12* period.[1] Our base case tests use the CRSP 3-month Treasury bill (T-bill) index for the risk-free asset and the CRSP cap-weighted total return index for the risky asset. This latter index includes all distributions for all domestic stocks trading on major U.S. exchanges. As an alternative case for additional illustrations, we replace the above two indexes by a 10-year Treasury index and the CRSP equal-weighted total return index.[2] All of these various indexes are in nominal terms, so we adjust them for inflation by using the U.S. CPI index, also supplied by CRSP. We use real indexes since investors saving for retirement should be focused on real (not nominal) wealth goals.

In the case of the parametric model, i.e., processes (18) and (19), we use the methods in Dang and Forsyth (2016) to calibrate the process parameters. We use a threshold technique (Cont and Mancini, 2011) to identify jump frequency and distribution, and the methods in (Dang and Forsyth, 2016) to determine the remaining parameters. Annualized estimated parameters for both the cap-weighted and equal-weighted indexes are provided in Table 1.

[1] More specifically, results presented here were calculated based on data from Historical Indexes, ©2015 Center for Research in Security Prices (CRSP), The University of Chicago Booth School of Business. Wharton Research Data Services was used in preparing this article. This service and the data available thereon constitute valuable intellectual property and trade secrets of WRDS and/or its third-party suppliers.

[2] The 10-year Treasury index was constructed from monthly returns from CRSP back to 1941. The data for 1926-1941 were interpolated from annual returns in Homer and Sylla (2005).

	Real 3-month T-bill Index	Real 10-year Treasury Index
Mean return	.00827	.02160
Volatility	.019	.065

TABLE 2: *Mean annualized real rates of return for bond indexes (*$\log[B(T)/B(0)]/T$*). Volatilities (annualized) computed using* log *returns. We show the volatilities for information only, the parametric model uses a constant average real interest rate. Sample period 1926:1 to 2015:12.*

Table 2 shows the average real interest rates for the 3-month T-bill and 10-year U.S. Treasury indexes over the entire sample period from 1926 to 2015.

7 Bootstrap resampling

In order to use the proposed data driven NN approach, we will sample directly from the historical data. A single bootstrap resampled path is constructed as follows. Suppose the investment horizon is T years. We divide this total time into k blocks of size b years, so that $T = kb$. We then select k blocks at random (with replacement) from the historical data (from both the deflated stock and bond indexes). Each block starts at a random month. We then form a single path by concatenating these blocks. Since we sample with replacement, the blocks can overlap. To avoid end effects, the historical data is wrapped around, as in the circular block bootstrap (Politis and White, 2004; Patton et al., 2009). We repeat this procedure for many paths. The sampling is done in blocks in order to account for possible serial dependence effects in the historical time series. The choice of blocksize is crucial and can have a large impact on the results (Cogneau and Zakalmouline, 2013). We simultaneously sample the real stock and bond returns from the historical data. This introduces random real interest rates in our samples, in contrast to the constant interest rates assumed in the synthetic market tests and in the determination of the optimal controls.

To reduce the impact of a fixed blocksize and to mitigate the edge effects at each block end, we use the stationary block bootstrap (Politis and White, 2004; Patton et al., 2009). The blocksize is randomly sampled from a geometric distribution with an expected blocksize $\hat{b}$. The optimal choice for $\hat{b}$ is determined using the algorithm described in Patton et al. (2009). This approach has also been used in other tests of portfolio allocation problems recently (e.g. Dichtl et al., 2016). Calculated optimal values for $\hat{b}$ for the various indexes are given in Table 3.

When we use our resampling method in the proposed data driven NN approach, we will simultaneously sample the same block from all data sets (i.e. equity indices and bond indices). Clearly, Table 3 shows that the optimal blocksize varies amongst the time series in question. It is, therefore, not clear which is the best choice of blocksize for use in our simultaneous resampling method. As a result, we will carry out tests with a variety of blocksizes, in the ranges suggested by Table 3.

8 Numerical Results: Parametric Model

In this section, we give results based on the parametric model described in Section 5. Optimal controls will be computed using both the HJB equation method and the data driven NN technique (15). All examples will assume the scenario given in Table 4.

Data series	Optimal expected block size $\hat{b}$ (months)
Real 3-month T-bill index	50.1
Real 10-year Treasury index	4.7
Real CRSP cap-weighted index	1.8
Real CRSP equal-weighted index	10.4

TABLE 3: *Optimal expected blocksize $\hat{b} = 1/v$ when the blocksize follows a geometric distribution $Pr(b = k) = (1-v)^{k-1}v$. The algorithm in Patton et al. (2009) is used to determine $\hat{b}$.*

	Base Case	Alternative Case
Investment horizon (years)	30	30
Equity market index	Cap-weighted	Equal-weighted
Risk-free asset index	3-month T-bill	10-year Treasury
Initial investment W_0 (\$)	0.0	0.0
Real investment each year (\$)	10.0	10.0
Rebalancing interval (years)	1	1

TABLE 4: *Input data for examples. Cash is invested at $t = 0,1,\ldots,29$ years. Market parameters are provided in Tables 1 and 2*

.

8.1 Optimal control: HJB equation

The optimal control is computed by solving an HJB equation as described in Dang and Forsyth (2014); Forsyth and Labahn (2018).

8.1.1 HJB equation, base case: CRSP value weighted index and 3-month T-bill

As a first example, we consider the base case input data summarized in Table 4. An investor with a horizon of 30 years makes real contributions each year of \$10, allocated between the CRSP cap-weighted and 3-month T-bill indexes and rebalanced annually.

We first use a constant proportion strategy, where we rebalance to a fixed weight in stocks at each rebalancing date ($p = 0.5$), and determine the expected value of the terminal real wealth for this strategy. We then use this expected value as a constraint and determine the optimal strategy which solves problem (5). In other words, the value of W^* in the objective function of (5) is determined by setting $\mathbf{E}(W_T)$ to be the same as for the constant weight strategy. We compute and store the optimal strategy (from the HJB solution).

We evaluate the performance of the various strategies using Monte Carlo simulation, where we simulate the market using the SDEs in equations (18-19). We use the constant weight strategy and the optimal strategy determined from the HJB solution. Table 5 compares the results for these strategies. Due to the highly skewed distribution function for the final wealth W_T, the most relevant statistics are the median and the probability of shortfall. Both of these statistics are highly favourable for the optimal strategy.

