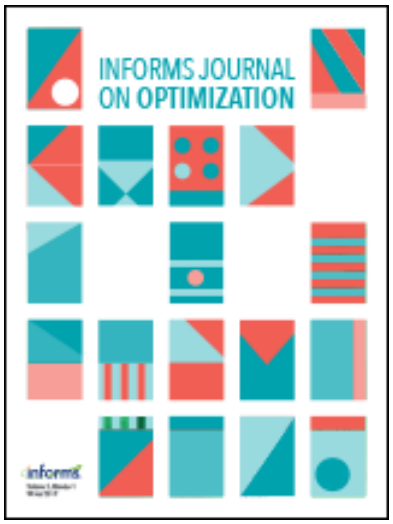


INFORMS Journal on Optimization

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Portfolio Optimization Under Regime Switching and Transaction Costs: Combining Neural Networks and Dynamic Programs

Xiaoyue Li, John M. Mulvey

To cite this article:
Xiaoyue Li, John M. Mulvey (2021) Portfolio Optimization Under Regime Switching and Transaction Costs: Combining Neural Networks and Dynamic Programs. INFORMS Journal on Optimization 3(4):398-417. https://doi.org/10.1287/ijoo.2021.0053

http://pubsonline.informs.org/journal/ijoo

INFORMS JOURNAL ON OPTIMIZATION
Vol. 3, No. 4, Fall 2021, pp. 398–417
ISSN 2575-1484 (print), ISSN 2575-1492 (online)

Portfolio Optimization Under Regime Switching and Transaction Costs: Combining Neural Networks and Dynamic Programs

Xiaoyue Li,[a] John M. Mulvey[a]

[a] Department of Operations Research and Financial Engineering, Princeton University, Princeton, New Jersey 08540

Contact: xiaoyuel@princeton.edu, https://orcid.org/0000-0002-6227-8281 (XL); mulvey@princeton.edu, https://orcid.org/0000-0002-4290-0870 (JMM)

Received: August 20, 2020
Revised: November 22, 2020; December 30, 2020
Accepted: January 4, 2021
Published Online in Articles in Advance: October 21, 2021

https://doi.org/10.1287/ijoo.2021.0053

Abstract. Multiperiod financial models provide superior capabilities over single-period myopic approaches but, in general, suffer from the curse of dimensionality. Prominent features include transaction costs, rebalancing gains, intermediate cashflows, and short- versus long-term trade-offs. In this paper, we propose and test an algorithm combining dynamic programming with a recurrent neural network. The dynamic program provides advanced starts for the neural network. Empirical tests show the benefits of this novel strategy with optimizing a hidden Markov model in the presence of linear transaction costs. Test problems with 50–250 time steps and up to 11 risky assets are solved efficiently, relative to stand-alone dynamic programs or neural networks. The recurrent neural network addresses transaction costs within difficult multiperiod optimization models in polynomial run time.

Keywords: multiperiod portfolio optimization • neural networks • regime-based investment • dynamic programs • stochastic programs • finite-state Markov models • nonlinear programs • financial optimization

1. Introduction

Multiperiod stochastic financial planning models have been one of the most intricate and important problem domains in computational finance. Elegant as single-period models are, they fail to adequately address issues such as transaction costs and intermediate cash flows. In addition, long-term investors benefit by considering short-term versus long-term goals, as well as change in the underlying financial environment dynamics. These issues are, nonetheless, significant in real-world situations, and much effort has been made by researchers to address these barriers.

It is nontrivial to tackle these practically useful long-term planning problems (Birge 2007). For numerical methods, the state space grows at an exponential rate as the planning horizon lengthens along with the number of branches per time period and the number of risky assets—the curse of dimensionality. Even with today's computing power, it is still impractical to apply traditional concepts. One remedy is to take advantage of the structure of the stochastic control system and solve the problem analytically with the Hamilton-Jacobian-Bellman (HJB) equations. Such methods usually require a large set of assumptions. Famous examples include Merton (1969) and Samuelson (1969). The Merton problem (Merton 1969) has been a central topic in mathematical finance since it was posted. Merton's initial problem involves a continuous-time portfolio optimization under no transaction costs, where the asset prices follow geometric Brownian motion. Many extensions have been discovered by researchers, including adjusting the planning horizon, imposing transaction costs, changing the asset price dynamics, and so on. In this paper, we propose an efficient algorithm for finding a promising trading strategy in a market with multiple regimes that are not directly observable and where proportional transaction costs incur.

One of the most well-studied extensions to the original Merton problem is the case with linear transaction costs. Davis and Norman (1990) show that, under normative assumptions, the optimal trading strategy under linear transaction costs is to formulate a no-transaction wedge and to trade only when the current position is out of the wedge. Dumas and Luciano (1991) conduct a solution of the portfolio management problem under a finite horizon with constant relative risk-aversion objective function. Bertsimas and Pachamanova (2008) propose a robust optimization method for multiperiod portfolio management in the presence of transaction costs by incorporating future asset returns as uncertain coefficients in the optimization problem. In general, as Mulvey et al. (2001, p. 51–58.) mentions, in solving multistage asset-allocation problems, when linear transaction incurs, "the form of the optimal policy is to establish a no-trade zone around each of the asset targets (proportions)." These studies enable us to solve our problem by focusing on identifying the boundaries of such a no-trade zone, instead of solving a complex optimization system.

Meanwhile, it is well established that market returns possess heavier tails than normal distributions. Thus, a model with two or more regimes can better describe the historical return patterns (Mulvey and Liu 2016). The market has vastly distinctive behavior under recession versus under expansion, and thus investors benefit from dynamic strategies of managing their portfolios. In fact, Nystrup et al. (2015) argue for the superiority of regime-based allocation over a static strategy. In particular, an example is provided to show that, even with no level of forecasting skills, one can benefit from having a regime-based strategy. Elliott and Siu (2009) propose a market model of one risky asset and one risk-free asset, with finite possible underlying regimes. Assuming that the investors can directly observe the current regime, they solve a Hamilton-Jacobian-Bellman system to get the optimal continuous-time trading strategy for an investor aiming to maximize the constant relative risk-aversion (CRRA) utility at the horizon. Zhou and Yin (2003) solve the optimal mean-variance portfolio in a market where the underlying regime is observable. Bauerle and Rieder (2004) tackle a similar problem, where the agents aim to maximize terminal utility at the end of a finite horizon. They model the market with a continuous-time finite-state Markov chain and assume that the agents have complete information. Their results show that, under no transaction costs and under CRRA utility, the optimal solution is a myopic strategy. Çanakoğlu and Özekici (2009) model the market with a discrete Markov process, where risky asset returns depend on the underlying state. They use a dynamic program to find an explicit solution for a multiperiod investor with exponential utility. Large assumptions are imposed: (i) there are no constraints on borrowing or shorting; (ii) no intermediate cash flows are allowed; and (iii) transaction costs are negligible. Liu (2014) integrates intermediate consumption in the model and finds an optimal investment-consumption strategy in order to maximize the expected discounted total utility of consumption and the expected discounted utility from terminal wealth. Sotomayor and Cadenillas (2009) find an explicit solution for an uncertain-horizon problem, where the agents maximize their expected total discounted hyperbolic absolute risk-aversion (HARA) utility of consumptions until bankruptcy. These models ignore transaction and market impact costs.

In the aforementioned studies, the investors are assumed to be able to observe the underlying regime of the market. However, in reality, the state of the market is usually labeled with a significant lag. Extensions have been made to the market models, and investors are interested in investment strategies when the regimes are not directly observable. The hidden Markov model (HMM) provides a simple yet effective framework for modeling financial data with time-varying patterns. It was proved to be empirically useful by successfully capturing the stock price change during the 2008 financial crisis. Birge (2012) uses a particle method to identify the hidden Markov process, serving as a robust estimation for the underlying process for a meaningful dynamic optimization. Bauerle and Rieder (2007) extend their previous study (Bauerle and Rieder 2004) by solving the optimal portfolio under an unobservable intensity process. A Wonham filter reduces the unobservable process to an observable one: the belief is updated continuously and is embedded in the state space. Çanakoğlu and Özekici (2012) extend their previous work by modeling the market with a hidden Markov process. Under identical assumptions as Çanakoğlu and Özekici (2009), they solve for the explicit solution under exponential utility, log utility, and power utility. Kolm and Ritter (2014) propose a framework to maximize total utility on a portfolio in a market with hidden Markov states. Transaction costs are assumed to be superlinear in this market. Palczewski et al. (2015) apply a numerical method to find a promising portfolio allocation in a market with state-dependent drift and transaction costs. A Kalman-Bucy filter deals with the unobservable regimes, which reduces the running time from $O(N^5)$ with a direct dynamic program to $O(N^{2.5})$, where N equals the number of time steps. The resulting algorithm provides satisfactory performance on maximizing exponential utility over a 40-year period, with a running-time of four days on a laptop. Nystrup et al. (2018) provide a practical method for dealing with dynamic portfolio management—instead of looking for the optimal strategy for the whole horizon, they attain a satisfactory strategy for a truncated look-ahead period and update the portfolio on a rolling basis. The model is extended to multiple risky assets in Nystrup et al. (2019).

For our study, we focus on optimizing a portfolio of assets under an unobservable regime-switching market where linear transaction costs incur. Our goal is to maximize the expected CRRA utility at the end of a finite planning horizon. The research on multiperiod portfolio problems often faces the curse of dimensionality. As the number of risky assets and the number of time steps increases, traditional methods such as dynamic programming would fail due to the exponentially growing state space. Machine learning methods, including neural networks, find their role in financial applications by serving as an efficient solution for the growth of state space. The application of neural networks in finance can be traced back to the 1990s. Kamijo and Tanigawa (1990) employ a recurrent neural network to identify "triangle" patterns in Japanese stock prices to find clues for trends of future prices changes. Such patterns are not clearly defined by rule-based approaches, yet neural networks achieve a promising accuracy in recognizing these patterns. Hejazi and Jackson (2016) illustrate the efficiency and effectiveness of neural networks in calculating the Greeks of variable annuity contracts. They train the neural

networks with a small set of variable annuity policies with pre-estimated Greeks, and the neural networks learn to evaluate the Greeks and output similar results as traditional methods but with much shorter running time. In addition, recent years have witnessed growing applications of neural networks in asset allocation problems. Casas (2001) utilizes an artificial neural network to predict the best performer out of three asset classes for the next period and allocates 100% to the predicted winner. Despite the lack of diversification, the out-of-sample test shows satisfying returns. Li and Forsyth (2019) study the allocation strategy for target-based defined contribution pension plans. When the assets follow a known parametric stochastic process, they find that the allocation learned by the neural networks is close to the Hamilton-Jacobi-Bellman solution. As they note, the neural network approach can be easily scaled to multiasset problems with high dimensions. Mulvey et al. (2020) employ a deep neural network to learn the allocation for a set of mean-reverting assets, in a world with linear transaction costs. They first solve the analytical solution with Hamilton-Jacobi-Bellman equations in the absence of transaction costs; then they use the analytical solution as a starting point for the neural network and learn the trading strategy when transaction costs incur. Following Mulvey et al. (2020), we take a two-step procedure in our study to deal with the asset allocation problem under regime switching and transaction costs: (1) first we start with a simpler problem with no transaction costs, so that dynamic programming provides optimal allocation within reasonable running time; (2) then we use the dynamic programming solution as a starting point and take advantage of neural networks to find the optimal no-trade zone.

