Time-Varying Minimum-Cost Portfolio Insurance under Transaction Costs Problem via Beetle Antennae Search Algorithm (BAS)

,

Abstract

Portfolio insurance is a hedging strategy which is used to limit portfolio losses without having to sell off stock when stocks decline in value. Consequently, the minimization of the costs related to portfolio insurance is a very important investment strategy. On the one hand, a popular option to solve the static minimum-cost portfolio insurance problem is based on the use of linear programming (LP) methods. On the other hand, the static portfolio selection under transaction costs (PSTC) problem is usually approached by nonlinear programming (NLP) methods. In this article, we define and study the time-varying minimum-cost portfolio insurance under transaction costs (TV-MCPITC) problem in the form of a time-varying nonlinear programming (TV-NLP) problem. Using the Beetle Antennae Search (BAS) algorithm, we also provide an online solution to the static NLP problem. The online solution to a time-varying financial problem is a great technical analysis tool and along with fundamental analysis will enable the investors to make better decisions. To the best of our knowledge, an approach that incorporates modern meta-heuristic optimization techniques to provide a more realistic online solution to the TV-MCPITC problem is original. In this way, by presenting an online solution to a time-varying financial problem we highlight the limitations of static methods. Our approach is also verified by numerical experiments and computer simulations as an excellent alternative to conventional MATLAB methods.

Keywords: Portfolio constrained optimization; time-varying transaction costs; time-varying nonlinear programming; nature-inspired algorithms; beetle search optimization.
2020 Msc: 90C30, 90C90, 90C59, 91G10.

1 Introduction

In financial models, cost minimization in portfolios is always of great importance. Popular fields include insurance costs, risk management, option replication, transaction costs, etc. and can be approached efficiently using conventional optimization methods. For example, a new approach for modeling and approximating the value of portfolios of interdependent real options based on the use of both influence diagrams and simulation-and-regression is presented in [1]. This optimization problem is approached with a transparent valuation algorithm that explicitly takes into account vector-valued exercise decisions and the state variables multidimensional resource component which generally occur in real option portfolios. In [2], the authors develop a

new trapezoidal fuzzy numbers with an adaptive index in order to capture the coherence of the investor's expectation. In this way, the membership degrees for favorable and unfavorable scenarios are transformed consistently to avoid the logical confusion. They approach these new trapezoidal fuzzy numbers with a fuzzy mean-variance model and mean-variance-skewness model for optimal asset allocation. Also, in [3] the classic theory of portfolio selection in both of its branches, deciding the efficient financial positions between such a set of choices and selecting the financial position that maximizes some utility function whose functional type involves some risk measure. In [4], a dynamic proportion portfolio insurance (DPPI) strategy based on the popular constant proportion portfolio insurance (CPPI) strategy is proposed. The analysis of this strategy is approached with genetic programming which uses risk variables arising from the market conditions in order to build the equation tree for the risk multiplier.

In this paper, we define and study time-varying versions of the minimum-cost portfolio insurance (MCPI) problem and the portfolio selection under transaction costs (PSTC) problem. The time-varying MCPI and the time-varying PSTC financial problems are novel approaches with respect to the corresponding static financial problems. We also define and study the time-varying minimum-cost portfolio insurance under transaction costs (TV-MCPITC) financial problem. Note that, the aforementioned time-varying financial problems have never been defined before in the bibliography.

The main financial problem here is the TV-MCPITC problem, which is an NLP problem. Such an NPL problem can be efficiently solved by using the `fmincon`, `GA`, and `particleswarm` MATLAB functions in a MATLAB environment. We also approach the TV-MCPITC problem with a modified version of the Beetle Antennae Search (BAS) algorithm. BAS is nature-inspired meta-heuristic optimization algorithm capable of efficient global optimization and, over the last years, it has extensively been used in several scientific fields

(see [5–9]). For example, in [5], the authors presented a nonconvex model for the portfolio selection problem under transaction costs and cardinality constraints and approached it with BAS algorithm. In [8], a predictive collision avoidance method for underactuated surface vessels is proposed and approached by an improved BAS algorithm, which enhances the optimization performances of the original BAS algorithm. Our modified time-varying version of BAS is called TV-BAS and we apply it in a MATLAB environment. The advantages of the TV-BAS algorithm are that it is faster than the aforementioned MATLAB functions with similar efficiency and that it can easily be implemented in different programming languages. In addition, we construct some popular interpolation methods which make the solutions of TV-BAS, `fmincon`, `GA`, and `particleswarm` MATLAB functions even more efficient than if we had used the corresponding MATLAB functions with standard inputs. In conclusion, we propose a well-tuned method of solving such problems.

The paper is organized as follows. Section 2 describes the time-varying minimum-cost portfolio insurance problem, the time-varying transaction costs portfolio selection problem and the TV-MCPITC optimization problem. In section 3, the TV-MCPITC problem is approached by a properly modified meta-heuristic BAS algorithm. Section 4 describes the proposed algorithmic procedures for data preparation. Section 5 describes the main algorithm (TV-BAS) for solving the TV-MCPITC problem and section 6 contains the numerical examples. The numerical examples use real-world data and examine the efficiency of the TV-BAS, `fmincon`, `GA`, and `particleswarm` MATLAB functions as well as the efficiency of the proposed algorithmic procedures in different portfolios settings. Finally, the concluding remarks are presented in section 7.

2 Minimum-Cost Portfolio Insurance under Transaction Costs Problem

In finance, a collection of financial assets that investors own is known as a portfolio. Portfolio optimization plays a significant role in financial decisions. One way to minimize the expense of a portfolio is to reduce the cost of insurance (see [2, 10–18]). For example, an optimization problem to minimize the cost of portfolio insurance is defined in [10]. The problem constructs a portfolio which replicates the targeted payoff in a subset of states if the asset span is a lattice-subspace and it is approached with the theory of Riesz spaces. The performances of the two major portfolio insurance strategies: option-based portfolio insurance (OBPI) and CPPI, are compared in [16] by using the stochastic dominance approach. They notice that the CPPI method can perform better than the OBPI method in the third order stochastic dominance. The researchers in [17] examine the use of leveraged exchange traded funds (LETFs) in the context of a CPPI strategy. The

benefit of using LETFs in solving such a strategy is that it makes the allocation of a higher percentage of the portfolio in the risk-free rate compared to a conventional CPPI. In [18], the authors explain how portfolio insurance works, the key strategies used, their development in recent years, and possible links between their use and the stability of the financial market. They also point out that the key advantage of portfolio insurance is that it facilitates the distribution of financial risk among those entities most capable of absorbing it. Also, it is pointed out that the downside is that it can create conditions for increased fragility in the financial market, making issuers of portfolio insurance exposed to potentially unexpectedly high losses. In general, portfolio insurance is a dynamic hedging strategy that emphasizes buying and selling securities periodically to maintain a limit of the portfolio value.