8.1.2 HJB equation: alternative case, CRSP equal-weighted index and 10-year Treasury index

To provide a second example for the parametric model, we use alternative assets. In particular, as indicated in Table 4, we replace the CRSP cap-weighted index with its equal-weighted counterpart, and we substitute

				Probability of Shortfall	
Strategy	$E[W_T]$	$Median[W_T]$	$std[W_T]$	$W_T < 500$	$W_T < 600$
Constant proportion ($p = 0.5$)	705	630	350	.28	.45
Optimal	705	775	153	.12	.17

TABLE 5: *Parametric model results from 160,000 Monte Carlo simulation runs for base case input data given in Table 4 and corresponding parameters from Tables 1 (threshold) and 2. The expected surplus cash flow for the optimal adaptive strategy is 16.7, assumed to be invested in the risk-free asset.*

				Probability of Shortfall	
Strategy	$E[W_T]$	$Median[W_T]$	$std[W_T]$	$W_T < 700$	$W_T < 900$
Constant proportion ($p = 0.5$)	1082	875	852	.33	.52
Optimal	1082	1238	338	.17	.23

TABLE 6: *Parametric model results from 160,000 Monte Carlo simulation runs for alternative case input data given in Table 4 and corresponding parameters from Tables 1 (threshold) and 2. The expected surplus cash flow for the optimal adaptive strategy is 51, assumed to be invested in the risk-free asset.*

the 10-year Treasury bond index for the 3-month Treasury bill index. See Tables 1 and 2 for relevant corresponding parameter estimates. We retain the same assumptions regarding investment horizon, rebalancing frequency, and real cash contributions as for the base case Table 4.

Table 6 presents the results for the constant proportion, and optimal adaptive strategies. The results are very similar in qualitative terms to those seen earlier for the base case in Table 4, though investing in these two assets leads to a terminal wealth distribution with a higher mean and standard deviation relative to using the cap-weighted index and 3-month T-bills. Note that the median of W_T is higher than the mean for the optimal strategy, and the probabilities of shortfall are much reduced compared with the constant proportion strategy.

8.2 Optimal NN controls

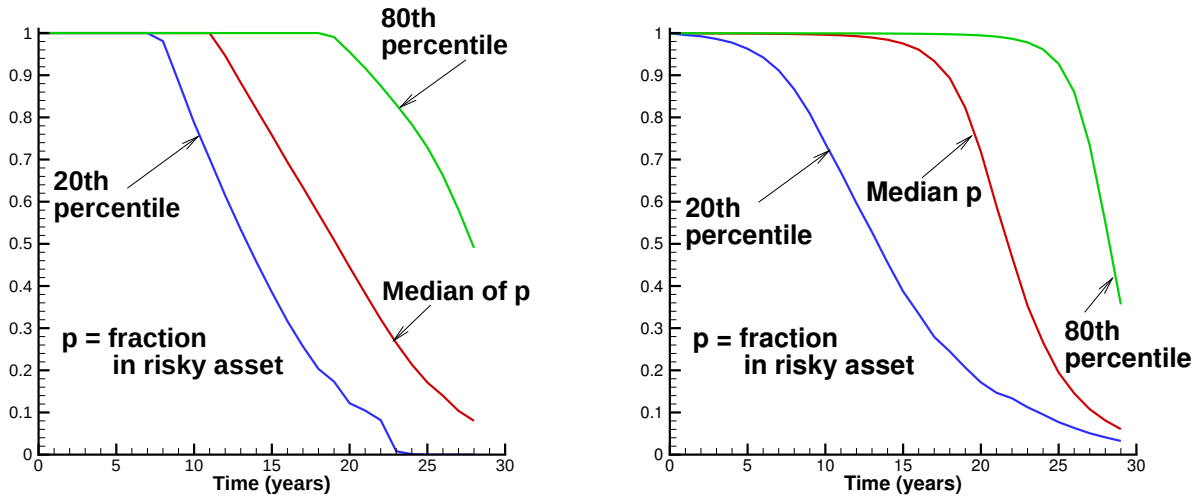
We compute the approximate optimal control by solving the proposed NN approximation (15) to the original problem (5), as described in Section 4. Specifically, we have found that a parsimonious NN model with one hidden layer of three nodes is sufficient to produce nearly optimal performance. In this investigation, the feature vector consists of simply the current wealth and time-to-go. We explicitly compute the objective function and its first order derivatives but approximate the Hessian matrix using a finite difference approximation. The NN approximation problem (15) is solved using a trust region method (Coleman and Li, 1996). For the NN learning, it is known that standardizing features is important for efficient learning. For (15), however, the feature state, wealth $W(t_n)$, changes with the control iterate during the optimization process. Hence we cannot standardize features based on standard deviations of wealth values of the current iteration. Instead, at each rebalance time t_n, $n = 1, \ldots, N-1$, we use standard deviations associated with the constant proportion strategy to scale the wealth feature variable.

8.3 Base case NN controls: CRSP cap-weighted index and 3-month T-bill

We generate $L = 160{,}000$ i.i.d. random return paths for the parametric jump model calibrated from the historic market data, as described in §5, using equations (18) and (19). We solve the NN learning optimization

Training Error on Synthetic Data : Market Cap Weighted					
Strategy	$E(W_T)$	$std(W_T)$	$median(W_T)$	$Pr(W_T < 500$	$Pr(W_T < 600)$
constant proportion ($p = .5$)	705	350	630	0.28	0.45
NN adaptive	705	159	782	0.13	0.18
Optimal	705	153	782	0.12	0.17

TABLE 7: *Training comparison base case data, Table 4. Value weighted CRSP and 3 month T-bill. Training carried out using 160,000 sampled paths. Compare with Table 5.*



(a) *HJB control* (b) *NN control*

FIGURE 2: *Percentiles of the control (fraction in equity index), NN and HJB equation solution, base case example, Table 4 (cap-weighted CRSP index and 3-month T-bill).*

problem (15).

Table 7 presents training performance comparisons of the optimal controls obtained for the cap-weighted index and 3-month T-bill two asset base case. Comparing to the performance of the optimal controls computed by solving HJB equation in Tables 5, we observe that the proposed NN approach (15) achieves excellent performance using a parsimonious NN model with $d = 2$, $l = 3$, $M = 2$ for the 2-asset case, totaling only 12 parameters for z and x. Considering that there is always error due to sampling, the slight suboptimality arising from the NN approximation seems to be quite acceptable.