The contributions of this paper are threefold. First, by combining dynamic programs and neural networks, we provide an efficient numerical method to solve a large multiperiod portfolio allocation problem under regime-switching market and transaction costs. Second, the performance of our combined method is shown to be close to optimum in a stylized case. To our knowledge, this is the first paper to carry out such comparison. Last, the superiority of the combined method opens up the possibility for more research on financial applications of generic method such as neural networks, provided that solutions to simplified subproblems are available via traditional methods.

The rest of the paper is organized as follows. Section 2 describes the multiperiod portfolio optimization problem with unobservable regime switching and linear transaction costs. In Section 3, we provide a promising trading strategy. Section 4 compares the performance of our strategy with an approximate optimal strategy learned by dynamic programming under a simple case. We find that our method achieves nearly optimal performance with a much shorter running time. The advantage in running time becomes more significant with larger test cases. In Section 5, empirical results with market data are provided. Our algorithm is able to offer satisfactory results when there are multiple risky assets in the market, as well as when there are more than two underlying possible regimes. Section 6 concludes.

2. Problem Setup

In this section, we fully describe the mechanics of a multiperiod portfolio optimization problem. There are $n \geq 1$ risky assets and one risk-free asset in the market, and the prices of risky assets follow correlated geometric Brownian motion with parameters depending on the underlying regime. The regime is not directly observable by the investors, but investors can infer the probability of being in certain regimes based on the observed asset returns. The investor may rebalance her portfolio at the end of each period in order to maximize their terminal utility but incurs linear transaction costs in the process.

2.1. General Multiperiod Portfolio Allocation Problem

In a general multiperiod portfolio optimization problem with T periods, an agent decides the allocation at the beginning of each period, in order to maximize a utility function with respect to expected return, volatility of return, as well as other potential reward and risk measures. The optimization problem has the form

$$\begin{aligned}
\underset{\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_{T-1} \in \mathbb{R}^n}{\text{Maximize}} \quad & \mathit{Utility}[Z_1, Z_2, \ldots] & & \\
\text{subject to} \quad & \mathbf{1}^T \boldsymbol{\pi}_t = 1 & & \forall t = 0, \ldots, T-1 \\
& W_t^{\rightarrow} = W_t(\boldsymbol{\pi}_t^T(1 + \boldsymbol{r}_t)) & & \forall t = 0, \ldots, T-1 \\
& \boldsymbol{\pi}_t^{\rightarrow} = \frac{\boldsymbol{\pi}_t \odot (1 + \boldsymbol{r}_t)}{\boldsymbol{\pi}_t^T(1 + \boldsymbol{r}_t)} & & \forall t = 0, \ldots, T-1 \\
& W_{t+1} = W_t^{\rightarrow} - C(W_t^{\rightarrow}; \boldsymbol{\pi}_t^{\rightarrow}, \boldsymbol{\pi}_{t+1}) & & \forall t = 0, \ldots, T-1,
\end{aligned} \tag{1}$$

where $\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_{T-1} \in \mathbb{R}^n$ are the decision variables representing the allocation at the beginning of each period, $\mathbf{1} \in \mathbb{R}^n$ is the vector with all ones, W_t is the wealth at the beginning of period t, $W_t^{\rightarrow}$ is the wealth at the end of

period t, $\boldsymbol{r}_t \in \mathbb{R}^n$ is the vector of returns in period t, $\boldsymbol{\pi}_t^{\rightarrow} \in \mathbb{R}^n$ is the allocation vector at the end of period t, $\odot$ is the element-wise multiplication operator, and $C(W_t^{\rightarrow}; \boldsymbol{\pi}_t^{\rightarrow}, \boldsymbol{\pi}_{t+1})$ is the dollar value of transaction and market impact costs when the allocation is rebalanced to π_{t+1} from $\pi_t^{\rightarrow}$ with current wealth being $W_t^{\rightarrow}$. Initial wealth W_0 is assumed to be given. The distribution of asset returns r_t and the form of transaction costs $C(\cdot;\cdot,\cdot)$ depends on specific circumstances. The arguments of utility function $Z_1, Z_2, \ldots$ can be, but are not limited to, expected terminal wealth $\mathbb{E}[W_T]$, volatility of terminal wealth $std[W_T]$, worst case measures such as conditional value-at-risk, and so on.

The rest of this section provides a specific description of our regime-shifting market and the problem that the investor faces in a multiperiod investment model.

2.2. The Market

We consider a market where there are multiple regimes. In each regime, the prices of risky assets follow correlated geometric Brownian motion, with parameters depending on the regime. For tractability, regime switching occurs at the end of each period. Following recent literature on portfolio optimization involving multiple regimes (Bauerle and Rieder 2004, Bauerle and Rieder 2007, Çanakoğlu and Özekici 2009, Nystrup et al. 2018, Nystrup et al. 2019), we further suppose that the regime-switching behavior satisfies the Markov property; that is, given the regime of the current period, the regime probability of the next period is independent of the regimes of all previous periods and is expressed by means of a transition matrix.

Within each period, the vector of prices of risky assets, $\boldsymbol{X}_t = [X_{t,1}, \ldots, X_{t,n}]'$, follows $dX_{t,i} = \kappa_{S_t,i} X_{t,i} dt + \eta_{S_t,i} X_{t,i} dW_{t,i}$ for i = 1, …, n, where $S_t \in \{1, 2, .., N\}$ is the regime at time t, and $W = \{W_t = [\boldsymbol{W}_{t,1}, \ldots, W_{t,n}]', t \geq 0\}$ is a standard correlated n-dimensional Brownian motion. The returns of the risky assets, therefore, follow multivariate normal distributions with regime-dependent parameters. Mathematically, the vector of returns of risky assets in period i, $r_t = [r_{t,1}, \ldots, r_{t,n}]'$, follows the dynamic

$$r_t \sim \mathbb{N}(\boldsymbol{\mu}_{S_t}, \boldsymbol{\Sigma}_{S_t}), \qquad S_t \in \{1, 2, .., N\}, \tag{2}$$

where N is the number of possible regimes, and $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ for $k \in \{1, 2, \ldots, N\}$ are the parameters of the normal distribution under regime k.

In addition, there is a risk-free asset in the market, whose price grows at rate r_f. The risk-free rate may depend on the underlying regimes as well. Mathematically, the price of the risk-free asset, Y_t, follows $dY_t = r_f Y_t dt$.

We assume linear transaction and market impact cost in this market. In particular, it is taken to be a constant proportion of the dollar amount of any buy or sell of the risky asset. Throughout the paper, we use W to denote investor wealth and $\boldsymbol{\pi} = [\pi^1, \pi^2, \ldots, \pi^n]'$ to denote the proportion of total wealth that is invested in the risky assets. For an investor with total wealth W and position $\boldsymbol{\pi}^-$, if he makes a transaction to adjust his position to $\boldsymbol{\pi}^+$, then the transaction cost to pay would be $cW||\boldsymbol{\pi}^+ - \boldsymbol{\pi}^-||_1$, where c is a fixed constant in (0, 1).

2.3. The Investor

The investors interact with the market in order to collect returns and therefore meet their investment goals. We make the assumption that the investors can only trade at the end of each period, right after the returns of this period are realized. Although the regimes are not directly observable, investors are capable of estimating the probability of future regimes by learning the historical data. Therefore, the investors make rebalancing decisions based on the current performance information and their beliefs about the future.

Our goal is to maximize the expected utility of terminal wealth, $\mathbb{E}[U(W_T)]$, at the end of a given horizon T:

$$\underset{\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots, \boldsymbol{\pi}_{T-1}}{\text{Maximize}} \ \mathbf{E}[U(W_T)]. \tag{3}$$

Under the framework of Pratt (1964) and Arrow (1971), a rational investor chooses actions to maximize a nondecreasing utility function. In this paper, we choose U to be the constant relative risk-aversion (CRRA) utility:

$$U(W) = \begin{cases} \dfrac{W^\gamma}{\gamma}, & \gamma \neq 0 \\ \log(W), & \gamma = 0. \end{cases}$$

Here we will focus on the scenario when $\gamma \leq 0$, in which case the utility is strictly concave and the investor is decreasingly risk-averse.

Leverage limits are imposed to mimic real-world situations. In particular, we restrict all entries in $\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots, \boldsymbol{\pi}_{T-1}$ to be in the interval $[\pi_l, \pi_u]$, where π_l and π_u are the lower and upper bounds, respectively, of the position an investor may take on each of the risky assets.

The task is to choose the positions at the beginning of each period $\pi_0, \pi_1, \pi_2, \ldots, \pi_{T-1}$ to achieve the goal in Equation (3). There are two main difficulties in this market environment: first, even without transaction costs, it is hard, if not impossible, to attain a tractable analytical optimal solution under unobservable regime switching with constraints on leverage limit and other assumptions; and, second, even when an optimal strategy under no transaction costs presents, it will no longer be optimal if transaction costs are nonnegligible. In the next section, we develop a procedure to mitigate these two issues and find a promising trading strategy under our market constraints.

3. Methodology

We present our two-step procedure of tackling the multiperiod portfolio allocation problem in this section. In the first stage, we adopt numeric dynamic programming to obtain an optimal allocation strategy under no transaction costs. Then, we feed the solution strategy as the starting point to a neural network and find an investment plan for the full problem described in Section 2.

3.1. Dynamic Programming

Bellman (1954) gave birth to dynamic programming, offering a profound influence on multistage decision problems. Dynamic programming takes advantage of the backward recursive property to solve the optimal portfolio under regime switching and zero transaction costs.

3.1.1. State Space. In a multiperiod portfolio optimization problem, the state space includes all the information that an agent uses to make a trading decision. It is well known that for an investor maximizing CRRA utility, she will maintain a constant proportion of her wealth in each asset. A CRRA utility function allows us to optimize the asset allocation as a proportion of the total wealth without knowing the wealth itself, enabling us to exclude wealth from the state space. In our context given no transaction costs, to fully describe the state of environment, the minimal sufficient state includes the probability distribution of the current regime and the time until horizon. In mathematical notation, the state is a pair $(\boldsymbol{p}, \tau)$, where $\boldsymbol{p} = [p_1, \ldots, p_N]'$ is the vector of probability of the current period being in each regime, and τ is the current time period.

To take advantage of dynamic programming, we discretize the state space to make it finite. With the assumption that a transaction can happen only at the end of each period, the horizon is by nature discrete and finite. We restrict the probability vector $\boldsymbol{p} = [p_1, \ldots, p_N]'$ to take values in a discretized set $\mathbf{P} = \{[p_1, \ldots, p_N]' \,|\, p_i \in \{0, \delta, 2\delta, \ldots, M\delta = 1\}\ \forall i \in \{1, \ldots, N\}, \Sigma_{i=1}^{N} p_i = 1\}$, where δ is a small unit change. As δ is small, the set is representative of all probability distributions that p may take.

Since the process is Markov, the probability provides complete information for the agent to make a decision. We thus conclude that the set

$$\mathbf{S} = \{(\boldsymbol{p}, \tau) \,|\, \boldsymbol{p} = [p_1, \ldots, p_N]' \in \mathbf{P}, \tau \in \{0, 1, 2, \ldots, T-1\}\},$$

with each term explained in the previous paragraphs, includes all the essential states in our context.

3.1.2. Action Space. The action space is a set, $\mathbf{A}$, of all actions that the agent may take in accordance with the nonanticipatory conditions. In the multiperiod portfolio optimization problem, the agent may choose the amount of wealth to invest in the risky asset. The action space is thus $\{\boldsymbol{\pi} = [\pi^1, \ldots, \pi^n]' \,|\, \pi^j \in [\pi_l, \pi_u]\ \forall j \in \{1, \ldots, n\}\}$, where π_l and π_u are limits on position, as specified earlier.