In this section, we define the time-varying minimum-cost portfolio insurance (TV-MCPI) problem as well as the time-varying portfolio selection under transaction costs (TV-PSTC) problem. The TV-MCPI problem minimizes the insurance costs of a portfolio while keeping its payoff above a floor price. The TV-PSTC problem minimizes the transaction costs of a portfolio while trying to achieve maximum payoff. By combining the TV-MCPI and TV-PSTC, we also define the TV-MCPITC problem. The TV-MCPITC problem minimizes the insurance costs and the transaction costs of a portfolio while trying to achieve maximum payoff and keeping that payoff above a floor price at the same time.

2.1 Definition of the TV-MCPI Financial Problem

Our approach to the portfolio insurance problem is a time-varying analog of the corresponding static problem defined and studied in a number of papers, such as [10–14]. As far as we are aware of, our time-varying version of the portfolio insurance problem is a novel approach that comprises modern meta-heuristic optimization techniques to provide an online, thus more realistic, solution to the TV-MCPITC problem.

The *space of marketed securities* is $X = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{m\times n}$ where $x_i \in \mathbb{R}^m$ is the security i, where $i = 1, 2, \ldots, n$, and comprises data from the last m observations of its price. In the discrete TV-MCPI problem, we convert the *space of marketed securities* to a time-varying vector $X(t) = [x_1(t), x_2(t), \ldots, x_n(t)] \in \mathbb{R}^n$ where $x_i(t) \in \mathbb{R}$ is the security i, $i = 1, 2, \ldots, n$, and the time $t \in [1, m]$ denotes the new value that it comes during the calculation of the solution to the TV-MCPI problem.

A *portfolio* is a vector $\eta_p = (\eta_{p_1}, \eta_{p_2}, \ldots, \eta_{p_n})$ of $\mathbb{R}^n$ where η_{pi} is the number of shares of the ith security. In a two period model, if $\eta_p = (\eta_{p_1}, \eta_{p_2}, \ldots, \eta_{p_n})$ is a portfolio that is not zero at the time $t = 0$, then its payoff at the time $t = 1$ is given by the formula

$$G(\eta_p) = \sum_{i=1}^{n} \eta_{p_i} x_i(1), \tag{2.1}$$

where $x_i(1) \in \mathbb{R},\ i = 1, 2, \ldots, n$. The G operator is one-to-one and is called the *payoff operator*. In our m period model, we consider the portfolio $\eta_p = \eta(0)$ as the initial portfolio and $\eta(t) \in \mathbb{R}^n$, $t \in [1, m]$, as the requested one. Here we have two versions of the G operator. One version is the operator G_1 which is identical to the operator G and the other version is G_2 which removes the insurance costs of the previous period. Hence, we have

$$G_1(\eta(t-1)) = \sum_{i=1}^{n} \eta(t-1) x_i(t)$$

and

$$G_2(\eta(t-1)) = \sum_{i=1}^{n} \eta(t-1) x_i(t) - \sum_{i=1}^{n} \eta(t-1) p(t-1),$$

where $p(t) = (p_1(t), p_2(t), \ldots, p_n(t)) \in \mathbb{R}^n$ is a vector of time-varying security prices and $\sum_{i=1}^{n} \eta(t-1)p(t-1)$ is the insurance cost of the previous period. Note that it must hold $\sum_{i=1}^{n} \eta(0)p(0) = 0$ in order to be true the Equation (2.1). Consequently, for $t = 1$ we have that $G_1 = G_2 = G$.

If we also have a "floor" price $\phi(t) \in \mathbb{R}$ then the insured payoff on the $\eta(t-1)$ portfolio at the $\phi(t)$ "floor" and in the $p(t)$ price is the supremum $G_2(\eta(t-1)) \vee \phi(t)$. The *time-varying minimum-cost insured portfolio* $\eta(t)$ at the floor $\phi(t)$ and in the price $p(t)$ is the solution to the following cost minimization constraint problem:

$$\begin{aligned} &\min_{\eta(t)} && p^{\mathrm{T}}(t)\cdot\eta(t)\\ &\text{subject to} && G_1(\eta(t))\geq G_2(\eta(t-1))\vee\phi(t). \end{aligned}$$

This problem is the TV-MCPI and it can also be written in the following time-varying LP (TVLP) form:

$$\min_{\eta(t)} \quad p^{\mathrm{T}}(t)\cdot\eta(t) \tag{2.2}$$

$$\text{subject to} \quad -X^{\mathrm{T}}(t)\cdot\eta(t)\leq\min\{-\text{payoff},-\phi(t)\} \tag{2.3}$$

$$0\leq\eta(t)\leq X^{\mathrm{T}}(t)\cdot\eta(t-1)\cdot\left[\frac{1}{x_1(t)},\ldots,\frac{1}{x_n(t)}\right], \tag{2.4}$$

where payoff $=(X^T(t)-p^{\mathrm{T}}(t-1))\cdot\eta(t-1)$.

We convert the discrete TV-MCPI problem to continuous-time form by interpolating the $X(t)$, $p(t)$ and the $\phi(t)$ into continuous functions with any method of preferences (see Subsection 4.1). Consequently, $X(t),\ p(t),\ \phi(t)\in C[0,m-1]$, where $C[0,m-1]$ is the space of all continuous real functions on the interval $[0,m-1]$. The optimal minimum-cost insured portfolio is equal to $\eta(t)=[\eta_1(t),\ldots,\eta_n(t)]$, where $\eta(t)$ is the online solution.

2.1.1 Insurance Pricing

The amount of money an individual or business must pay for an insurance policy is termed as insurance premium. The insurance premium for any asset also depends on the degree of risk that it carries. Risk represents the probability that the actual return may differ from the expected return. In order to be more realistic, we assume that the insurance costs of our portfolio comprise of a fixed charge plus a rate of the variance (risk) of the assets. Let θ be the price rates associated with the risk of assets and ζ be the fixed price. The fixed-plus-linear time-varying insurance prices function is given by

$$p(t)=\zeta+\theta\mathrm{Var}\left[\frac{Y}{\max(Y)}\right], \tag{2.5}$$

where $Y=X(t-\tau:t,:)$. The number $\tau\leq m-1$, $\tau\in\mathbb{N}$, is a constant number and it denotes the 'number of time periods'. The risk of the asset is calculated by the variance of its last τ normalized prices.