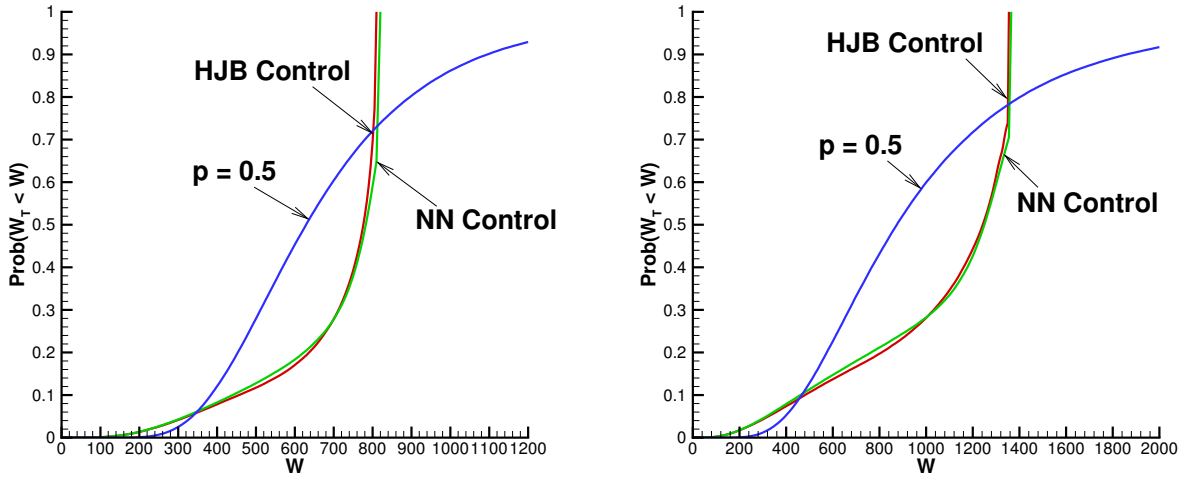
Corresponding to the base case, Table 4, Figure 2 compares the percentiles of the NN controls with those computed using the HJB equation. We observe that the curves from the NN and HJB controls are qualitatively similar, indicating similar investment strategies are obtained.

8.4 Alternative case NN controls: CRSP equal-weighted index and 10-year Treasury index

We also compare the NN controls for the alternative case, CRSP equal-weighted index and 10-year treasury index, in Table 8. Again, it is remarkable that our parsimonious NN model, trained on sampled data, is able to get very close to the optimal results.

Training Error on Synthetic Data: Equal Weighted					
Strategy	$E(W_T)$	$std(W_T)$	$median(W_T))$	$Pr(W_T) < 700$	$Pr(W_T < 900)$
constant proportion ($p = .5$)	1082	852	875	0.33	0.52
NN adaptive	1082	349	1250	0.18	0.24
Optimal	1082	338	1238	0.17	0.23

TABLE 8: *Training comparison alternative case data, Table 4. Equal weighted CRSP and 10 year treasury. Training carried out using 160000 sampled paths. Compare with Table 6.*



(a) *Cap-weighted CRSP, 3 month T-bill.*

(b) *Equal-weighted CRSP, 10 year Treasury.*

FIGURE 3: *Comparison of the cumulative distribution functions for the control computed using HJB equation and Neural Network (NN). 160,000 MC samples used. The constant weight strategy ($p = 0.5$) also shown. Base case (Figure 3(a)) and alternative case (Figure 3(b)) data sets, as in Table 4. $W^* = 806$ (base case) and $W^* = 1355$ (alternative case). Surplus cash is not included in the distribution functions for the HJB and NN controls.*

8.5 Comparison of Cumulative Distribution Functions: Parametric Market Model

Figure 3 shows the cumulative distribution functions, computed using 160,000 Monte Carlo simulations, for both the base case assets and the alternative case. The controls were computed by (i) solving the HJB equation (which gives the optimal strategy), (ii) using the NN approximation and (iii) using the constant weight strategy $p = 0.5$, i.e., rebalance to a fraction of 0.5 in equities at each rebalancing date.

Figure 3 shows that the control computed using a very parsimonious NN model can reproduce very closely the entire distribution function of the terminal wealth generated using the optimal HJB equation control.

8.6 Test performance of the NN control in the historic market scenarios

We compute and store the NN strategy, based on sampled data, which is generated using the parametric model described by equations (18) and (19). We then test this learned control on the bootstrapped historical market data.

Tables 9 and 10 report test performance of the strategies computed using the simulated returns and present their performance on on the bootstrapped resampled historical data. We show the results obtained

Test Error: Market Cap Weighted					
Strategy	$E(W_T)$	$std(W_T)$	$median(W_T)$	$Pr(W_T < 500$	$Pr(W_T < 600)$
Expected Blocksize $\hat{b}$ = 0.5 years					
constant proportion (p = .5)	680	278	627	0.28	0.46
NN adaptive	699	154	766	0.12	0.19
Expected Blocksize $\hat{b}$ = 1 years					
constant proportion (p = .5)	680	277	626	0.28	0.45
NN adaptive	698	154	765	0.13	0.19
Expected Blocksize $\hat{b}$ = 2 years					
constant proportion (p = .5)	677	264	628	0.27	0.46
NN adaptive	704	144	764	0.11	0.18
Expected Blocksize $\hat{b}$ = 5 years					
constant proportion (p = .5)	675	250	635	0.27	0.44
NN adaptive	713	136	776	0.10	0.16
Expected Blocksize $\hat{b}$ = 8 years					
constant proportion (p = .5)	666	231	631	0.26	0.44
NN adaptive	718	127	777	0.08	0.15
Expected Blocksize $\hat{b}$ = 10 years					
constant proportion (p = .5)	667	223	634	0.25	0.44
NN adaptive	721	123	779	0.08	0.14

TABLE 9: *NN control computed and stored based on sampling the parametric market model. Tests carried out using 10000 bootstrap resamples of historical data. Cap-weighted CRSP index and 3-month T-bill market data. Compare to the training performance in Table 7.*

by rebalancing to a constant equity weight $p = 0.5$. Since the optimal choice of blocksize is not clear (being quite different for the stock index and the bond index) we show results for a range of reasonable blocksizes.