In general, investors may face richer choices of actions, including picking the execution time. In this paper, since the execution time is limited to the end of each period, the agent is only left with the freedom of choosing positions. Also, we ignore additional constraints, such as limits on groups of assets.

3.1.3. Value Function. A value function $V : \mathbf{S} \to \mathbb{R}$ maps states to real numbers. It evaluates how well an agent may end up, given that the agent is currently in some state $s \in \mathbf{S}$. In our setting, we define the value function to be the expected terminal utility following the optimal trading strategy from the current state to the horizon (cost-to-go) and assuming that the current wealth is \$1:

$$V(s_i) = \underset{\{\boldsymbol{\pi}\}}{\text{Maximize}}\, \mathbf{E}[U(W_T) \,|\, s_i, W_i = 1]\ \forall s_i \in \mathbf{S}.$$

The value function should satisfy the Bellman equation:

$$V(s_i) = \underset{\boldsymbol{\pi}_i}{\text{Maximize}}\, \Sigma_{s_{i+1}, W_{i+1}} p(s_{i+1}, W_{i+1} \,|\, s_i, \boldsymbol{\pi}_i) \mathbf{E}[W_{i+1}^{\gamma} V(s_{i+1})], \tag{4}$$

where $p(s_{i+1}, W_{i+1} | s_i, \boldsymbol{\pi}_i)$ is the probability of transiting to state s_{i+1} with wealth W_{i+1} at the next stage, given that the agent takes action π_i under state s_i.

The dynamic programming algorithm of searching for the optimal portfolio under regime switching and zero transaction costs is presented as pseudo-code in Algorithm 1.

Algorithm 1 (Optimal Asset Allocation Under Regime Switching and Zero Transaction Costs)

Result: Optimal position of risky assets $\boldsymbol{\pi}^*(\boldsymbol{p}, t)$

$t = T - 1$;

for each $\boldsymbol{p} = [p_1, \ldots, p_N]' \in \mathbf{P}$ **do**

  $V(\boldsymbol{p}, T) = \frac{1^\gamma}{\gamma}$

end

Function `updateBelief`$(\boldsymbol{r}, \boldsymbol{p})$**:**

  $p_k^{new} = \frac{pdf(r; \mu_k, \Sigma_k) * p_k}{\sum_{l=1}^{N} pdf(r; \mu_l, \Sigma_l) * p_l}$;

  return $\boldsymbol{p}^{new} = [p_1^{new}, \ldots, p_N^{new}]'$;

Function `newWealth`$(\boldsymbol{r}, \boldsymbol{\pi})$**:**

  $W^{new} = \Sigma_{i=0}^{n} \boldsymbol{\pi}^i * (1 + r^i)$;

  return W^{new};

while $t \geq 0$ **do**

  for $p = [p_1, \ldots, p_N]' \in \mathbf{P}$ **do**

    Simulate M return scenarios $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_M \in \mathbb{R}^{n+1}$ in period t based on the probability of regimes $\boldsymbol{p}$;

    $\boldsymbol{\pi}^*(\boldsymbol{p}, t) = \text{argmax}_\pi (\frac{1}{M} \Sigma_{s=1}^{M} \texttt{newWealth}(\boldsymbol{r}_s, \boldsymbol{\pi})^\gamma V(\texttt{updateBelief}(\boldsymbol{r}_s, \boldsymbol{p}), t+1))$;

    $\boldsymbol{\pi}^*(\boldsymbol{p}, t) = \max(\pi_l, \min(\pi_u, \boldsymbol{\pi}^*(\boldsymbol{p}, t)))$

    $V(\boldsymbol{p}, t) = (\frac{1}{M} \Sigma_{s=1}^{M} \texttt{newWealth}(\boldsymbol{r}_s, \boldsymbol{\pi}^*)^\gamma V(\texttt{updateBelief}(\boldsymbol{r}_s, \boldsymbol{p}), t+1))$

  end

  $t = t - 1$

end

The time complexity of a dynamic program depends on the size of the spaces. As either state space or action space grows, the number of required simulations grows accordingly, to ensure all possible actions being explored at all states. This phenomenon is called *curse of dimensionality*. As discussed before, we note that when transaction costs are not considered, the state space is rich enough by including the probability distribution of the current regime and horizon. On the other hand, if we address the problem of adding transaction costs directly, then the state space must also include current position π to provide complete information to the agent. Therefore, we choose not to consider transaction cost in the first step, where we exploit dynamic programming. Instead, we take advantage of neural networks to find the strategy under transaction costs in the second algorithmic stage.

One may choose to directly apply analytical solution for the problem with no transaction costs (Mulvey et al. (2020)). For example, Çanakoğlu and Özekici (2012) provide analytical solution to a problem similar to the one in this paper. Unlike our problem, where the observable time series comprises the asset returns (a continuous space), they require the observable state to be discrete. Yet, an investor can take advantage of Çanakoğlu and Özekici's results if she carefully discretizes the observable space. Compared to their work, our numerical method also provides a number of desirable features: it can (i) easily incorporate constraints on shorting and leverage, (ii) allow risk-free rates to have state-dependent distributions, and (iii) give fast solutions for practical problems (Sections 4 and 5). Our combined numerical method works well on reasonable-sized problems with few additional constraints.

3.2. Neural Networks

Now that we solve the multiperiod portfolio optimization under the assumption of zero transaction costs, we take a step further and seek the optimal trading strategies under proportional transaction costs. When the transaction costs are nonnegligible, it is no longer appropriate to simply apply the strategy under no transaction costs. On one hand, the investor is likely to gain higher expected terminal utility if she stays close to the optimal

position under no transaction costs; on the other hand, rendering frequent transactions can be costly, and thus she also wants to reduce the total dollar amount of transactions.

3.2.1. No-Trade Zone. It is widely known that the optimal trading strategy under geometric Brownian motion (GBM) assets and proportional transaction cost is to form a no-trade zone (Davis and Norman 1990). When the current position is inside the no-trade zone, no action is required. Otherwise, an agent should rebalance her portfolio to the closest point (either the lower or the upper boundary) in the no-trade zone. The strategy of forming a no-trade zone has also been adopted in more complicated settings, where investors are advised to rebalance only when their positions are far from the theoretical optimal position under no transaction costs. The strategy of the no-trade zone has good explanatory power and is shown to be tractable in practical settings.

We assume that an agent in the regime-switching world adopts this strategy. For example, let us assume that a no-trade zone is determined to have shape as shown in Figure 1. To demonstrate how a no-trade-zone strategy works, we walk through an example as follows. Assume that the probability of the current regime being normal is 0.2, and assume the three different current positions:

- Case (a): Our current position of risky asset is point a. Since this position is higher than the upper bound of the no-trade zone, we will sell some of the risky asset and rebalance to position a' on the upper boundary.
- Case (b): Our current position of risky asset is point b. Since point b is inside the no-trade zone, no action is needed.
- Case (c): Our current position of risky asset is point c. Since this position is lower than the lower bound of the no-trade zone, we will purchase some risky asset and rebalance to position c' on the lower boundary.

3.2.2. Recurrent Neural Networks. A recurrent neural network (RNN) is an artificial neural network whose nodes form a directed connection and is often applied to problems with temporal structure (Connor and Martin 1994). In a vanilla recurrent neural network, at each time step, we feed the output from the previous time step (a hidden state) along with a time-dependent input into the neural network. Figure 2 provides a visualization of the computational graph, unfolding an RNN in temporal order. Formally, given a sequence $\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$, an RNN computes the hidden states $\boldsymbol{h}_t$ and outputs $\boldsymbol{o}_t$ by

$$\boldsymbol{h}_t = \theta(\boldsymbol{W}_{hh}\boldsymbol{h}_{t-1}, \boldsymbol{W}_{xh}\boldsymbol{x}_t),$$
$$\boldsymbol{o}_t = \phi(\boldsymbol{W}_{ho}\boldsymbol{h}_t),$$

where $\boldsymbol{W}_{hh}$, $\boldsymbol{W}_{xh}$, and $\boldsymbol{W}_{ho}$ are the weights connecting the respective nodes, and θ and ϕ are problem-based activation functions. Extensions of RNN include long short-term memory (Chung et al. 2014) and bidirectional RNN (Schuster and Paliwal 1997), for which we will not go into details in this paper.

The target is to learn the weights connecting the input nodes, hidden nodes, and output nodes, which represent the relative strength of the connection between the nodes. In general, the weights in an RNN are shared across time. Sharing weights effectively reduces the number of parameters to train and thus avoids overfitting. Given the temporal structure of our problem, we choose to use RNN for learning the boundaries of the no-trade zone.

Figure 1. (Color online) Example of No-Trade-Zone Strategy

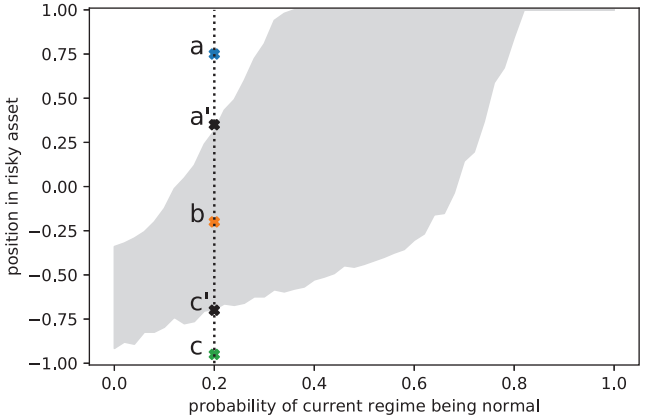
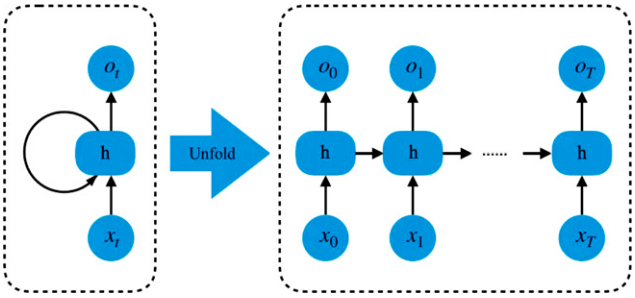


Figure 2. (Color online) Typical Computational Graph of Recurrent Neural Networks



Notes. At each time step, the neural network h takes the input x_t and the previous state, and it may produce output o_t. The graph to the right unfolds the structure on the left.

3.2.3. Searching for the Optimal No-Trade Zone. Recent years have witnessed wide applications of neural networks, including in the fields of partial differential equations and mathematical finance (Han and E 2016). Neural networks are a powerful tool, especially when an advanced starting point is provided. In our work, we develop a solution by using dynamic programming as a starting point and search for a no-trade zone around this allocation by using neural networks. We parameterize the upper and lower boundaries of the no-trade zone as

$$\boldsymbol{u}(\boldsymbol{p},\tau) = \boldsymbol{\pi}(\boldsymbol{p},\tau) + f_u(\boldsymbol{p},\tau), \tag{5}$$

$$\boldsymbol{l}(\boldsymbol{p},\tau) = \boldsymbol{\pi}(\boldsymbol{p},\tau) - f_l(\boldsymbol{p},\tau), \tag{6}$$

where $\boldsymbol{\pi}(\boldsymbol{p},\tau)$ is the optimal position under no transaction cost, and $\boldsymbol{u}(\boldsymbol{p},\tau)$ and $\boldsymbol{l}(\boldsymbol{p},\tau)$ are lower and upper boundaries of the no-trade zone, respectively. The neural networks approximate the nonnegative functions of interest, f_u and f_l, that is, the "width" of the no-trade zone around $\boldsymbol{\pi}(\boldsymbol{p},\tau)$ for each pair $(\boldsymbol{p},\tau)$.