2.2 Definition of the TV-PSTC Financial Problem

Transaction costs, such as brokerage commissions, bid-ask spreads, taxes or even fund loads, can be used to model a variety of costs. In this paper, we consider the transaction costs to be separable as presented in [19]. That is, the sum of the transaction costs associated with individual trades is equal to

$$\kappa(t)=\sum_{i=1}^{n}\kappa_i(t), \tag{2.6}$$

where κ_i is the transaction cost function for asset i. In reality, transaction costs are nonconvex functions of the traded amount. Actually, the costs of either buying or selling would probably be concave. For example, a fixed price for any non-zero exchange is ordinary, and there could exist one or more breakpoints at which the transaction cost per share may depreciate. We assume a simplistic model that involves fixed plus linear costs. But our approach is expanded to handle more complex transaction cost functions. Let α^+, α^- be the cost rates related to buying and selling asset i and β^+, β^- the fixed costs associated with buying and selling asset i. The fixed-plus-linear time-varying transaction cost function is given by

$$\kappa_i(t)=\begin{cases} 0, & \eta_i(t)=\eta_i(t-1)\\ \beta_i^+ +\alpha_i^+(\eta_i(t)-\eta_i(t-1))x_i(t), & \eta_i(t)>\eta_i(t-1)\\ \beta_i^- +\alpha_i^-(\eta_i(t-1)-\eta_i(t))x_i(t), & \eta_i(t)<\eta_i(t-1)\end{cases} \tag{2.7}$$

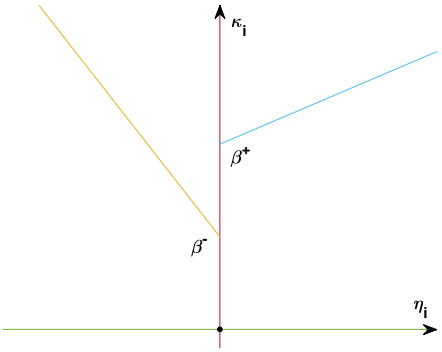


Figure 1: Fixed plus linear transaction costs $\kappa_i(t)$ as a function of transaction amount $(\eta_i(t) - \eta_i(t-1))x_i(t)$. There is no cost if there is not a transaction, i.e., $\kappa_i(t) = 0$.

where $x_i(t)$ is the price of the i stock at time t. Fig. 1 shows the transaction cost function. This function is clearly nonconvex, except in the case of zero fixed costs.

2.2.1 The TV-PSTC problem

A related problem of Subsection 2.1 is the minimization of the total transaction costs subject to portfolio constraints. Among all possible transactions that produce portfolios achieving a given expected return and satisfying portfolio constraints, we would like to select those which generate the smallest total cost. Hence, the TV-PSTC problem is written as follows:

$$\begin{aligned} &\min_{\eta(t)} && \kappa(t) && (2.8)\\ &\text{subject to} && \sum_i \eta_i(t) \cdot r_i(t) \geq r_p(t) && (2.9)\\ &&& \eta_i(t) \in \mathbb{R}_0^+, \ \forall i, && (2.10) \end{aligned}$$

where $\kappa(t)$ is defined in (2.6) and $\mathbb{R}_0^+$ denotes non-negative real numbers.

2.3 The TV-MCPITC Financial Problem

Combining the two aforementioned problems of subsections 2.1 and 2.2, the TV-MCPITC problem is formulated as follows:

$$\begin{aligned} &\min_{\eta(t)} && p^{\mathrm{T}}(t) \cdot \eta(t) + \kappa(t) && (2.11)\\ &\text{subject to} && G_1(\eta(t)) \geq (G_2(\eta(t-1)) - \kappa(t-1)) \vee \phi(t) && (2.12)\\ &&& \eta_i(t) \in \mathbb{R}_0^+, \ \forall i. && (2.13) \end{aligned}$$

This problem can also be written in the TV-NLP form as follows:

$$\min_{\eta(t)} \quad p^{\mathrm{T}}(t)\cdot\eta(t)+\kappa(t) \tag{2.14}$$

$$\text{subject to} \quad -X^{\mathrm{T}}(t)\cdot\eta(t)\le\min\{\kappa(t)-\text{payoff},-\phi(t)\} \tag{2.15}$$

$$0\le\eta(t)\le X^{\mathrm{T}}(t)\cdot\eta(t-1)\cdot\left[\frac{1}{x_1(t)},\ldots,\frac{1}{x_n(t)}\right], \tag{2.16}$$

where

$$\text{payoff}=(X^{T}(t)-p^{\mathrm{T}}(t-1))\cdot\eta(t-1).$$

Note that we remove the previous insurance costs from the portfolio's payoff and also remove the previous transaction cost. Hence, the TV-MCPITC problem becomes more realistic. The following algorithmic procedure defines the MATLAB function for generating the right hand side in (2.14).

Algorithm 1 Algorithmic procedure for Eq. (2.14).

Input: The insurance prices $p=p(t)$, the stock prices $A=A(t)$, the portfolio $\eta=\eta(t)$ and the previous $\eta_{-1}=\eta(t-1)$, the fixed costs β^{-}, β^{+} and the costs rates α^{-}, α^{+}.

1: **function** f = minfunc(η, η_{-1}, p, A, β^{-}, β^{+}, α^{-}, α^{+})

2: Set $f=p'\eta+sum((\eta>\eta_{-1}).(\beta^{+}+\alpha^{+}(\eta-\eta_{-1}).A')+(\eta<\eta_{-1}).(\beta^{-}+\alpha^{-}(\eta_{-1}-\eta).A'))$

3: **return** f

4: **end function**

Output: The function in (2.14).

3 Time-Varying Minimum-Cost Portfolio Insurance under Transaction Costs Problem via BAS

3.1 Penalty Function

Penalty function methods work in a series of sequences (see [20]), each time modifying a set of penalty parameters and starting a new sequence with the previous solution. The following penalty function is minimized during the construction of any sequence:

$$P(x,R)=f(x)+\Omega(R,g(x),h(x)), \tag{3.1}$$

where R is a set of penalty parameters, Ω is the penalty term chosen to favor the selection of feasible points over infeasible points, $f(x)$ is the function that we want to minimize, $g(x)$ and $h(x)$ are the inequality and equality constraint functions, respectively. Different penalty terms are used for equality or inequality constraints.