Comparing the result in Tables 9 and 10 to the results in Tables 7 & 8 respectively, we observe that the test performance comparisons with the constant proportion strategies are similar to the training performance comparisons, suggesting robustness of the NN control. In particular, in all cases, the NN control (trained using the parametric model) has a higher $Median(W_T)$ and smaller probabilities of shortfall, compared to the constant weight strategy.

8.7 Performance of the optimal controls directly learned from historic market data

We now abandon the parametric market model, and operate directly on the historical market data. Since we have no *optimal* solution for this case, we contrast performance from the NN control with the performance from the constant weight strategy, rebalancing to a constant weight $p = 0.5$ in equities at each rebalancing date.

We emphasize that a key advantage of the proposed optimal NN control framework is that it allows direct learning of the controls from the market data, bypassing the parametric modeling all together. In §8.3, we have seen that the performance of the optimal NN controls, trained from samples determined from the parametric market model, is comparable to that from the optimal HJB controls.

Furthermore, in §8.6, the performance of the NN control (trained from parametric model samples) using the bootstrapped historical market data is shown to be robust. Here we report the training performance of the optimal NN controls directly from the market data. As an additional robustness check, we compute and store the NN controls trained on resampled historical market data, and then test the performance of these learned controls on samples from the parametric market model.

Table 11 presents training performance comparisons of the NN controls directly learned from the historical cap-weighted index and 3-month T-bill market data. A range of blocksizes for the bootstrap resampling is reported. For each blocksize, we determine W^* such that $E[W_T]$ for the NN control is the same as $E[W_T]$ for the constant proportion strategy. The NN controls for each blocksize are stored, and then used as controls

Test Error: Market Equal Weighted					
Strategy	$E(W_T)$	$std(W_T)$	$median(W_T)$	$Pr(W_T) < 700$	$Pr(W_T < 900)$
Expected Blocksize $\hat{b}$ = 0.5 years					
constant proportion (p = .5)	1005	586	868	0.33	0.53
NN adaptive	1057	331	1177	0.18	0.26
Expected Blocksize $\hat{b}$ = 1 years					
constant proportion (p = .5)	1005	586	868	0.33	0.53
NN adaptive	1057	331	1177	0.18	0.26
Expected Blocksize $\hat{b}$ = 2 years					
constant proportion (p = .5)	961	465	865	0.31	0.54
NN adaptive	1082	293	1167	0.13	0.22
Expected Blocksize $\hat{b}$ = 5 years					
constant proportion (p = .5)	936	382	869	0.29	0.54
NN adaptive	1111	261	1182	0.10	0.18
Expected Blocksize $\hat{b}$ = 8 years					
constant proportion (p = .5)	921	346	867	0.28	0.54
NN adaptive	1124	243	1185	0.08	0.16
Expected Blocksize $\hat{b}$ = 10 years					
constant proportion (p = .5)	919	336	870	0.27	0.54
NN adaptive	1132	233	1190	0.07	0.16

TABLE 10: *NN control computed and stored based on sampling the parametric market model. Tests carried out using 10000 bootstrap resamples of historical data. Equal-weighted CRSP index and 10 year treasury market data. Compare to the training performance in Table 7.*

for Monte Carlo simulations based on the parametric market model. Table 12 reports the results of these tests. Comparing these tables to Table 7 and 9 respectively, we note a striking similarity of training and test performance comparisons. There is some degradation in performance as the blocksize increases, when we test on the parametric model, but $median(W_T)$ is always larger for the NN control, compared to the constant weight strategy. However, this is a good test of robustness of the controls. Small blocksizes simulate an i.i.d. process, which is the underlying assumption of the parametric market model. Increasing the blocksize causes a deviation from the i.i.d. assumption, and hence is stress test for the control learned with a large blocksize.

We carry out a similar sequence of tests using the alternative market data, based on an equal-weight index, and 10 year treasuries. Table 13 presents training performance comparisons of the NN controls directly learned from the historical equal-weighted index and 10-year treasury market data. Table 14 reports the test performance of these NN controls, learned from the market data, on the simulated data from the corresponding parametric market model.

Comparing Tables 13 and 14 to Tables 8 and 10 respectively, we note that the same similarity in training and test performance comparisons. In Table 14, we note that there is some degradation in performance, in terms of $E(W_T)$ as the blocksize is increased, but, as discussed previously, this can be traced to the fact that use of large blocksizes causes departures from the i.i.d. process assumed in the parametric model. In any case, the important statistic here is $median(W_T)$, which is always larger for the NN control compared to the constant weight strategy. However, once again, the NN control is quite robust. Of course, this suggests that if we believe that the true market process is better modeled as i.i.d., then use of smaller blocksizes will obviously yield superior results.

8.8 Three assets

In addition to the clear advantage of learning the strategy directly from market data, and avoiding the need to specify and calibrate a parametric market mode, we can, with trivial modification, use the NN approach to solve high dimensional asset allocation problems. Recall that the overall complexity of the proposed NN

Training Error: Market Cap Weighted					
Strategy	$E(W_T)$	$std(W_T)$	$median(W_T)$	$Pr(W_T) < 500$	$Pr(W_T < 600)$
Expected Blocksize $\hat{b}$ = 0.5 years					
constant proportion (p = .5)	680	278	627	0.28	0.46
NN adaptive	680	142	764	0.13	0.20
Expected Blocksize $\hat{b}$ = 1 years					
constant proportion (p = .5)	680	277	626	0.28	0.45
NN adaptive	680	142	762	0.13	0.20
Expected Blocksize $\hat{b}$ = 2 years					
constant proportion (p = .5)	677	264	628	0.27	0.46
NN adaptive	677	127	748	0.12	0.19
Expected Blocksize $\hat{b}$ = 5 years					
constant proportion (p = .5)	675	250	635	0.27	0.44
NN adaptive	675	112	731	0.10	0.16
Expected Blocksize $\hat{b}$ = 8 years					
constant proportion (p = .5)	666	231	631	0.26	0.44
NN adaptive	666	97	711	0.08	0.15
Expected Blocksize $\hat{b}$ = 10 years					
constant proportion (p = .5)	667	223	634	0.25	0.44
NN adaptive	667	92	707	0.08	0.14