To increase the stability of learning and to avoid overfitting, we make the reasonable assumption that the width of the no-trade zone does not vary much with time, which allows all time steps to share the same parameterization based solely on the probability estimation of the current regime; that is, the boundaries of the no-trade zone are calibrated as

$$\boldsymbol{u}(\boldsymbol{p},\tau) = \boldsymbol{\pi}(\boldsymbol{p},\tau) + f_u(p), \tag{7}$$

$$\boldsymbol{l}(\boldsymbol{p},\tau) = \boldsymbol{\pi}(\boldsymbol{p},\tau) - f_l(p). \tag{8}$$

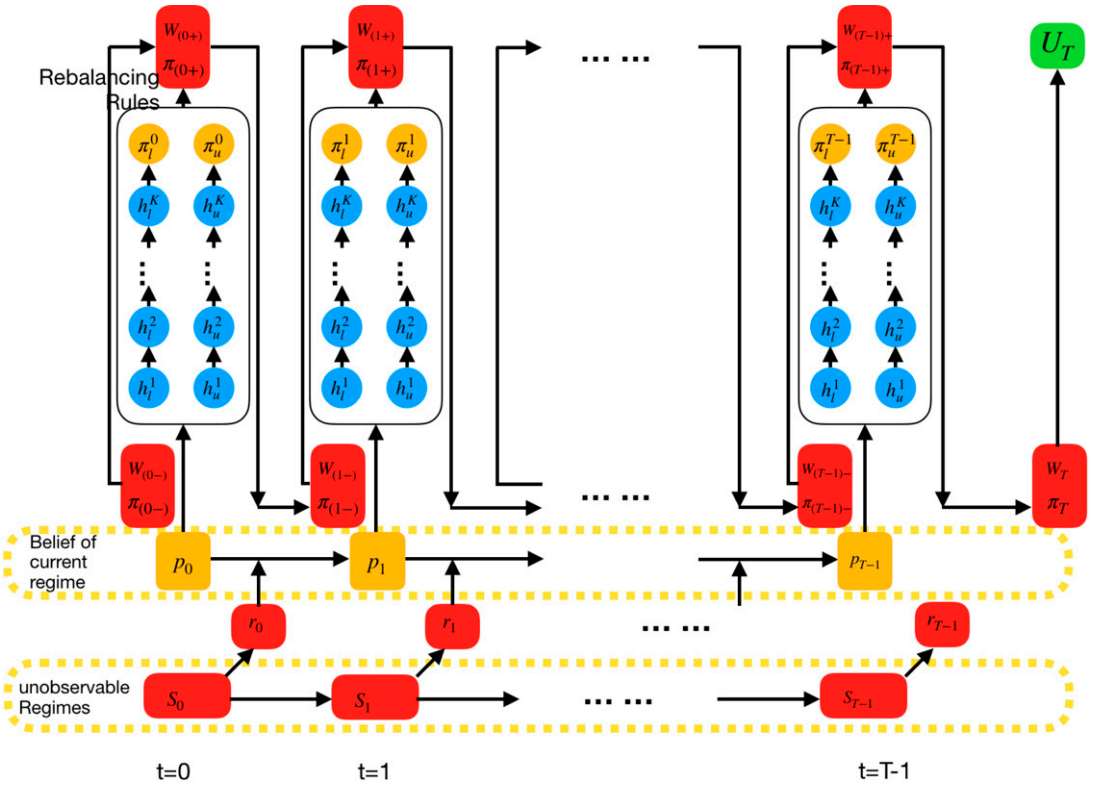
Next, we build a network based on the structure of Han and E (2016) and Mulvey et al. (2020). The computational graph for the neural network appears in Figure 3. The relationships inside the computational graph are follows:

- S_t for $t=0,\ldots,T-1$ represents the underlying regime of the market in period t. The regime evolves according to a Markov process and is not directly observable to the investors.
- $\boldsymbol{r}_t \in \mathbb{R}^n$ for $t=0,\ldots,T-1$ represents the realized return of the risky assets in period t. It is normally distributed with mean μ_{S_t} and covariance Σ_{S_t}.
- $\boldsymbol{p}_t \in \mathbb{R}^N$ for $t=0,\ldots,T-1$ is the investor's belief of the current regime. The investor updates her belief of the underlying regime based on her previous belief and the new realized return using Bayesian rule.
- $\boldsymbol{h}_l^k$ and $\boldsymbol{h}_u^k$ for $k = 1, 2, 3$ represent the hidden layers. At each period, we build a three-layer feedforward neural network for each of the lower and upper boundaries, where the hidden layers are labeled as $\boldsymbol{h}_l^k$ and $\boldsymbol{h}_u^k$ for $k = 1, 2, 3$, respectively. The objective is to optimize over the weights in these neural networks. To increase the stability of the results, we use the same set of weights across all periods. For $t=0,\ldots,T-1$, $\boldsymbol{\pi}_l^t$ and $\boldsymbol{\pi}_u^t$ are the output lower and upper boundaries, respectivel, of the no-trade zone learned by the neural network.
- $W_{(t-)}$ and $\boldsymbol{\pi}_{(t-)}$ for $t=0,\ldots,T-1$ depict the wealth and asset allocation at time t before rebalancing, respectively. At time t, the investors can rebalance their portfolio. The new allocation $\boldsymbol{\pi}_{(t+)}$ is chosen based on Section 3.2.1, according to the old allocation $\boldsymbol{\pi}_{(t-)}$ and the no-trade zone calibrated by the neural network. The wealth after rebalancing is calculated as $W_{(t+)} = W_{(t-)} - cW_{(t-)}||\boldsymbol{\pi}_{(t+)} - \boldsymbol{\pi}_{(t-)}||_1$.

We use Monte Carlo simulations to generate regimes and prices of the risky assets based on estimated parameters in order to train the neural networks. At the end of period T, the loss function equals the negative of expected terminal utility,

$$loss = -\mathbf{E}[U(W_T)],$$

Figure 3. Computational Graph for the Neural Network



and is minimized using gradient backpropagation with TensorFlow (Abadi et al. 2015). We apply cosine decay with restarts to the learning rate, using method tensorflow.cosine_decay_restarts (Loshchilov and Hutter 2016). Such learning rate pattern helps us gain faster convergence and stabler results.

4. Comparison of Performance

To our knowledge, there is no study evaluating the performance of a neural network on learning the optimal trading strategy involving regime switching. Fortunately, assuming the Bayesian rule is the optimal way of updating belief of the current regime, we are able to approximate the optimal no-trade zone with dynamic programming when there is only one risky asset. In this section, we compare the results of (i) directly using dynamic programming to find the no-trade zone and (ii) using dynamic programming to find the optimal portfolio under zero transaction costs, and then feeding it into a neural network as a starting point ("combined method" in short). We find that, in the case where there is one risky asset, method (ii) provides promising results. Notice that every result shown in this section makes the assumption that market dynamics is relatively stable over time, in order to compare the performance.

4.1. Approximate Optimal No-Trade Zone with Stand-Alone Dynamic Programming

As a point of comparison, we present an algorithm that directly uses dynamic programming to find the optimal no-trade zone under proportional transaction costs. Similar to the discussion in Section 3.1, the state space includes the probability distribution of current regime and the time until horizon. In addition, when transaction costs are considered, the state space should also include the vector of current asset allocation $\boldsymbol{\pi} = [\pi^1, \ldots, \pi^n]'$, because the current allocation will now affect the agent's rebalancing decision.

The action space becomes richer as well. Instead of just looking for the optimal portfolio $\boldsymbol{\pi}^* = [\pi^{1*}, \ldots, \pi^{n*}]'$, the dimension doubles as we aim to find the boundaries of the no-trade zone. The action space now consists of all admissible pairs (π_{lb}, π_{ub}), where $\boldsymbol{\pi}_{lb}$ and $\boldsymbol{\pi}_{ub}$ are the lower bound and the upper bound, respectively, of the no-trade zone.

Again, we take the dynamic programming algorithm and solve the problem backward (Algorithm 1). We define two value functions, namely, before rebalancing and after rebalancing, for all possible states. First, we

calibrate the no-trade zone at time $T-1$, and we then use the value function method to find the no-trade zone backward, until we solve the problem for the entire horizon.

Note that if one directly uses dynamic programming to approach the no-trade zone boundaries, then the dimension of state space grows with the number of risky assets in the market. The algorithm suffers from the curse of dimensionality, and execution time grows exponentially with the number of assets. Fortunately, the running time is reasonable for the case of one risky asset, enabling us to provide a benchmark on the performance comparison.

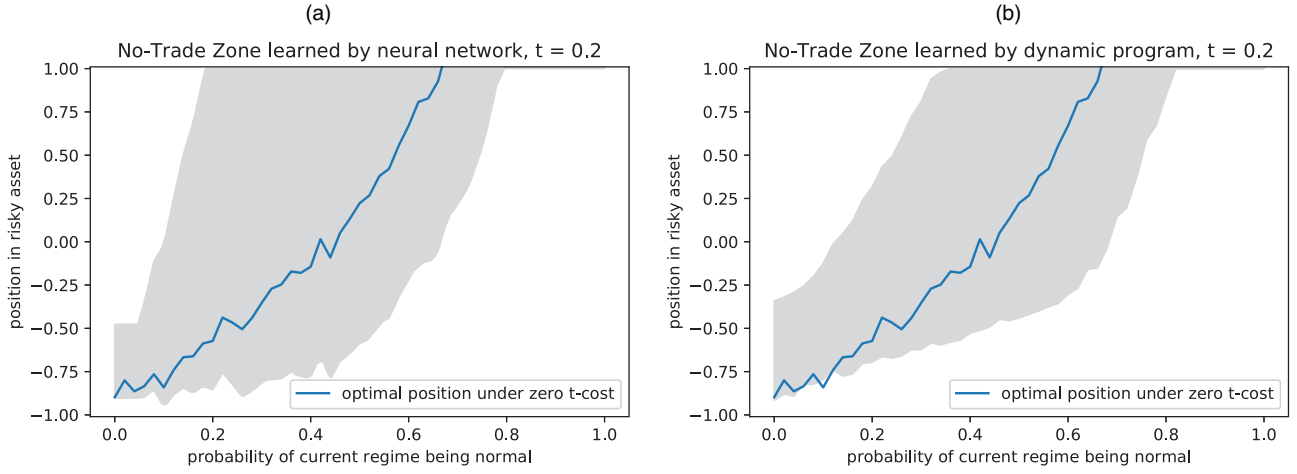
4.2. Comparison of Results

We run both methods on a 50-period model with two possible underlying regimes. There are one risky asset and one risk-free asset in the market. The horizon is one year (approximately 50 weeks), and the investor is allowed to rebalance at the end of each week. We calibrate the parameters based on weekly returns of the S&P 500 Index from January 1990 to November 2019. Parameters are specified as follows: the annualized return on the risk-free asset is $r = 1.54\%$, regardless of underlying regimes; under regime 1, the annualized return of the risky asset is normally distributed with mean $\mu_1 = 0.00312(50)$ and variance $\sigma_1^2 = 0.00022(50)$; under regime 2, the annualized return of the risky asset is normally distributed with mean $\mu_2 = -0.00175(50)$ and variance $\sigma_2^2 = 0.00116(50)$; and the transition matrix of regimes is $\begin{bmatrix} 0.981 & 0.019 \\ 0.047 & 0.953 \end{bmatrix}$. The allocation in the risky asset is limited to the range $[-100\%, 100\%]$. The transaction cost is chosen to be 0.5%.