The first sequence starts with a small value of the penalty parameter R, which increases in subsequent sequences. Changing the penalty parameter R in successive penalty function method sequences depends on whether an exterior or an interior penalty term is being used. If the optimum point of the unconstrained objective function is the true optimum of the constrained problem, an initial penalty parameter $R=0$ (or any other value of R) will solve the constrained problem. Otherwise, if the constraints make the optimum of the unconstrained objective function infeasible, a number of sequences of the unconstrained optimization algorithm must be applied on a penalized objective function. When this happens, the constrained optimum point is usually a boundary point. The main advantage of this method is that it makes possible to handle any constraints (convex or nonconvex).

In this paper, we approach the TV-MCPITC problem via a modified version of a BAS algorithm. BAS is a nature-inspired meta-heuristic optimization algorithm capable of efficient global optimization. In our modified version of BAS, we make use of the aforementioned penalty functions. So, in our case, we only use the penalty function (3.1) as it is and the user sets the initial value of the penalty parameter R, which stays constant through all the sequences that BAS generates. By adding the penalty function inside the BAS algorithm as presented in subsection 3.2, we are able to make a modified BAS algorithm even more efficient

than the original. The reason is that the penalty function allows the BAS method to handle more efficiently any constraints (convex or nonconvex). Apart from that, the philosophy behind the BAS algorithm stays the same as presented in [21].

We combine the penalty function with the *Bracket operator penalty*,

$$\Omega = R\langle g_j(x)\rangle^2, \tag{3.2}$$

where $\langle k\rangle = k$, if k is negative, otherwise $\langle k\rangle = 0$. This term handles inequality constraints. The bracket operator is an exterior penalty term because it applies a positive value to the infeasible points. This operator is primarily used to handle the inequality constraints. Corresponding algorithmic procedure is presented as a MATLAB function defined in Algorithm 2.

Algorithm 2 Penalty function algorithm for TV-MCPITC.

Input: The requirements of Algorithm 1 plus the penalty parameter R, b the right part of Eq. (2.15) and the lower limit η^- and upper limit η^+ of Eq. (2.16), respectively.

1: **function** P = penfunc(η, η_{-1}, p, A, b, R, η^-, η^+, β^-, β^+, α^-, α^+)
2: Set $\Omega = R(sum((A\eta > b).(-A\eta + b)^2 + (\eta^- > \eta).(\eta^- - \eta)^2 + (\eta > \eta^+).(\eta - \eta^+)^2))$
3: Set $P = minfunc(\eta, \eta_{-1}, p, -A, \beta^-, \beta^+, \alpha^-, \alpha^+) + \Omega$
4: **return** P
5: **end function**

Output: The outcome of Penalty Function.

3.2 Modified BAS Algorithm for solving the TV-MCPITC Problem

A meta-heuristic BAS optimization algorithm follows the searching behavior of a beetle. It is presented in [21]. BAS is a nature-inspired meta-heuristic optimization algorithm. The way beetle uses two antennae to track food is based on the intensity of the smell they detect on antennae. The BAS algorithm mimics such behavior to find the optimal solution to the problem. The searching behavior of beetles with two antennae could be formulated in such a way that it is associated with an objective function to be optimized. Such a strategy make it possible to introduce new optimization algorithms (see [22–25]).

In the following, we modified BAS algorithm by using the two MATLAB functions presented in Algorithms 1 and 2. Namely, function `minfunc` implements Algorithm 1 and function `penfunc` implements Algorithm 2. Note that the dots in the input arguments of functions `minfunc` and `penfunc` in Algorithm 3 mean that the rest of the input arguments stay the same as they are declared in algorithms 1 and 2.

Algorithm 3 BAS algorithm for the TV-MCPITC problem.

Input: The requirements of Algorithms 1 and 2 plus the parameters d, δ, tol and $kmax$.

1: **function** [η,f_η] = basfunc(η_{-1}, p, A, b, R, η^-, η^+, β^-, β^+, α^-, α^+, d, δ, tol, $kmax$)
2: Set $y_1 = \eta_{-1}$ and $y_2 = (nan)ones(size(\eta_{-1}))$
3: Set $k = 0$ and $len = length(\eta_{-1})$
4: **while** $k < kmax$ **OR** $\|penfunc(y_2, \dots) - penfunc(y_1, \dots)\|_2 > tol$ **do**
5: Set $b = rands(len, 1)$ and $b = \frac{b}{2^{-52}+\|b\|}$
6: Set $y_r = y_1 - db$ and $y_l = y_1 + db$
7: Set $y = |y_1 + \delta b(sign(penfunc(y_r, \dots) - penfunc(y_l, \dots)))|$
8: **if** $penfunc(y, \dots) < penfunc(y_1, \dots)$ **then**
9: Set $y_2 = y_1$ and $y_1 = y$
10: **end if**
11: Set $d = 0.991d + 0.001$, $\delta = 0.991\delta$ and $k = k + 1$
12: **end while**
13: **return** $\eta = y_1$, $f_\eta = minfunc(y_1, \eta_{-1}, p, -A, \dots)$
14: **end function**

Output: $\eta = \eta(t)$, f_η.

4 Data Preparation

The following Algorithm 4 shows how we construct the insurance prices vector of subsection 2.1.1 in order to be used in Subsection 4.1.

Algorithm 4 Algorithm for the data preparation of the portfolio's expected return and covariance.

Input: The marketed space $X = [x_1, x_2, \ldots, x_n]$ which is a matrix of n time series as column vectors of m prices; the number of time periods $\tau \leq m - 1$ $\tau \in \mathbb{N}$; the cost rates θ associated with the risk of assets, and the fixed costs ζ of Eq. (2.5).

1: Set $[m, n] = size(X)$
2: Set $p = zeros(m - \tau, n)$
3: **for** $i = 1 : m - \tau$ **do**
4: Set $Y = X(i : \tau + i - 1, :)$
5: Set $p(i, :) = \zeta + \theta \text{var}(Y / \max(Y))$
6: **end for**

Output: The matrix p composed of the insurance prices for a number of time periods of each portfolio's time-series.

4.1 Data Interpolation

The data inputs in the TV-MCPITC optimization model are time-series. A time-series is a sequence of time-indexed data points, which means that the data input in the TV-MCPITC is initially discrete. Because we are trying to find the online solution to a time-varying optimization problem, we must convert those data inputs from discrete to corresponding continuous-time form. We achieve that successfully by transforming arrays and matrices of time-series to continuous-time functions.

In subsections 4.1.1, 4.1.2 and 4.1.3 we suggest some popular interpolation methods that are also offered by *MathWorks*, and we show clearly how to use them with BAS in order to produce faster results in the case where time-series are input data. Using our custom interpolation functions, Algorithms 5, 6 and 7 demonstrate the transformation of arrays and matrices which comprise of time series into an interpolated time function. The data that were used in Figures 2 and 3 are the daily close prices of *NASDAQ Composite (^IXIC)* in the year 2019. We check our methods, for comparison purposes, against the solutions produced by the `fmincon`, `GA`, `particleswarm` MATLAB functions.