TABLE 11: *Training performance comparisons on the market data: The NN controls are learned directly from the cap-weighted index and 3-month T-bill market data. The expected blocksize is used in the bootstrap resampling algorithm, 10,000 samples used. Bootstrap resampling also used for the* $p = 0.5$ *control.*

Test Error: Market Cap Weighted						
Strategy	$E(W_T)$	$std(W_T)$	$median(W_T)$	$Pr(W_T) < 500$	$Pr(W_T < 600)$	$\hat{b}$ years
constant proportion (p = .5)	705	350	630	0.28	0.45	NA
NN adaptive	682	146	765	0.13	0.20	0.5
NN adaptive	679	147	763	0.14	0.21	1.0
NN adaptive	671	141	747	0.13	0.20	2.0
NN adaptive	658	149	740	0.16	0.23	5.0
NN adaptive	643	140	716	0.16	0.22	8.0
NN adaptive	640	138	711	0.16	0.22	10.0

TABLE 12: *The NN controls are learned directly from the cap-weighted index and 3-month T-bill market data, using bootstrap resampling, with the expected blocksize* $\hat{b}$ *indicated. These controls are then used in Monte Carlo simulations based on the parametric market model.*

Training Error: Market Equal Weighted					
Strategy	$E(W_T)$	$std(W_T)$	$median(W_T)$	$Pr(W_T) < 700$	$Pr(W_T < 900)$
Expected Blocksize $\hat{b}$ = 0.5 years					
constant proportion (p = .5)	1005	586	868	0.33	0.53
NN adaptive	1005	289	1205	0.18	0.27
Expected Blocksize $\hat{b}$ = 1 years					
constant proportion (p = .5)	1005	586	868	0.33	0.53
NN adaptive	1005	289	1205	0.18	0.27
Expected Blocksize $\hat{b}$ = 2 years					
constant proportion (p = .5)	961	465	865	0.31	0.54
NN adaptive	961	207	1081	0.13	0.24
Expected Blocksize $\hat{b}$ = 5 years					
constant proportion (p = .5)	936	382	869	0.29	0.54
NN adaptive	936	151	1007	0.09	0.20
Expected Blocksize $\hat{b}$ = 8 years					
constant proportion (p = .5)	921	346	867	0.28	0.54
NN adaptive	921	124	972	0.07	0.18
Expected Blocksize $\hat{b}$ = 10 years					
constant proportion (p = .5)	919	336	870	0.27	0.54
NN adaptive	919	114	964	0.07	0.17

TABLE 13: *Training performance comparisons on the market data: The NN controls are learned directly from the equal-weighted CRSP index and 10 year treasury market data. The expected blocksize is used in the bootstrap resampling algorithm, 10,000 samples used. Bootstrap resampling also used for the* $p = 0.5$ *control.*

Test Error: Market Equal Weighted						
Strategy	$E(W_T)$	$std(W_T)$	$median(W_T)$	$Pr(W_T) < 700$	$Pr(W_T < 900)$	$\hat{b}$ years
constant proportion (p = .5)	1082	852	875	0.33	0.52	NA
NN adaptive	1011	303	1205	0.18	0.26	0.5
NN adaptive	1011	303	1205	0.18	0.26	1.0
NN adaptive	939	254	1081	0.18	0.25	2.0
NN adaptive	889	226	1007	0.17	0.25	5.0
NN adaptive	865	214	972	0.17	0.24	8.0
NN adaptive	860	212	964	0.17	0.24	10.0

TABLE 14: *The NN controls are learned directly from the equal-weighted CRSP index and 10 year treasury market data, using bootstrap resampling, with the expected blocksize* $\hat{b}$ *indicated. These controls are then used in Monte Carlo simulations based on the parametric market model.*

Training with Expected Blocksize $\hat{b}$ = 0.5 years: Market Cap Weighted					
Strategy	$E(W_T)$	$std(W_T)$	$median(W_T)$	$Pr(W_T) < 500$	$Pr(W_T < 600)$
Expected Blocksize $\hat{b}$ = 0.5 years					
constant proportion (p = (0.6,0.1,0.3))	860	450	758	0.18	0.31
NN adaptive	860	264	986	0.15	0.20
Expected Blocksize $\hat{b}$ = 1 years					
constant proportion (p = (0.6,0.1,0.3))	857	429	761	0.18	0.30
NN adaptive	865	264	994	0.15	0.20
Expected Blocksize $\hat{b}$ = 2 years					
constant proportion (p = (0.6,0.1,0.3))	849	414	758	0.18	0.30
NN adaptive	867	254	986	0.13	0.19
Expected Blocksize $\hat{b}$ = 5 years					
constant proportion (p = (0.6,0.1,0.3))	841	383	769	0.17	0.29
NN adaptive	878	246	994	0.12	0.18
Expected Blocksize $\hat{b}$ = 8 years					
constant proportion (p = (0.6,0.1,0.3))	827	350	769	0.16	0.28
NN adaptive	886	236	996	0.11	0.16
Expected Blocksize $\hat{b}$ = 10 years					
constant proportion (p = (0.6,0.1,0.3))	826	337	772	0.16	0.27
NN adaptive	893	230	1002	0.10	0.15

TABLE 15: *Performance comparisons using the cap-weighted index, 3-month T-bill and 10-year treasury: the first data set with the expected blocksize $\hat{b}$ = 0.5 years is the training set. Test performance comparisons are reported on the remaining data sets.*

control approach is $O(\max((l(d+M))^3, (l^2(d+M)^2LN))$ where d is the number of features, l is the number of nodes, M is the number of assets, and N is the number of timesteps.

In this section, we report performance on a three asset portfolio. As a point of comparison, we consider a base case strategy to be a constant proportion strategy, which rebalances to the weights: 60% in the equity index, 10% in the 3-month T-bill, and 30% in the 10-year Treasury. As before, we select W^* so that $E[W_T]$ is the same for the NN control and for the constant weight strategy, for the training set. All tests in this section are based on bootstrap historical market simulations.

Table 15 presents performance comparisons using the cap-weighted index, 3-month T-bill and 10-year treasury. The data set with the expected blocksize $\hat{b} = 0.5$ years is used to train the NN control model and its test performance is reported on the other five data sets with different expected blocksizes. Table 16 reports the same comparisons except that the data set with the expected blocksize $\hat{b} = 10$ years is used to train the NN control model.