For the stand-alone dynamic program, five iterations are carried out with 20,000 paths for each iteration. The no-trade zone is approximated as the average of the results of all iterations. The no-trade zone found by the dynamic program is close to theoretical optimal, provided that enough simulations have been run.

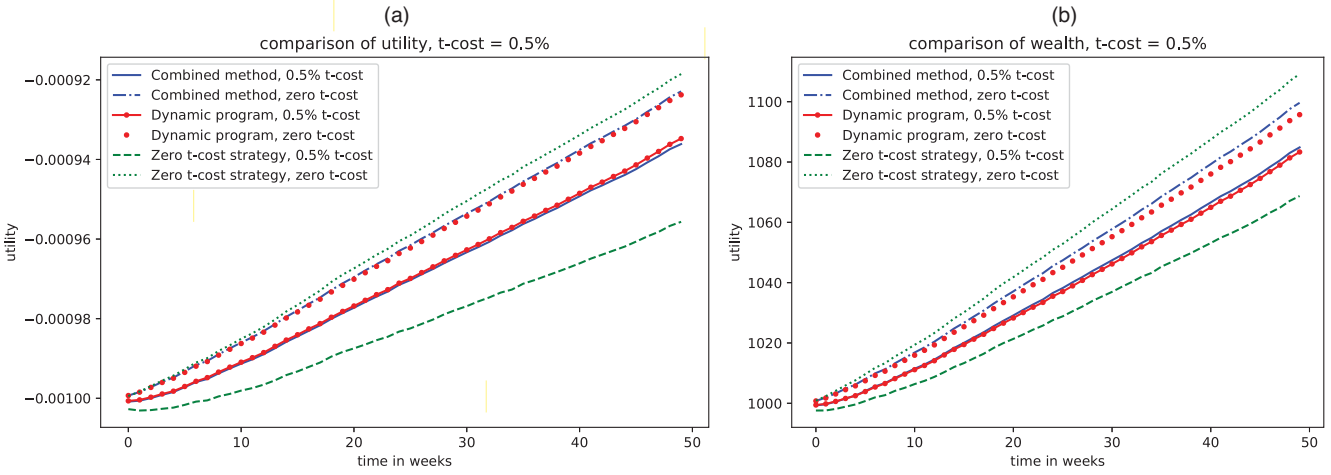
For the combined method, after learning the optimal strategy under zero transaction costs with dynamic programming, we build two neural networks for the upper and lower bounds, respectively. Each of the neural networks has three hidden layers with 20, 40, and 80 neurons, respectively. The two networks take in the probability distribution of the current regime, and output $f_u(p)$ and $f_l(p)$, enabling us to parameterize the upper and lower bounds of the no-trade zone with Equations (7) and (8). We train the neural networks for 500 steps with the Adam optimizer at learning rate 10^{-4}. To oversee training, we check the utility every 50 steps and stop training when it converges. The no-trade zones learned by neural networking and dynamic programming are shown in panels (a) and (b) of Figure 4, respectively. Even though the no-trade zones are not identical, it turns out that, based on 10,000 sample paths simulated with the estimated parameters, the strategy learned by neural networking provides similar utility and wealth growth to the one learned by dynamic programming over a 50-week period. The comparisons of utility path and of wealth path are presented in panels (a) and (b) of Figure 5. Both methods significantly improve the naive strategy, where the investor rebalances to the optimal solution under zero transaction costs at the end of all periods. Figure 5 shows that the utility achieved by the combined method is essentially the

Figure 4. (Color online) No-Trade Zones Learned by Neural Networking and Dynamic Programming at Time = 0.2 Years



Notes. (a) No-trade zone learned by neural network. (b) No-trade zone learned by dynamic programming.

Figure 5. Comparison of Methods on a Planning Period of 50 Weeks



Notes. Rebalancing is allowed weekly. The dotted lines are the average utility and wealth, respectively, yielded with the same strategies in a market with zero transaction costs. The dotted lines are unattainable in the presence of transaction costs; yet, the differences between the corresponding dotted and solid lines indicate the loss of ignorance of transaction costs. (a) Utility path. (b) Wealth path.

same as that achieved by the full dynamic program, which is close to theoretical optimal. This justifies the effectiveness of employing neural networks and provides the cornerstone for the following experiments.

Figure 6 shows detailed information for one random path. We include the sample returns with underlying regime, the investor's perception of underlying regime, the utility and wealth paths of each strategy, as well as the amount of risky asset to be held with each strategy.

The strategy learned by the combined method is generating nearly as much growth in both utility and wealth as the strategy learned by the full dynamic program. Both successfully avoid paying too much transaction fees (Table 1), as compared with the strategy where one rebalances every period as if there are no transaction costs. The result suggests that the combined method involving neural networking is promising for discovering the appropriate strategy in our setting with proper learning rates. We stress that adopting neural networks is a time-efficient method. In this particular example, where there is one risky asset, the dynamic program takes approximately 16.5 minutes to find the no-trade zone, whereas the combined method takes about 3.5 minutes. The time difference may not be significant in the case of one risky asset; however, as the number of risky assets grows, pure dynamic programming becomes impractical as the state space explodes exponentially, while we expect a polynomial, if not linear, running time using the neural networks (Section 5, Table 4).

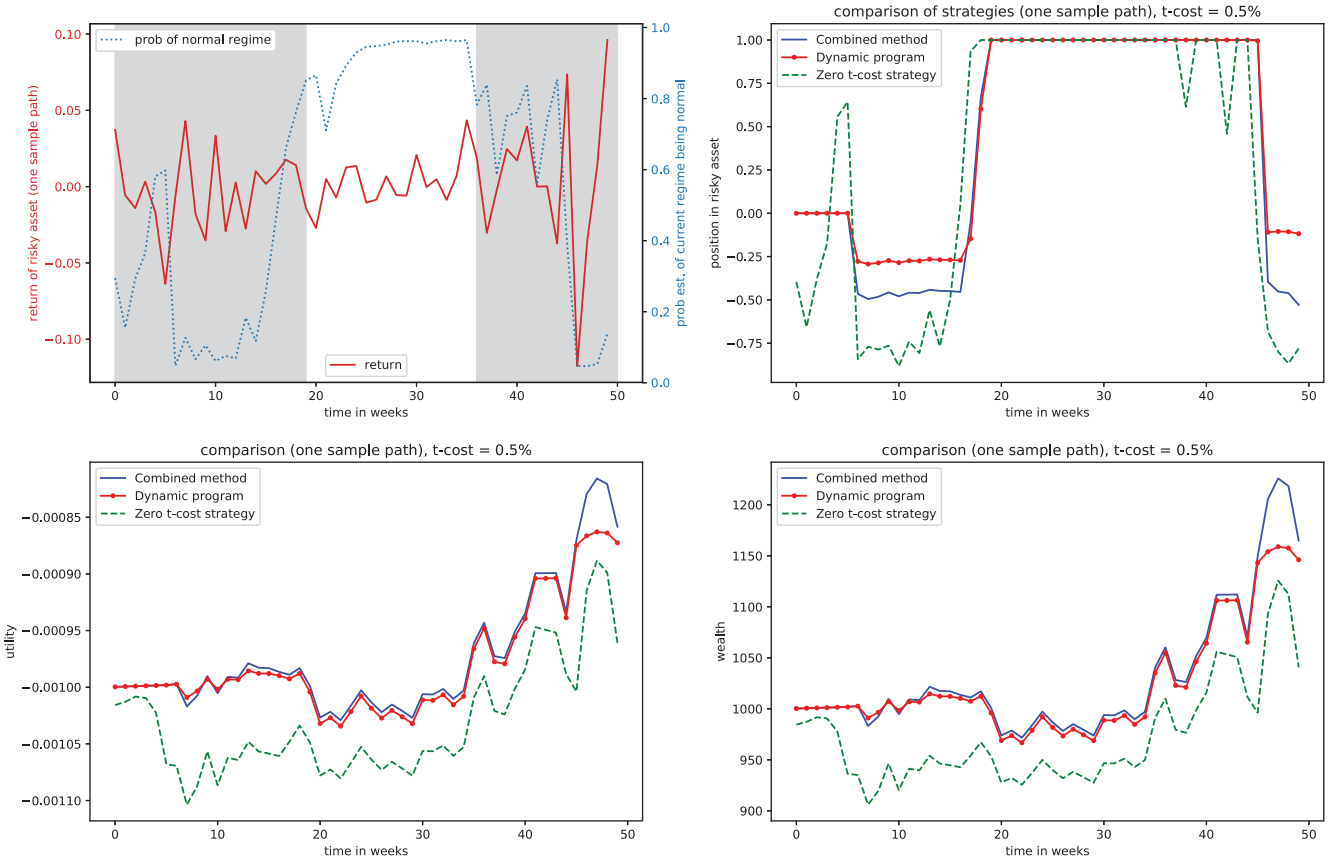
We also test the combined method involving neural networking on the monthly returns of the S&P 500 Index with transaction costs of 1.5%. The parameters are estimated as follows: under regime 1, the annualized return of the risky asset is normally distributed with mean $\mu_1 = 0.01107(12)$ and variance $\sigma_1^2 = 0.00064(12)$; under regime 2, the annualized return of the risky asset is normally distributed with mean $\mu_2 = 0.00241(12)$ and variance $\sigma_2^2 = 0.00297(12)$; and the transition matrix of regimes is $\begin{bmatrix} 0.965 & 0.035 \\ 0.042 & 0.958 \end{bmatrix}$. The results appear in Figure 7 and Table 2. Similar conclusions are drawn as earlier.

5. Empirical Example with Market Data

In this section, we provide four numerical experiments with real-world data. Herein, the allocation in each risky asset must fall in the range [−100%, 100%], so that there is a reasonable leverage limit on the portfolio. The first example considers one risky asset and one risk-free asset. The parameters are estimated based on S&P500 weekly data, and empirical results are provided with nonanticipativity constraint. Since the market is not perfectly stationary, we update the parameter estimation on a rolling basis, and we test the strategy on realized market returns. The objective of this example is to illustrate the benefit of our method in a real market, even when it is not always stationary. Results indicate the potential effectiveness of the combined method in real-world applications, where investors have no information on future returns other than estimating based on information up to the decision date.

Having justified the efficacy in the real market, we focus the rest of the section on addressing the computational advantage of the combined algorithm. In these examples, we estimate the parameters based on market returns

Figure 6. (Color online) Random Path of Returns of the Risky Asset



Note. The shaded time periods in the top left graph are under the crash regime, whereas the unshaded ones are under the normal regime.

and keep them fixed for simulations in order to compare the performance on average (as opposed to on one realized path). The second and third experiments aim to show the time efficiency of our combined method. Multiple risky assets are included in these trials for running time analysis. In the second experiment, we pick four large equities and show the performance of our method on a multiasset case. In particular, we present the no-trade zone for a multiasset setting. In the third example, we extend the method to 11 equities to show its ability of dealing with a large number of risky assets. An analysis of how risk-aversion coefficient affects the average terminal wealth is included. Lastly, we present an experiment of three regimes. In all of the four examples, the combined method is shown to have significant improvement on the benchmark strategy that rebalances every period. A table of running times of the neural network appears at the end of the section.