The MATLAB function `fmincon` is an NLP solver which finds the minimum of a nonlinear function under nonlinear equality and inequality constraints. This function is a gradient-based method designed to operate on problems where both objective and constraint functions are continuous as well as their first derivatives. The MATLAB function `GA` finds the local minimum of a function under linear equality and inequality constraints by using genetic algorithm (GA). GA is a meta-heuristic algorithm inspired by the process of natural selection. The MATLAB function `particleswarm` is a derivative-free global optimum solver which finds the local minimum of a function. PSO is inspired by the surprisingly organized behaviour of large groups of simple animals, such as flocks of birds, schools of fish, or swarms of locusts. The MATLAB functions `GA`, `particleswarm` and TV-BAS are perfect candidates for comparison because all of them include nature-inspired meta-heuristic optimization algorithms. We also compare obtained results with the MATLAB function `fmincon` because it is assumed to find the optimum solution to an NLP problem that the functions `GA`, `particleswarm` and TV-BAS must match.

4.1.1 Linear Interpolation

The linear interpolation is a popular choice to create the prices between two real-valued observation times of a time-series. We apply linear interpolation in our data input by making use of the two point-slope equation of a line $y - y_1 = \lambda(x - x_1)$, where $\lambda = \frac{y_2 - y_1}{x_2 - x_1}$. Since $x_2 - x_1 = 1$ is always valid for two consecutive points in a time-series, the points between them can be interpolated using $y = y_1 + (y_2 - y_1)(x - x_1)$. In Algorithm 5, we propose a procedure for the implementation of the previous ideas. The outcome is illustrated in Figure 2.

Algorithm 5 Algorithm for the linear interpolation

Input: The marketed space $X = [x_1, x_2, \ldots, x_n]$, which is a matrix of n time series as column vectors of m prices; the number of time periods $\tau \leq m-1$, $\tau \in \mathbb{N}$; the cost rates θ associated with the risk of assets, and the fixed costs ζ in Eq. (2.5).

Construct p from Algorithm 4.
function g = linots(data,t)
  Set T as the floor price of t
  if $t = T$ **then**
    return $g = data(t+1,:)$
  else
    return $g = data(T+1,:) + (data(T+2,:) - data(T+1,:))(t-T)$
  end if
end function
Set $f_p = @(t) linots(p,t)'$
Set $f_X = @(t) linots(X(\tau+1:end,:),t)$

Output: The linear interpolation of the insurance prices and the close prices of n time series into time-varying functions, $f_p(t)$ and $f_X(t)$, respectively.

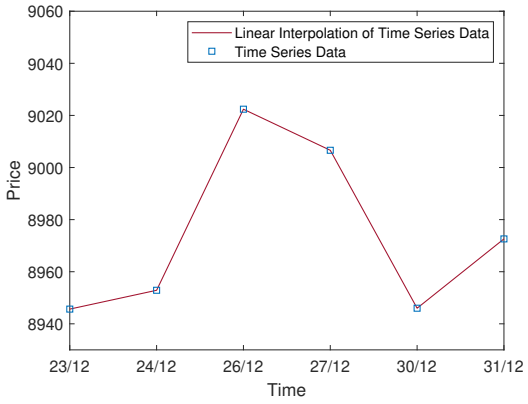


Figure 2: Linear Interpolation

4.1.2 Cubic Spline Interpolation

The cubic spline interpolation [26] is another possibility to create prices between two real-valued observation times. Algorithm 6 gives the corresponding computational procedure and Figure 3 shows an illustration of the outcome.

Algorithm 6 Algorithm for the cubic spline interpolation

Input: The marketed space $X = [x_1, x_2, \ldots, x_n]$, which is a matrix of n time series as column vectors of m prices; the number of time periods $\tau \leq m-1$, $\tau \in \mathbb{N}$; the cost rates θ associated with the risk of assets, and the fixed costs ζ in Eq. (2.5).

Construct p from Algorithm 4.
function g = sp(data)
  Set $data = data'$ and $[m,n] = size(data)$
  Set $del = data(:,2:end) - data(:,1:end-1)$
  Set $a = zeros(m,n)$
  Set $a(:,2:n-1) = 3(del(:,1:n-2) + del(:,2:n-1))$
  Set $a(:,1) = (5del(:,1) + del(:,2))/2$
  Set $a(:,n) = (del(:,n-2) + 5del(:,n-1))/2$
  Set $b = ones(n-2,1)$
  Set $c = spdiags([2,1,0; b, 4b, b; 0,1,2], [-1,0,1], n, n)$

return $d = (a/c)'$
end function
function g = splinots(data,d,t)
  Set T as the floor price of t
  if $t > size(data, 1) - 1$ **then**
    $g = data(end, :)$
  else if $t = T$ **then**
    $g = data(t + 1, :)$
  else
    Set $del = data(T + 2, :) - data(T + 1, :)$
    Set $dzz = del - d(T + 1, :)$ and $dzx = d(T + 2, :) - del$
    $g = (dzx - dzz)(t - T)^3 + (2dzz - dzx)(t - T)^2 + d(T + 1, :)(t - T) + data(T + 1, :)$
  end if
  return g
end function
Set $d = sp(p)$
Set $f_p = @(t)splinots(p, d, t)'$
Set $dd = sp(X(\tau + 1 : end, :))$
Set $f_X = @(t)splinots(X(\tau + 1 : end, :), dd, t)$

Output: The cubic spline interpolation of the insurance prices and the close prices of n time series into time-varying functions, $f_p(t)$ and $f_X(t)$, respectively.

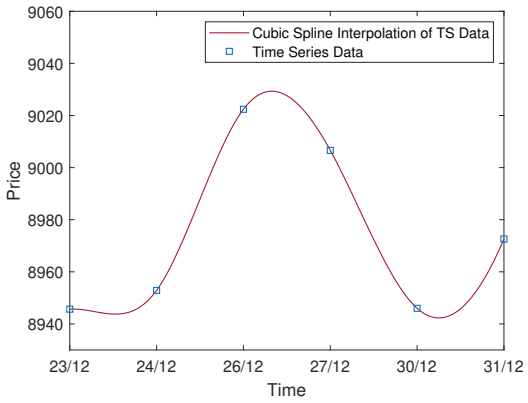


Figure 3: Cubic Spline Interpolation

4.1.3 Piecewise Cubic Hermite Interpolation

Lastly, the piecewise cubic Hermite interpolation, see [27], is another popular way to create prices between two real-valued observation times. Algorithm 7 gives the corresponding computational procedure and Figure 4 shows an illustration of the outcome.