We observe that, in the three-asset case, the training and testing performances of the NN controls continue to dominate that of the constant proportion strategy, achieving a performance enhancement level similar to the two-asset case. In addition, performance comparison is robust. In all cases, $median(W_T)$ for the NN control is larger than for the constant weight strategy, with reduced probability of shortfall.

Finally, we replace the the cap-weighted index by the CRSP equal-weighted index, but continue with the 3-month T-bill and 10 year treasury for the fixed income components of the portfolio. As before, we compare to a constant proportion strategy with weights 60 : 10 : 30 in the equity index, 3-month T-bill and 10-year treasury. We consider this alternative three-asset case and learn the NN controls from the market data directly. Table 17 presents performance comparisons using these portfolio components. The data set with the expected blocksize $\hat{b} = 0.5$ years is used to train the NN control model and its test performance is reported on the other five data sets with different expected blocksizes. Table 16 reports the same comparisons except that the data set with the expected blocksize $\hat{b} = 10$ years is used to train the NN control model. We observe similar performance comparisons as for the cap-weighted index case.

In Figure 4, we compare cumulative distributions of the NN control on the training data ($\hat{b} = .5$) and testing data ($\hat{b} = 10$) and compare them to that from the constant weights strategy ($\hat{b} = 10$). We observe

Training with Expected Blocksize $\hat{b}$ = 10 years: Market Cap Weighted					
Strategy	$E(W_T)$	$std(W_T)$	$median(W_T)$	$Pr(W_T) < 500$	$Pr(W_T < 600)$
Expected Blocksize $\hat{b}$ = 0.5 years					
constant proportion (p = (0.6,0.1,0.3))	860	450	758	0.18	0.31
NN adaptive	794	216	934	0.14	0.20
Expected Blocksize $\hat{b}$ = 1 years					
constant proportion (p = (0.6,0.1,0.3))	857	429	761	0.18	0.30
NN adaptive	798	215	941	0.14	0.19
Expected Blocksize $\hat{b}$ = 2 years					
constant proportion (p = (0.6,0.1,0.3))	849	414	758	0.18	0.30
NN adaptive	801	205	935	0.13	0.18
Expected Blocksize $\hat{b}$ = 5 years					
constant proportion (p = (0.6,0.1,0.3))	841	383	769	0.17	0.29
NN adaptive	812	195	941	0.11	0.17
Expected Blocksize $\hat{b}$ = 8 years					
constant proportion (p = (0.6,0.1,0.3))	827	350	769	0.16	0.28
NN adaptive	821	185	941	0.10	0.15
Expected Blocksize $\hat{b}$ = 10 years					
constant proportion (p = (0.6,0.1,0.3))	826	337	772	0.16	0.27
NN adaptive	826	180	941	0.09	0.14

TABLE 16: *Performance comparisons using the cap-weighted index, 3-month T-bill and i 10-year treasury: the last data set with the expected blocksize $\hat{b}$ = 10 years is the training set. Test performance comparisons are reported on the remaining data sets.*

Training with Expected Blocksize $\hat{b}$ = 0.5 years: Equal Weighted					
Strategy	$E(W_T)$	$std(W_T)$	$median(W_T)$	$Pr(W_T) < 700$	$Pr(W_T < 900)$
Expected Blocksize $\hat{b}$ = 0.5 years					
constant proportion (p = (0.6,0.1,0.3))	1152	792	950	0.30	0.46
NN adaptive	1152	398	1351	0.19	0.26
Expected Blocksize $\hat{b}$ = 1 years					
constant proportion (p = (0.6,0.1,0.3))	1152	789	950	0.29	0.46
NN adaptive	1153	398	1352	0.19	0.26
Expected Blocksize $\hat{b}$ = 2 years					
constant proportion (p = (0.6,0.1,0.3))	1079	594	945	0.27	0.46
NN adaptive	1174	358	1327	0.15	0.23
Expected Blocksize $\hat{b}$ = 5 years					
constant proportion (p = (0.6,0.1,0.3))	1034	456	952	0.23	0.45
NN adaptive	1203	321	1327	0.11	0.19
Expected Blocksize $\hat{b}$ = 8 years					
constant proportion (p = (0.6,0.1,0.3))	1012	398	954	0.21	0.44
NN adaptive	1215	301	1327	0.09	0.17
Expected Blocksize $\hat{b}$ = 10 years					
constant proportion (p = (0.6,0.1,0.3))	1007	376	961	0.21	0.43
NN adaptive	1227	288	1334	0.08	0.16

TABLE 17: *Performance comparisons using the equal-weighted index, 3-month T-bill and 10-year treasury: the first data set with the expected blocksize $\hat{b}$ = 0.5 years is the training set. Test performance comparisons are reported on the remaining data sets.*

Training with Expected Blocksize $\hat{b}$ = 10 years: Equal Weighted					
Strategy	$E(W_T)$	$std(W_T)$	$median(W_T)$	$Pr(W_T) < 700$	$Pr(W_T < 900)$
Expected Blocksize $\hat{b}$ = 0.5 years					
constant proportion (p = (0.6,0.1,0.3))	1152	792	950	0.30	0.46
NN adaptive	929	245	1075	0.18	0.27
Expected Blocksize $\hat{b}$ = 1 years					
constant proportion (p = (0.6,0.1,0.3))	1152	789	950	0.29	0.46
NN adaptive	931	245	1075	0.18	0.26
Expected Blocksize $\hat{b}$ = 2 years					
constant proportion (p = (0.6,0.1,0.3))	1079	594	945	0.27	0.46
NN adaptive	959	209	1075	0.13	0.22
Expected Blocksize $\hat{b}$ = 5 years					
constant proportion (p = (0.6,0.1,0.3))	1034	456	952	0.23	0.45
NN adaptive	987	176	1075	0.09	0.18
Expected Blocksize $\hat{b}$ = 8 years					
constant proportion (p = (0.6,0.1,0.3))	1012	398	954	0.21	0.44
NN adaptive	1001	157	1075	0.07	0.15
Expected Blocksize $\hat{b}$ = 10 years					
constant proportion (p = (0.6,0.1,0.3))	1007	376	961	0.21	0.43
NN adaptive	1007	146	1075	0.06	0.14

TABLE 18: *Performance comparisons using the equal-weighted index, 3-month T-bill and 10-year treasury: the last data set with the expected blocksize $\hat{b} = 10$ years is the training set. Test performance comparisons are reported on the remaining data sets.*

that, from these distributions, the performance of the NN control dominates that of the constant strategy in three-asset cases. We also note that here the test performance, which corresponds to a larger blocksize, has a better performance than the training data set, likely due to serial correlations.