5.1. Empirical Results with One Risky Asset

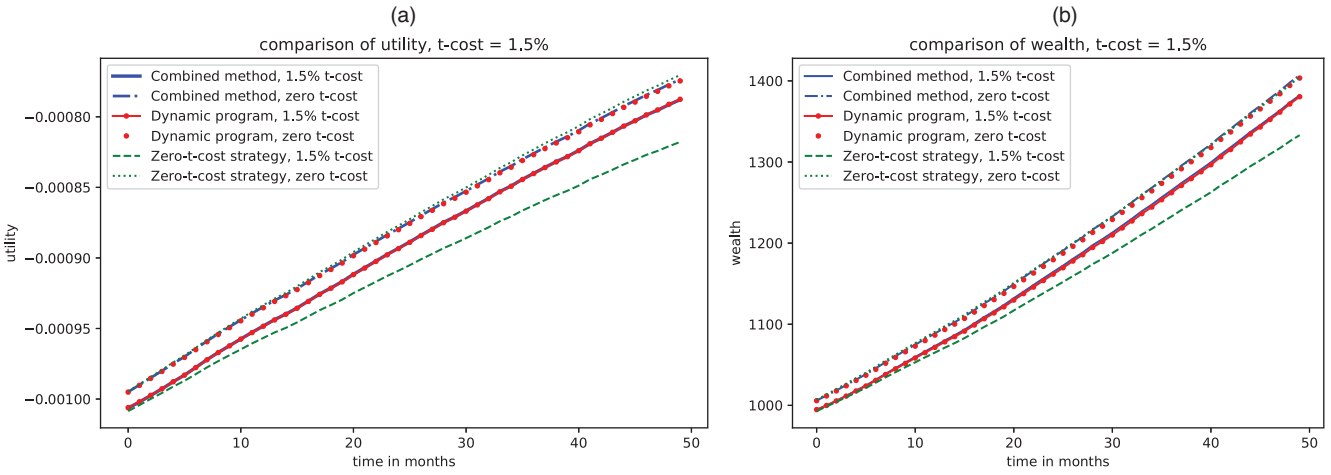
We define the S&P 500 Index as the risky asset in the market. To satisfy the nonanticipativity constraint, we use the hidden Markov model to fit the model parameters based on the weekly return of S&P500 from January 1990 to December 2015. A model is trained based on these parameters, and we apply the results to the 53-week period from January 2016 to December 2016. This test result is fully out-of-sample.

For interpretability, we designate the regimes "normal" and "crash," respectively. Since there are approximately 53 weeks in 2016, we train a 53-period model. In the first week, the annualized returns of the risky asset under the normal regime is normally distributed with mean $\mu_1 = 0.00279(53)$ and variance $\sigma_1^2 = 0.00026(53)$. The

Table 1. Average Total Transaction Costs Paid over a 50-Week Period with Different Strategies (One Risky Asset, Weekly Rebalance with 0.5% Transaction Cost, and Two Regimes)

Combined method	Dynamic program	Optimal under zero-*t*-cost
13.94	11.89	38.28

Figure 7. Comparison of Methods on a Planning Period of 50 Months (Monthly Rebalancing)



Notes. (a) Comparison of methods: utility path. (b) Comparison of methods: wealth path.

annualized returns of the risky asset under the crash regime is normally distributed with mean $\mu_2 = -0.00209(53)$ and variance $\sigma_2^2 = 0.00136(53)$. The transition matrix between the two regimes is estimated to be $\begin{bmatrix} 0.981 & 0.019 \\ 0.060 & 0.938 \end{bmatrix}$. The transaction cost equals 0.5%, and the coefficient of risk aversion is again $\gamma = -1$. The parameters of the hidden Markov model are updated weekly on a rolling basis, based on all information prior to the current week. The investor therefore runs a model every week to determine the allocation at the end of the weekly period. The model returns the allocation decision for every week until the horizon, yet only the first allocation decision is actually carried out. The advantage of running a full model is that one takes into account future transaction costs and possible regime switching. It takes up to 3.5 minutes and, on average, 1.7 minutes per week (on the Princeton computer cluster Adroit, with 32 2.60-GHz Intel Skylake CPU cores and 384 GB RAM), to decide the investment strategy, which is practical even for individual investors. We compare the strategy found by the combined method to the benchmark strategies: buy and hold 100% risky assets as well as a 60/40 fixed mix strategy. The starting wealth at the beginning of each backtest period is \$1,000 for all methods considered.

The result appears in Figure 8(a), along with empirical performance on market data for the years of 2017, 2018, and 2019 in panels (b), (c), and (d) of Figures 8, respectively, applying the no-trade zone strategy learned by neural networks. The figures show a promising performance of our combined model in most market conditions. When the market goes up, our algorithm tends to allocate all wealth into stocks and achieves high returns at the rate of the stock market; when the market shrinks, the model detects the downturn with a little lag and generates returns by switching to shorting the market. At the beginning of 2019, the combined method does not do as good as the benchmarks, which is caused by the market drop at the end of 2018. The hidden Markov model has not yet confirmed that early 2019 gets out of the crash regime, and therefore allocation to equities is low with our method. Other than this period, the combined method performs well compared with standard benchmarks.

To test how the strategy works when unpredictable events occur, we carry out empirical tests with the market data in 2020 and see whether our combined method performs well on the unexpected downturn in February and March. In particular, we apply the combined method to the following time windows: (i) January 1, 2019–May 18, 2020, and (ii) January 1, 2016–May 18, 2020. Results are shown in Figure 9, (a) and (b). Some statistics of the performance appear in Table 3. Our combined method tends to have high Sharpe ratio and annual return in most testing periods, with outstanding performance in avoiding losses in crash periods. When the market contracts, the hidden Markov model is capable of identifying the downturn trends. Our combined method shifts the allocation from long to short gradually, as the hidden Markov model becomes more certain of a crash period. With a

Table 2. Average Total Transaction Costs Paid over a 50-Month Period with Different Strategies (One Risky Asset, Monthly Rebalance with 1.5% Transaction Cost, and Two Regimes)

Combined method	Dynamic program	Optimal under-zero t-cost
18.72	16.97	59.02

Figure 8. (Color online) Empirical Tests: Wealth Path for the Years 2016–2019

Notes. (a) Wealth path for 2016. (b) Wealth path for 2017. (c) Wealth path for 2018. (d) Wealth path for 2019.

no-trade zone, our strategy avoids shifting position immediately when a contraction is suspected; rather, it waits as the crash becomes more evident and successfully avoids unnecessary turnovers that lead to transaction costs. In general, the combined method learns a dynamic strategy that holds risky assets in growth periods and shorts during contractions, providing a superior performance than static strategies in market dips with lower drawdowns, even when the market presents nonstationarity.

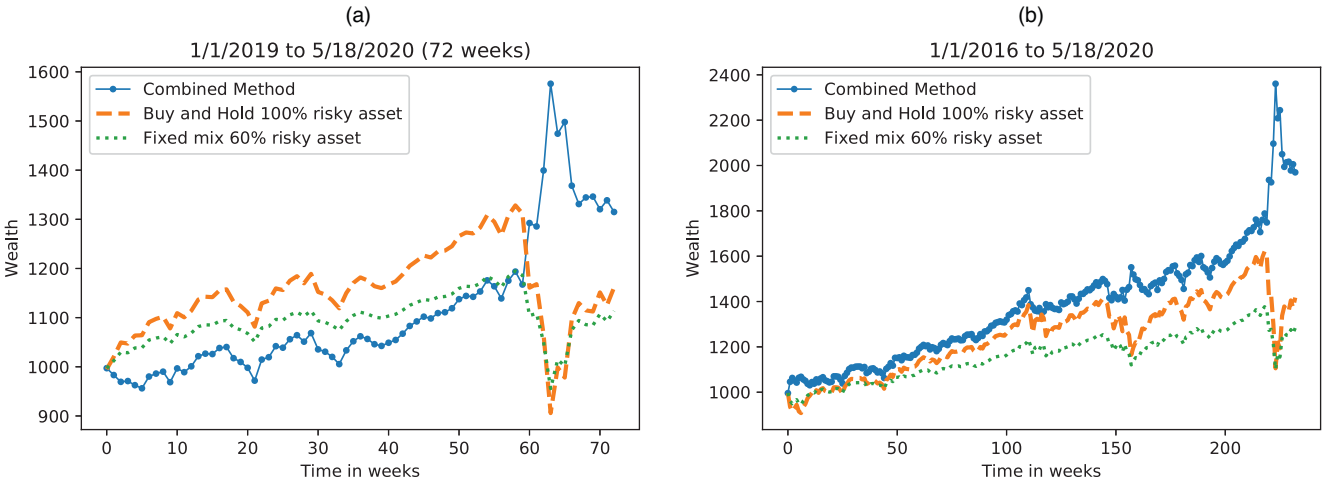
5.2. Empirical Results with Multiple Assets

In an effort to illustrate the power of combining algorithms in the case of multiple risky assets, we first work on an advanced starting point—an approximated optimal allocation strategy under zero transaction costs. It is actually easy to extend our dynamic program for searching the optimal strategy under zero transaction costs to the case involving multiple assets. The state space will not expand due to the increasing number of risky assets, unless we want to consider transaction costs in the dynamic program.

5.2.1. Four Risky Assets. A dynamic program is run in order to approximate the optimal strategy under regime switching and zero transaction costs. We choose the parameters based on four large stocks—AAPL, AMZN, BRK, and GE—from January 1, 2000, to December 14, 2019. For the case of four risky assets, a laptop is able to find the solution in less than one minute if we run five iterations with 50,000 paths in each iteration. The optimal allocation at the beginning of the first period is shown in Figure 10.

As a second step, we feed the results of dynamic programming into a neural network. The new graph with the same structure is described in Section 3.2. Three hidden layers are adopted in each neural network, with the number of neurons being 32, 64, and 128 in each layer, respectively. It turns out that the network can achieve

Figure 9. (Color online) Empirical Tests: Wealth Paths for Time Windows (a) January 1, 2019–May 18, 2020, and (b) January 1, 2016–May 18, 2020



results as good as shown in Figure 11(a). The corresponding average wealth path and no-trade zone is presented in panels (b) and (c) of Figure 11, respectively. The dotted lines are results of the strategy if no transaction cost were charged, and the solid lines are the actual wealth and utility achieved with the linear transaction costs in the model. The no-trade zone strategy learned by the neural network successfully saves the transaction costs and turns them into realized returns. We can see that both the utility and wealth paths are satisfying, and it only takes about 15 minutes to find the solution.

5.2.2. Eleven Risky Assets. The same experiment is carried out on 11 correlated risky assets. The parameters of the hidden Markov model are calibrated on the following stocks, based on weekly returns from January 1, 2000, to December 14, 2019: AAPL, IBM, AMZN, JPM, BRK, GE, JNJ, NSRGY, T, XOM, and WFC.