Algorithm 7 Algorithm for the piecewise cubic Hermite interpolation

Input: The marketed space $X = [x_1, x_2, \ldots, x_n]$, which is a matrix of n time series as column vectors of m prices; the number of time periods $\tau \leq m - 1$, $\tau \in \mathbb{N}$; the cost rates θ associated with the risk of assets, and the fixed costs ζ of Eq. (2.5).

Construct p from Algorithm 4.
function g = pchinots(data,t)
  Set T as the floor price of t.
  if $t = T$ **then**

return $g = data(t+1,:)$
else
Set $u = size(data, 2)$
if $t > length(data) - 2$ **then**
Set $del = data(T+1:T+2,:) - data(T:T+1,:)$
Set $d = zeros(3, m)$
else if $t < 1$ **then**
Set $del = data(T+2:T+3,:) - data(T+1:T+2,:)$.
Set $d = zeros(2, m)$
else
Set $del = data(T+1:T+3,:) - data(T:T+2,:)$
Set $d = zeros(3, m)$
end if
$[k1, k2] = find(sign(del(1:end-1,:)).sign(del(end-1:end,:)) > 0)$
for $i = 1 : k1$ **do**
$d(k1(i)+1, k2(i)) = 2(\min(\|del(k1(i), k2(i))\|, \|del(k1(i)+1, k2(i))\|).\max(\|del(k1(i), k2(i))\|, \|del(k1(i)+1, k2(i))\|)./(del(k1(i), k2(i)) + del(k1(i)+1, k2(i)))$
end for
if $t < 1$ **then**
Set $d(1,:) = (3del(1,:) - del(2,:))/2$
for $i = 1 : u$ **do**
if $sign(d(1,i)) \neq sign(del(1,i))$ **then**
Set $d(1,i) = 0$
else if $sign(del(1,i)) \neq sign(del(2,i))$ **and** $\|d(1,i)\| > \|3del(1,i)\|$ **then**
Set $d(1,i) = 3del(1,i)$
end if
end for
end if
if $t > length(data) - 2$ **then**
Set $d(3,:) = (3del(2,:) - del(1,:))/2$
for $i = 1 : u$ **do**
if $sign(d(3,i)) \neq sign(del(2,i))$ **then**
Set $d(3,i) = 0$
else if $sign(del(2,i)) \neq sign(del(1,i))$ **and** $\|d(3,i)\| > \|3del(2,i)\|$ **then**
Set $d(3,i) = 3del(2,i)$
end if
end for
end if
Set $dzx = del(2,:) - d(end-1,:)$ and $dxz = d(end,:) - del(2,:)$
return $g = (dxz - dzx)(t-T)^3 + (2dzx - dxz)(t-T)^2 + d(end-1,:)(t-T) + data(T+1,:)$
end if
end function
Set $f_p = @(t)pchinots(p,t)'$
Set $f_X = @(t)pchinots(X(\tau+1:end,:),t)$

Output: The piecewise cubic Hermite interpolation of the insurance prices and the close prices of n time series into time-varying functions, $f_p(t)$ and $f_X(t)$, respectively.

We developed four MATLAB functions to precisely implement Algorithms 5-7 and used them in numerical experiments. Namely, function `linots` implements Algorithm 5, functions `spl` and `splinots` implements Algorithm 6 and function `pchinots` implements Algorithm 7.

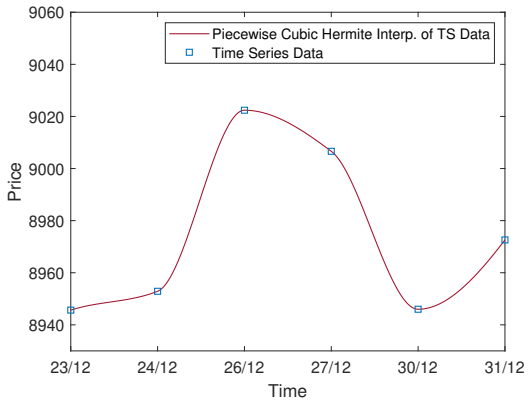


Figure 4: Piecewise Cubic Hermite Interpolation

4.2 Periods and Data Observations Handling

The time periods in finance can be divided into daily, weekly, monthly, quarterly, annual and their combinations. But their findings may not be equal in number for the same division of two time periods, which is due to the fact that financial markets may be closed (special calendar days), the year may be leap, one month may have fewer days etc. To solve the problem of missing observations for corresponding division periods, we calculate the parameter ω for each t within the Algorithm 9. This procedure divides the observations into proper time periods. For that reason, we designed Algorithm 8:

Algorithm 8 Algorithm for splitting the observations

Input: The *data*, which is a time-series as a vector of n prices, the time t and the vector *noep*, which contains the number of observations in each period.

function ω = omega(*noep*,t)
  Set T as the floor price of t
  if $T > 0$ **then**
    Set $\omega = sum(noep(1:T)) - 1$
    if $t \neq T$ **then**
      Set $\omega = (\omega + noep(T+1) \cdot (t - T))/t$
    else
      Set $\omega = \omega/t$
    end if
  else
    $\omega = noep(1) - 1$
  end if
  return ω
end function

Output: The parameter ω which splits the observations to the time periods.

5 The TV-BAS Algorithmic Procedure for Solving the TV-MCPITC Problem

First, we present a complementary algorithm which constructs the variables of the TV-NLP problem described in Subsection 2.3 combined with Algorithm 8.

Algorithm 9 Algorithm for the TV-NLP problem of subsection 2.3.

Input: The interpolated marketed space and insurance prices f_X and f_p, respectively; the time t and the vector *noep*, which contains the number of observations in each period; the floor ϕ, the previous portfolio $\eta_{-1} = \eta(t-1)$, and the previous insurance and transaction costs y_{-1}.

1: **function** $[\eta^-,\eta^+,A,b,p]$ = problem($noep$,t,f_p,f_X,ϕ,η_{-1},y_{-1})
2: Set $\omega = omega(noep, t)$ and floor $= \phi(\omega t)$
3: Set $p = f_p(\omega t)$ and $A = -f_X(\omega t)$
4: Set payoff $= A\eta_{-1} + y_{-1}$
5: Set $b = \min(\text{payoff}, -\text{floor})$
6: Set $\eta^- = zeros(length(p), 1)$
7: Set $\eta^+ = \text{payoff}./A'$
8: **end function**

Output: The η^-,η^+,A,b,p for the time t.