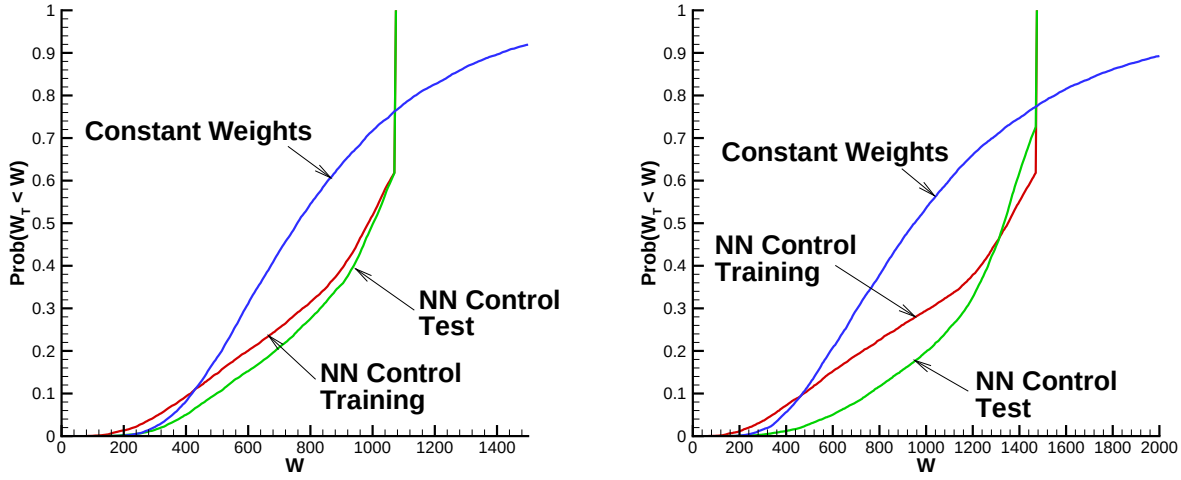
9 Conclusions

If realistic constraints are applied to a DC pension plan asset allocation policy (e.g. no leverage, no shorting, discrete rebalancing), then in order to determine the optimal control, the HJB equation must be solved numerically. Solving a multistage stochastic optimal investment problem is challenging when the number of state variables becomes large, since the curse of dimensionality comes into play when using the HJB PDE. In addition, a PDE method requires a parametric stochastic model to be specified and calibrated to market data. This is the dominant approach used computational finance for the last few decades.

In this paper, we consider solving the multi-stage optimal decision problem arising from a finite number of scenarios. When the set of scenarios are i.i.d. samples of a parametric stochastic model, the scenario optimization problem is an approximation to the parametric model based optimization problem. If the scenarios are market price observations (or generated from a resampling approach), solving the scenario optimization directly for the controls leads to a market data driven optimal investment strategy. The resulting scenario optimization problem is large since the number of unknowns is $O(NLM)$, where N is the number of periods, M is the number of assets and L is the number of scenario paths.

Instead of following a backward timestepping process using Bellman's principle to tackle the multi-stage stochastic optimization problem, we propose to solve the large scale scenario optimization problem directly. To overcome the challenge of a large dimensional optimization problem, we propose using a parsimonious NN model to represent the control at all different rebalancing times, using the features of current wealth and time-to-go as the inputs to the NN model. The resulting optimization problem has a dimension $O(l(d+M))$, where l is the number of hidden nodes, d is the number of features, and M is the number of assets.

We first validate the proposed NN optimization approach by comparing the training performance with the benchmark strategy computed by solving an HJB PDE equation, assuming a parametric market model.



(a) *Cap-weighted CRSP, 3 month T-bill, 10 year Treasury*

(b) *Equal-weighted CRSP, 3 month T-bill, 10 year Treasury.*

FIGURE 4: *Comparison of the cumulative distribution functions for the control computed using Neural Network (NN) and constant strategy. The constant weight strategy is* $p = (0.6, 0.1, 0.3)$. *Surplus cash is not included in the distribution functions for the NN controls.*

The HJB solution generates the provably optimal strategy in this case. Our computational results (using the parametric model) demonstrate that, using a parsimonious 2-layer NN with three hidden nodes, we obtain performance on par with that from the optimal HJB control. This validation is conducted for a 2-asset base case, with two portfolio composition scenarios: a cap-weighted real CRSP index and real 3-month T-bill, and an alternative case with an equal-weighted real CRSP index and real 10-year treasury.

In addition, we report test performance of the NN control on the bootstrap resampling market data sets and observe performance comparison similar to the training based on simulated model scenarios. This demonstrates robustness of the learned NN controls.

The proposed method provides a way of learning controls directly from the market data without first estimating a parametric model. To investigate this, we learn NN controls from bootstrap resamples of the historical market data, with different expected blocksizes, and examine both the training performance comparisons and test performance comparisons on the parametric market model data sets. We observe that the NN optimal controls from the market data sets generate superior training and testing performances compared to that of the constant proportion strategies, at a similar level to the parametric model based HJB method.

In addition, we also consider 3-asset investment problems with two choices of assets: (i) a cap-weighted CRSP index, 3-month bill and 10-year treasury or (ii) an equal-weighted CRSP index, 3-month bill and 10-year treasury. We observe similar levels of training and test improvement from the NN controls over those of the constant proportion 60:10:30 strategy.

Overall, the performance of the NN dynamically adaptive strategy significantly dominates that of the constant proportion strategies, achieving higher median values of the terminal wealth, and smaller probabilities of shortfall, compared with the constant weight strategies.

One main technical question which arises is the choice of blocksize to be used for training the NN control. If we believe that serial correlation is important going forward, then use of a larger blocksize would be recommended. On the other hand, a smaller blocksize would be appropriate if we view the processes as essentially i.i.d. We use a joint resampling process, where we use the same blocksize for

all the components of the portfolio. This preserves the historical correlations amongst the components of the portfolio. However, it is not clear how to select an optimal blocksize in this situation. We intend to investigate the choice of blocksize in the context of NN training with bootstrap resampling in the future. For now, we suggest the conservative approach of using a small blocksize, since this seems to be quite robust.