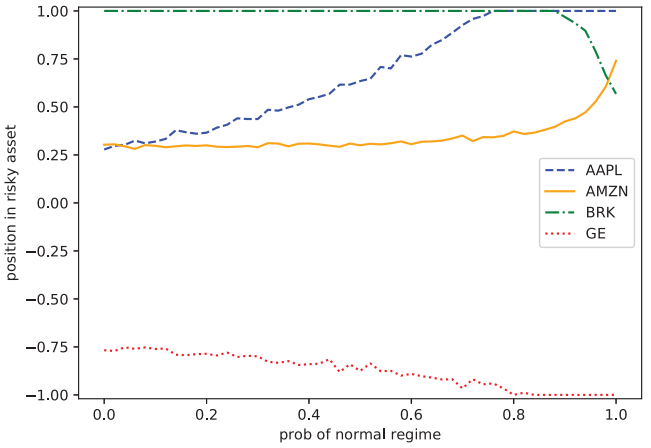
With five iterations and 50,000 paths in each iteration, it takes the dynamic program about 2.3 minutes to find the optimal allocation under zero transaction costs. As before, we feed the solution to a neural network, in this case projecting 64, 128, and 256 neurons in the three hidden layers, respectively. The algorithm outputs satisfactory results, as shown in panels (a) and (b) of Figure 12. We notice from the plots that if an investor rebalances her portfolio actively to the optimal solution under zero transaction costs, even though on average her wealth

Table 3. Statistics of Performance on Empirical Data with Different Strategies

Time period	Method	Sharpe ratio	Annual return	Annual volatility	Max drawdown	Max weekly return	Min weekly return
2016	Combined method	**1.342**	**16.06%**	10.70%	**4.53%**	5.03%	−2.39%
	Buy and hold 100%	0.810	10.84%	12.40%	8.77%	3.80%	−5.96%
	Fixed mix 60%	0.799	7.15%	7.44%	5.24%	2.29%	−3.57%
2017	Combined method	**2.980**	**19.87%**	5.80%	2.27%	2.60%	−1.44%
	Buy and Hold 100%	**2.980**	**19.87%**	5.80%	2.27%	2.60%	−1.44%
	Fixed Mix 60%	2.970	12.19%	3.48%	**1.30%**	1.57%	−0.85%
2018	Combined Method	**0.574**	**7.69%**	12.45%	**6.98%**	5.96%	−4.26%
	Buy and Hold 100%	−0.440	−8.16%	17.96%	17.51%	4.85%	−7.05%
	Fixed Mix 60%	−0.451	−4.12%	10.78%	10.64%	2.91%	−4.23%
2019	Combined Method	1.629	19.26%	10.20%	6.57%	4.41%	−3.10%
	Buy and Hold 100%	**2.215**	**27.33%**	10.72%	6.57%	4.41%	−3.10%
	Fixed Mix 60%	2.204	16.40%	6.43%	**3.94%**	2.65%	−1.85%
January 1, 2019–May 18, 2020	Combined Method	**0.978**	**22.76%**	21.14%	**16.55%**	12.61%	−8.65%
	Buy and Hold 100%	0.513	11.67%	25.15%	31.81%	12.10%	−14.98%
	Fixed Mix 60%	0.504	8.21%	15.10%	20.03%	7.26%	−8.99%
January 1, 2016–May 18, 2020	Combined Method	**1.041**	**22.13%**	14.56%	**16.55%**	12.61%	−8.65%
	Buy and Hold 100%	0.459	9.52%	17.87%	31.81%	12.10%	−14.98%
	Fixed Mix 60%	0.450	6.46%	10.73%	20.03%	7.26%	−8.99%

Note. The bold items in the table indicate the best performance metric for each corresponding period.

Figure 10. (Color online) Optimal Allocation at the Beginning of the First Period vs. the Probability of Being in the Normal Regime



grows, she is actually worse off when utility is considered. In addition, the transaction costs can be so high that the returns are almost entirely consumed: She could have doubled her wealth in 50 weeks with the same strategy if there were no transaction costs; however, when transaction costs incur, the naive rebalancing rewards her with only a slightly more than 10% return. On the other hand, when we apply the combined strategy with no-trade zones, the investor improves her utility over time and earns about a 58% return in 50 weeks.

Further, we present the average terminal wealth for various risk-aversion coefficients (Figure 13). The combined method consistently yields higher wealth than the naive rebalancing strategy, which is optimal only under zero transaction costs. Not surprisingly, the risk-averse, higher-terminal wealth can be expected as the strategies tend to pursue returns.

5.3. Empirical Results with Multiple Regimes

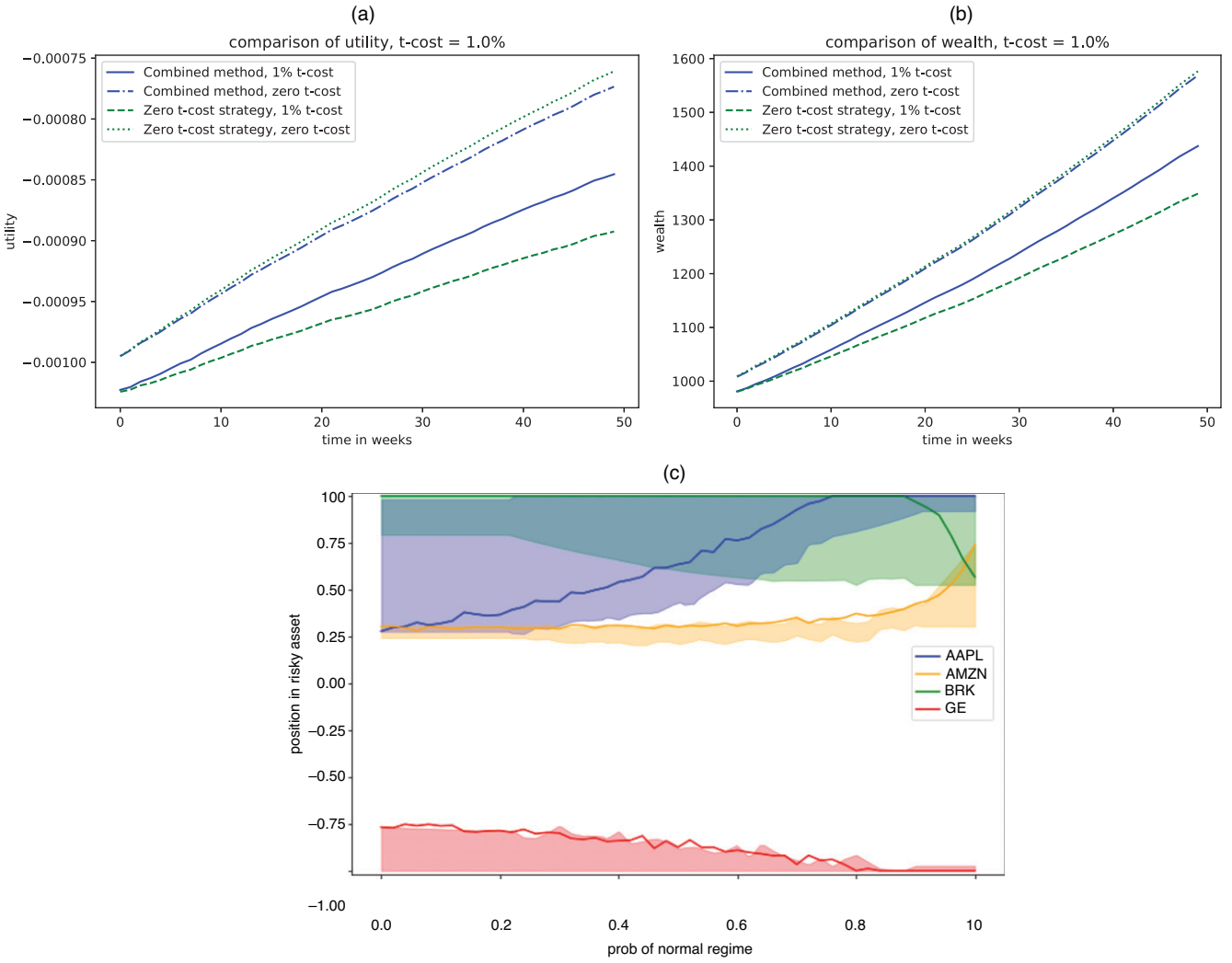
Given an application with frequent data, complex regime patterns may show up in the data. We construct a three-regime hidden Markov model to daily data from the S&P 500 since 2000. The average daily returns of the three regimes are estimated to be 0.00101, −0.00102, and −0.00094, respectively. The variances of the daily returns are estimated to be $4.019 * 10^{-05}$, $1.496 * 10^{-04}$, and $4.208 * 10^{-04}$, respectively. The calibrated transition matrix is $\begin{bmatrix} 0.8194 & 0.1792 & 0.0014 \\ 0.6715 & 0.3032 & 0.0217 \\ 0.0065 & 0.0131 & 0.9804 \end{bmatrix}$. We are interested in finding a strategy over a one-year horizon where the investor can rebalance daily. Assuming 250 trading days in one year, we carry out the algorithm and describe the findings in the next paragraphs.

As before, a dynamic program is carried out first to approximate the optimal strategy under zero transaction costs; then, the solution becomes a starting point for the neural network to identify a no-trade zone. The results are shown in panels (a) and (b) of Figure 14.

The neural network provides promising results in the case of three underlying regimes under a one-year period and daily rebalancing, with an additional 35 minutes to find the no-trade zone on top of the zero-*t*-cost strategy.

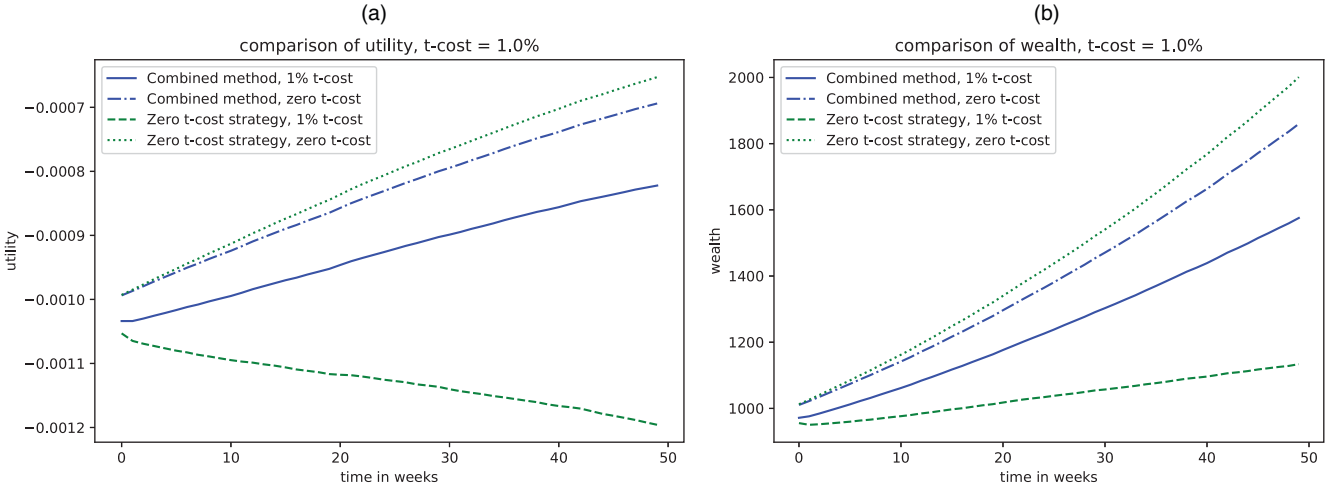
Three underlying regimes pose a much harder problem than the one with two regimes. With two possible regimes, it is sufficient to use one variable to describe the probability of the current regime distribution, and thus the space is one-dimensional, whereas, with three regimes, the space is two-dimensional. It is, in general, difficult for a dynamic programming algorithm to deal with the dimensionality issue caused by the regime space. The dynamic program does not suffer from the curse of dimensionality if the number of risky assets increases, but it suffers if the number of regimes increases. Therefore, the running time of the dynamic program under the case of three regimes is much longer than two regimes but is still acceptable for a problem with such a large size. In fact, it requires about 1.5 hours to find the optimal allocation under zero transaction costs. As an alternative, we suggest that one carefully discretize the space of observed returns and use analytical solutions in Çanakoğlu and Özekici (2012) to get a fast start, even though the method does not capture a leverage limit or possible variations of the riskless asset.