The main algorithm for solving the TV-MCPITC problem is the following algorithm 10 which also includes the TV-BAS MATLAB function.

Algorithm 10 The TV-BAS Algorithmic Procedure for Solving the TV-MCPITC Problem.

Input: The *data*, which is a time-series as a vector of n prices; the moving average's number of time periods $\tau \leq m-1$, $\tau \in \mathbb{N}$; the time interval $tspan = [t_{start}, t_{end}]$, where $t_{start}, t_{end} \in [0, length(data) - 1]$; the parameter γ which divides the $tspan$ in γ equal steps, and the vector *noep*, which contains the number of observations in each period. Furthermore, a given portfolio η_p, the penalty parameter R, the fixed costs β^-, β^+ and the costs rates α^-, α^+. Lastly, the requirements of Algorithm 4.

1: Construct p from Algorithm 4.
2: Construct f_p and f_X from Algorithm 5, 6 or 7.
3: **function** $[x,y]$ = TVBAS(η_p,γ,$tspan$,$noep$,f_p,f_X,ϕ,R, d,δ,β^-, β^+,α^-, α^+)
4: Set $t = tspan(1) : \frac{1}{\gamma} : tspan(2)$
5: Set $n = length(\eta_p)$ and $tot = length(t)$
6: Set $x = zeros(n, tot)$ and $y = zeros(1, tot)$
7: Set $[\eta^-, \eta^+, A, b, p] = problem(noep, t(1), f_p, f_X, \phi, \eta_p, 0)$
8: Set $[x(:,1), y(1)] = basfunc(\eta_p, p, A, b, R, \eta^-, \eta^+, \beta^-, \beta^+, \alpha^-, \alpha^+, 0.9, 0.8, 1e-6, 1200)$
9: **for** $i = 2 : tot$ **do**
10: Set $[\eta^-, \eta^+, A, b, p] = problem(noep, t(i), f_p, f_X, \phi, x(:,i-1), y(i-1))$
11: Set $[x(:,i), y(i)] = basfunc(x(:,i-1), p, A, b, R, \eta^-, \eta^+, \beta^-, \beta^+, \alpha^-, \alpha^+, 0.9, 0.8, 1e-6, 1200)$
12: **end for**
13: **end function**

Output: The online solution of the TV-MCPITC Problem.

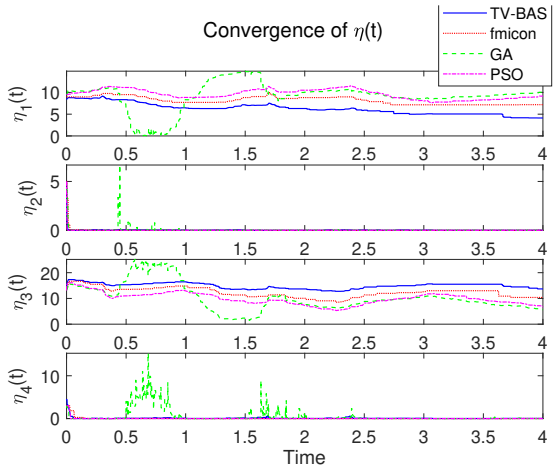
6 Numerical Examples

In the following examples, we set $\beta_i^+ = \beta_i^- = 0.1$, $\alpha_i^+ = 0.06$ and $\alpha_i^- = 0.04$. Hence, the function $\kappa_i(t)$ is defined as follows:

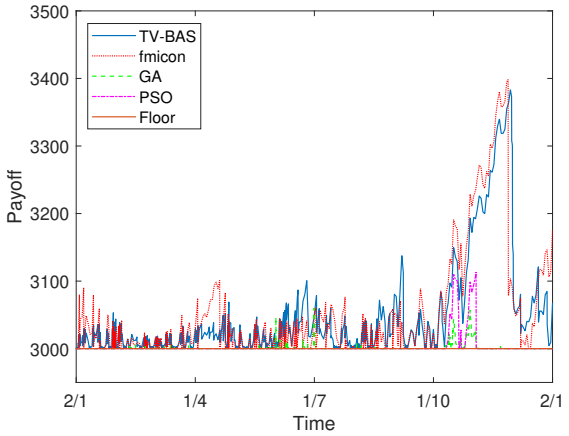
$$\kappa_i(t) = \begin{cases} 0, & \eta_i(t) = \eta_i(t-1) \\ 0.1 + 0.06(\eta_i(t) - \eta_i(t-1))x_i(t), & \eta_i(t) > \eta_i(t-1) \\ 0.1 + 0.04(\eta_i(t-1) - \eta_i(t))x_i(t), & \eta_i(t) < \eta_i(t-1). \end{cases}$$

6.1 Numerical Example A

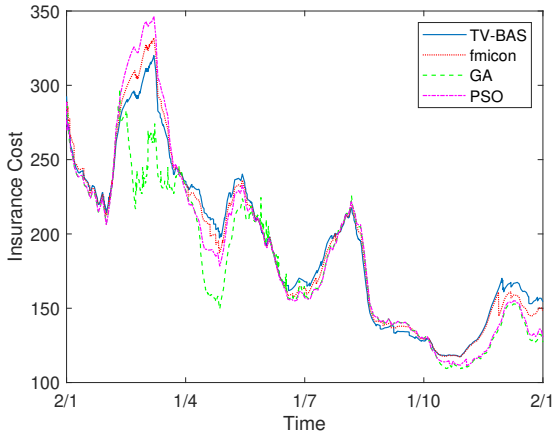
Table 1 includes the ticker symbols of the stocks that we use in the portfolio of this example. A ticker symbol, or a stock symbol, is an arrangement of characters which is used for identification of a Bond, Stock,



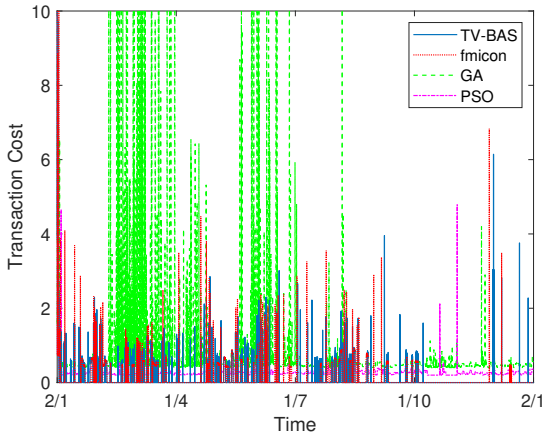
(a) Convergence of Portfolios.



(b) Payoff of Portfolios.