We remark that the proposed NN optimization framework makes it computationally feasible to solve problems with a larger number of assets, and without specifying a parametric market model. Although we have focused on a DC pension plan example, it is straightforward to apply these techniques to other objective functions. For example, a problem of much current interest is optimal asset allocation for defined benefit (DB) plans, with the objective function being to minimize shortfall with respect to a target funding ratio.

10 Acknowledgments

This research was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) grants: RGPIN-2017-03760 and RGPIN-2014-03978. The authors have no conflicts of interest to report.

References

Basak, S. and G. Chabakauri (2010). Dynamic mean-variance asset allocation. *Review of Financial Studies 23*, 2970–3016.

Bauerle, N. and S. Grether (2015). Complete markets do not allow free cash flow streams. *Mathematical Methods of Operations Research 81*, 137–146.

Bertsekas, D. P. and J. N. Tsitsiklis (1996). *Neuro-dynamic Programming*. Belmont, Massachusetts: Athena Scientific.

Bjork, T., A. Murgoci, and X. Zhou (2014). Mean variance portfolio optimization with state dependent risk aversion. *Mathematical Finance 24*, 1–24.

Cogneau, P. and V. Zakalmouline (2013). Block bootstrap methods and the choice of stocks for the long run. *Quantitative Finance 13*, 1443–1457.

Coleman, T. and Y. Li (1996). An interior, trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on Optimization 6*, 418–445.

Cong, F. and C. Oosterlee (2016). Multi-period mean variance portfolio optimization based on Monte-Carlo simulation. *Journal of Economic Dynamics and Control 64*, 23–38.

Cont, R. and C. Mancini (2011). Nonparametric tests for pathwise properties of semimartingales. *Bernoulli 17*, 781–813.

Dang, D.-M. and P. Forsyth (2016). Better than pre-commitment mean-variance portfolio allocation strategies: a semi-self-financing Hamilton-Jacobi-Bellman equation approach. *European Journal of Operational Research 250*, 827–841.

Dang, D. M. and P. A. Forsyth (2014). Continuous time mean-variance optimal portfolio allocation under jump diffusion: A numerical impulse control approach. *Numerical Methods for Partial Differential Equations 30*, 664–698.

Dichtl, H., W. Drobetz, and M. Wambach (2016). Testing rebalancing strategies for stock-bond portfolos across different asset allocations. *Applied Economics 48*, 772–788.

Donnelly, C., R. Gerrard, M. Gullen, and J. Nielsen (2015). Less is more: increasing retirment gains by using an upside terminal wealth constraint. *Insurance: Mathematics and Economics 64*, 259–267.

Forsyth, P. and G. Labahn (2018). ϵ-monotone Fourier methods for optimal stochastic control in finance. to appear, Journal of Compuational Finance.

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

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

Gu, S., R. Kelly, and D. Xu (2018). Empirical asset pricing via machine learning. SSRN:3159577.

Guan, G. and Z. Liang (2015). Mean-variance efficiency of DC pension plan under stochastic interest rate and mean-reverting returns. *Insurance: Mathematics and Economics 61*, 99–109.

He, L. and Z. Liang (2013). Optimal investment strategy for the DC plan with return of premimums clauses in a mean-variance framework. *Insurance: Mathematics and Economics 53*, 643–649.

Hejazi, S. and K. Jackson (2016). A neural network approach to efficient valuation of large portfolios of variable annuities. *Insurance: Mathematics and Economics 70*, 169–181.

Homer, S. and R. Sylla (2005). *A History of Interest Rates*. New York: Wiley.

Kou, S. and H. Wang (2004). Option pricing under a double exponential jump diffusion model. *Management Science 50*, 1178–1192.

Kou, S. G. (2002). A jump-diffusion model for option pricing. *Management Science 48*, 1086–1101.

Li, D. and W.-L. Ng (2000). Optimal dynamic portfolio selection: Multiperiod mean-variance formulation. *Mathematical Finance 10*, 387–406.

Lioui, A. (2013). Time consistent vs. time inconsistent dynamic asset allocation: some utility cost calculations for mean variance preferences. *Journal of Economic Dynamics and Control 37*, 1066–1096.

Menoncin, F. and E. Vigna (2017). Mean-variance target based optimisation for defined contribution pension schemes in a stochastic framework. *Insurance: Mathematics and Economics 76*, 172–184.

Patton, A., D. Politis, and H. White (2009). Correction to: Automatic block-length selection for the dependent bootstrap. *Econometric Reviews 28*, 372–375.

Politis, D. and H. White (2004). Automatic block-length selection for the dependent bootstrap. *Econometric Reviews 23*, 53–70.

Vigna, E. (2014). On efficiency of mean-variance based portfolio selection in defined contribution pension schemes. *Quantitative Finance 14*, 237–258.

Vigna, E. (2017). Tail optimality and preferences consistency for intertemporal optimization problems. Working papers, Collegio Carlo Alberto, Universita di Torino.

Vinyals, O., T. Edwards, S. Bartunov, P. Georgiev, A. S. Vezhnevets, M. Yeo, A. Makhzani, H. KÃijttler, J. Agapiou, J. Schrittwieser, J. Quan, S. Gaffney, S. Petersen, K. Simonyan, T. Schaul, H. van Hasselt, D. Silver, T. Lillicrap, K. Calderone, P. Keet, A. Brunasso, D. Lawrence, A. Ekermo, J. Repp, and R. Tsing (2017). StarCraft II a new challenge for reinforcement learning. arXiv:1708.04782.

Wu, H., L. Zhang, and H. Chen (2015). Nash equilibrium strategies for a defined contribution pension management. *Insurance: Mathematics and Economics 62*, 202–214.

Yao, H., Z. Yang, and P. Chen (2013). Markowitz's mean-variance defined contribution pension fund management under inflation: A continuous time model. *Insurance: Mathematics and Economics 53*, 851–863.

Zhou, X. Y. and D. Li (2000). Continuous-time mean-variance portfolio selection: A stochastic LQ framework. *Applied Mathematics and Optimization 42*, 19–33.