Figure 11. (Color online for panels (a) and (b)) Comparison of Results for the Case of Four Risky Assets Using Strategy Learned by Neural Networking vs. Naive Rebalancing Strategy



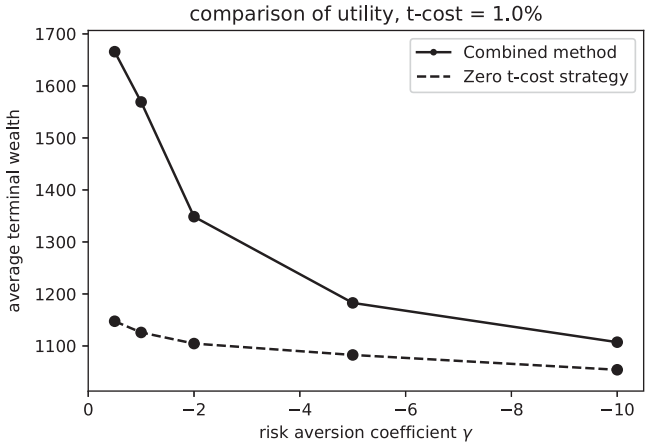
Notes. (a) Utility path. (b) Wealth path. (c) No-trade zone learned by neural network. Each color represents the no-trade zone of an asset.

Figure 12. (Color online) Comparison of Results for the Case of 11 Risky Assets Using Strategy Learned by Neural Networking vs. Naive Rebalancing Strategy



Notes. (a) Utility path. (b) Wealth path.

Figure 13. Average Terminal Wealth for the Case of 11 Risky Assets, with Various Risk-Aversion Coefficients γ



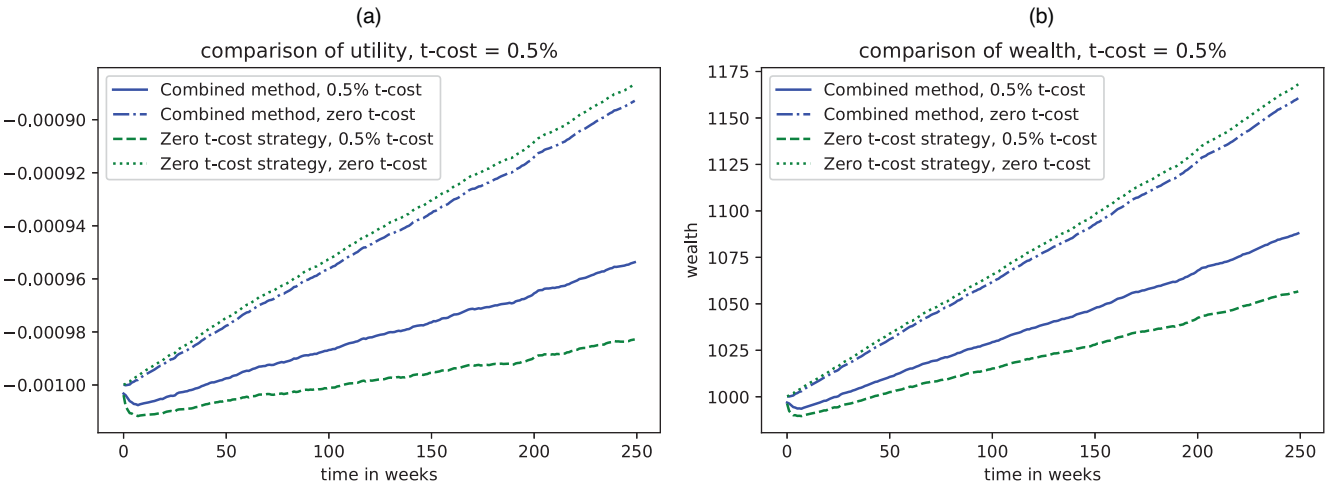
The running times of the neural network are summarized in Table 4. The table indicates a polynomial, if not linear, running time of the neural network with respect to the number of regimes, the number of risky assets, and the number of periods that are considered in the model. Overall, our method successfully avoids the curse of dimensionality and is able to provide satisfactory results.

6. Conclusions

Previous studies have shown that combining a fast-start algorithm with a neural network provides an efficient strategy to many applications (Silver et al. 2016, Mulvey et al. 2020). In this work we propose a method that efficiently finds a trading strategy under an unobservable regime-switching market where linear transaction costs incur. Our algorithm adopts a dynamic program to find the optimal allocation under zero transaction costs and employs the solution as a starting point to a neural network to find the no-trade zone. The algorithm avoids the curse of dimensionality as the number of risky assets increases. It achieves a close-to-optimal solution under the simple case where there are only two possible regimes and one risky asset in the market. For larger and more complicated cases, at this point we do not have the obvious tools to find an approximately optimal strategy, since traditional methods suffer badly from the curse of dimensionality. On the other hand, our approach successfully shortens the running time and provides satisfactory results in the presented cases.

As future research, the discussed combined approach (dynamic programming and recurrent neural networks) promises to improve existing algorithms for applications such as systematic trading models (Almgren and Chriss

Figure 14. (Color online) Comparison of Results for the Case of Three Underlying Regimes, Using Strategy Learned by Neural Networking vs. Naive Rebalancing Strategy



Notes. (a) Utility path. (b) Wealth path.

Table 4. Running Time of Neural Network Under Different Cases

Number of regimes	Number of risky assets	Number of periods	Neurons in hidden layers	Number of steps trained	Running time (minutes:seconds)
2	1	50	[20,40,80]	500	3:30
2	4	50	[32,64,128]	1,000	14:31
2	11	50	[64,128,256]	1,000	28:51
3	1	250	[20,40,80]	1,000	35:00

2001). These high-frequency decisions may be addressed by means of online learning. Importantly, a careful linkage of dynamic optimization algorithms and machine learning will be an active domain going forward.

References

Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, et al. (2015) TensorFlow: Large-scale machine learning on heterogeneous systems, www.tensorflow.org/about/bib.

Almgren R, Chriss N (2001) Optimal execution of portfolio transactions. *J. Risk* 3(2):5–40.

Arrow KJ (1971) The theory of risk aversion. *Essays in the Theory of Risk-Bearing* (Markham, Chicago), 90–120.

Bauerle N, Rieder U (2004) Portfolio optimization with Markov-modulated stock prices and interest rates. *IEEE Trans. Automatic Control* 49(3): 442–447.

Bauerle N, Rieder U (2007) Portfolio optimization with jumps and unobservable intensity process. *Math. Finance* 17(2):205–224.

Bellman R (1954) The theory of dynamic programming. *Bull. Amer. Math. Soc. (N.S.)* 60(6):503–515.

Bertsimas D, Pachamanova D (2008) Robust multiperiod portfolio management in the presence of transaction costs. *Comput. Oper. Res.* 35(1): 3–17.

Birge JR (2007) Optimization methods in dynamic portfolio management. Birge JR, Linetsky V, eds. *Financial Engineering*, vol. 15 (Elsevier, Amsterdam), 845–865.

Birge JR (2012) Particle methods for data-driven simulation and optimization. *TutORials in Operations Research* (INFORMS, Catonsville, MD), 92–102.

Çanakoğlu E, Özekici S (2009) Portfolio selection in stochastic markets with exponential utility functions. *Ann. Oper. Res.* 166(1):281.

Çanakoğlu E, Özekici S (2012) Hara frontiers of optimal portfolios in stochastic markets. *Eur. J. Oper. Res.* 221(1):129–137.

Casas CA (2001) Tactical asset allocation: an artificial neural network based model. *Proc. Internat. Joint Conf. Neural Networks* (IEEE, Piscataway, NJ), 1811–1816.

Chung J, Gulcehre C, Cho K, Bengio Y (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. NIPS 2014 Workshop Deep Learning. Preprint, submitted December 11, https://arxiv.org/abs/1412.3555.

Connor JT, Martin RD (1994) Recurrent neural networks and robust time series prediction. *IEEE Trans. Neural Networks* 5(2):240–254.

Davis MH, Norman AR (1990) Portfolio selection with transaction costs. *Math. Oper. Res.* 15(4):676–713.

Dumas B, Luciano E (1991) An exact solution to a dynamic portfolio choice problem under transactions costs. *J. Finance* 46(2):577–595.

Elliott RJ, Siu TK (2009) Robust optimal portfolio choice under Markovian regime-switching model. *Methodology Comput. Appl. Probab.* 11(2): 145–157.

Han J, E W (2016) Deep learning approximation for stochastic control problems. Preprint, submitted November 2, https://arxiv.org/abs/1611.07422.

Hejazi SA, Jackson KR (2016) A neural network approach to efficient valuation of large portfolios of variable annuities. *Insurance Math. Econom.* 70:169–181.

Kamijo Ki, Tanigawa T (1990) Stock price pattern recognition-a recurrent neural network approach. *Proc. Internat. Joint Conf. Neural Networks* (IEEE, Piscataway, NJ), 215–221.

Kolm PN, Ritter G (2014) Multiperiod portfolio selection and Bayesian dynamic models. *Risk* 28(3):50–54.

Li Y, Forsyth PA (2019) A data-driven neural network approach to optimal asset allocation for target based defined contribution pension plans. *Insurance Math. Econom.* 86:189–204.

Liu RH (2014) A finite-horizon optimal investment and consumption problem using regime-switching models. *Internat. J. Theoret. Appl. Finance* 17(4):1–18.

Loshchilov I, Hutter F (2016) SGDR: Stochastic gradient descent with warm restarts. Preprint, submitted August 13, https://arxiv.org/abs/1608.03983.

Merton RC (1969) Lifetime portfolio selection under uncertainty: The continuous-time case. *Rev. Econom. Statist.* 51(3):247–257.

Mulvey JM, Liu H (2016) Identifying economic regimes: Reducing downside risks for university endowments and foundations. *J. Portfolio Management* 43(1):100–108.

Mulvey JM, Lu N, Sweemer J (2001) Rebalancing strategies for multi-period asset allocation. *J. Wealth Management* 4(2):51–58.

Mulvey JM, Sun Y, Wang M, Ye J (2020) Optimizing a portfolio of mean-reverting assets with transaction costs via a feedforward neural network. *Quant. Finance* 20(8):1–23.

Nystrup P, Madsen H, Lindström E (2018) Dynamic portfolio optimization across hidden market regimes. *Quant. Finance* 18(1):83–95.

Nystrup P, Boyd S, Lindström E, Madsen H (2019) Multi-period portfolio selection with drawdown control. *Ann. Oper. Res.* 282(1–2):245–271.

Nystrup P, Hansen BW, Madsen H, Lindström E (2015) Regime-based vs. static asset allocation: Letting the data speak. *J. Wealth Management* 42(1):103–109.

Palczewski J, Poulsen R, Schenk-Hoppé KR, Wang H (2015) Dynamic portfolio optimization with transaction costs and state-dependent drift. *Eur. J. Oper. Res.* 243(3):921–931.

Pratt JW (1964) Risk aversion in the small and in the large. *Econometrica* 32(1–2):122–136.

Samuelson PA (1969) Lifetime portfolio selection by dynamic stochastic programming. *Rev. Econom. Statist.* 5(3):239–246.

Schuster M, Paliwal KK (1997) Bidirectional recurrent neural networks. *IEEE Trans. Signal Processing* 45(11):2673–2681.

Silver D, Huang A, Maddison CJ, Guez A, Sifre L, Driessche GVD, Schrittwieser J, et al. (2016) Mastering the game of go with deep neural networks and tree search. *Nature* 529(7587):484–489.

Sotomayor LR, Cadenillas A (2009) Explicit solutions of consumption investment problems in financial markets with regime switching. *Math. Finance* 19(2):251–279.

Zhou XY, Yin G (2003) Markowitz's mean-variance portfolio selection with regime switching: A continuous-time model. *SIAM J. Control Optim.* 42(4):1466–1482.