(c) Insurance Costs of Portfolios.



(d) Transaction Costs of Portfolios.

Figure 5: The convergence, the payoff, the insurance costs and the transaction costs for a portfolio consisting of 4 stocks, in numerical example A.

Mutual Fund, Exchange traded fund (ETF) or any other security traded on the stock exchange. Every listed security in financial markets has a unique ticker symbol.

Market			
FB	INTC	MSFT	VZ

Table 1: Market vector stocks

In this example, we find the quarterly optimal portfolio for a period of one year. Let $X = [x_1, x_2, x_3, x_4]$, where X comprises the daily close prices of Table 1 from 18/10/2018 to 2/1/2020 into x_1, x_2, x_3, x_4, respectively. For the aforementioned time series, we use the first 50 prices of the observations to calculate the insurance prices p of Algorithm 4. Consequently, we set $\tau = 50$, $\zeta = 5$ and $\theta = 3e3$ in Algorithm 4. The data are used from the period from 2/1/2019 to 2/1/2020 with 253 observations. That time interval is divided into quarters $Q1, Q2, Q3$ and $Q4$, which denote the first, second, third and fourth quarter of a year, respectively. It is known that every quarter comprises of three months. In particular, $Q1$ of 2019 has 61 observations, $Q2$ has 63, $Q3$ has 64 and $Q4$ along with the one observation from the year 2020 has 65 observations. So, we have

$noep = [61, 63, 64, 65]$ at $tspan = [0\ 4]$. From Algorithm 8 it follows that

$$\omega = \begin{cases} 61 - 1, & t \in [0, 1) \\ (61 \cdot 1 - 1 + 63 \cdot (t-1))/t, & t \in [1, 2) \\ (61 \cdot 1 + 63 \cdot 1 - 1 + 64 \cdot (t-2))/t, & t \in [2, 3) \\ (61 \cdot 1 + 63 \cdot 1 + 64 \cdot 1 - 1 + 65 \cdot (t-3))/t, & t \in [3, 4] \end{cases}$$

where the 253 observations have been divided in terms of the quarter which they belong. Also, we use the linear data interpolation in order to convert p and X into functions of time $f_p(t)$ and $f_X(t)$, respectively, through Algorithm 6. Given a portfolio $\eta_p = [2, 5, 7, 3]^{\mathrm{T}}$; we set $\gamma = 1000$, $R = 1e5$ and the constant floor $\phi = 3000$.

We present the results in Figures 5a-5d where:

- Figure 5a shows the outcome $\eta(t)$ of TV-BAS and the outcomes of `fmincon`, `GA` and `particleswarm` MATLAB functions;
- Figure 5b shows the payoff of the portfolio $\eta(t)$, which is $f_X(\omega t)\eta(t)$, compared with the outcomes of `fmincon`, `GA` and `particleswarm` MATLAB functions;
- Figure 5c shows the insurance costs of the portfolio $\eta(t)$ compared with the outcomes of `fmincon`, `GA` and `particleswarm` MATLAB functions;
- Figure 5d shows the transaction costs of the portfolio $\eta(t)$ compared with the outcomes of `fmincon`, `GA` and `particleswarm` MATLAB functions.

The results that are depicted in Figure 5a show that the TV-BAS solves the TV-MCPITC problem and produces its online solution $\eta(t)$. The solution of the TV-BAS is similar to the solution of the MATLAB functions `fmincon`, `GA` and `particleswarm`. The insurance costs of the portfolio $\eta(t)$ are shown in Figure 5c and its payoff is shown in Figure 5b. The transaction costs are shown in Figure 5d, where it is observable that the TV-BAS solution requires on average the least transaction costs compared to the solutions generated by `fmincon`, `GA` and `particleswarm`. It is also observable that the transaction costs of the portfolio increase when the insurance costs of the portfolio reduce. The time consumption of this numerical example is presented in Table 4, and shows that the TV-BAS is on average much faster compared with the `fmincon` MATLAB function. The `fmincon` function is faster than `GA` and `particleswarm` MATLAB functions and the `GA` is the slowest. Overall, the TV-BAS worked excellently in solving the TV-MCPITC problem.

6.2 Numerical Example B

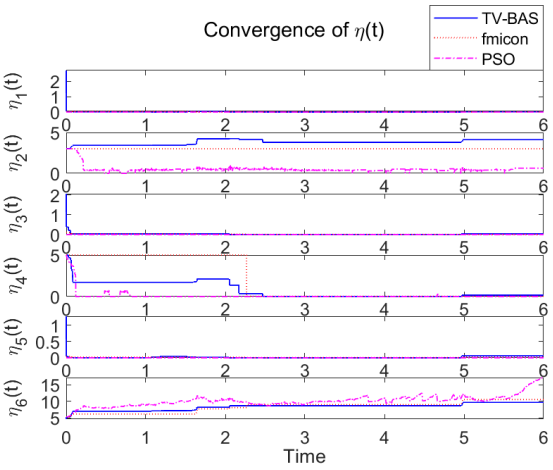
Table 2 includes the ticker symbols of the stocks that we use in our portfolio.

Market					
AMD	C	MRVL	MS	MU	XOM

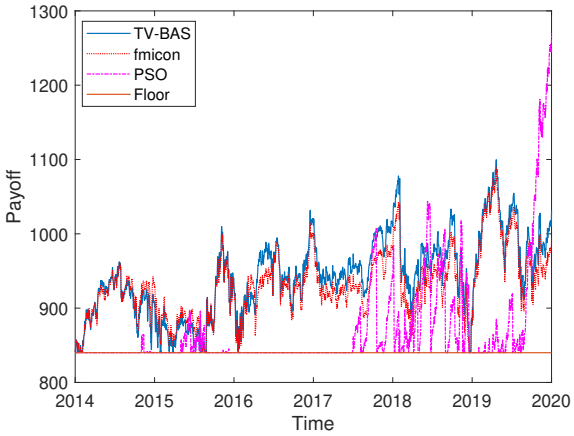
Table 2: Market vector stocks

In this example, we find the yearly optimal portfolio for a period of six years. Let $X = [x_1, x_2, x_3, x_4, x_5, x_6]$, where X comprises the daily close prices of Market stocks of Table 2 from 19/3/2013 to 2/1/2020 into $x_1, \ldots, x_6$, respectively. For the aforementioned time series, we use the first 200 prices of the observations to calculate the insurance prices p of Algorithm 4. Consequently, we set $\tau = 200$, $\zeta = 2$ and $\theta = 1e2$ in Algorithm 4. The rest of our data is the period from 2/1/2014 to 2/1/2020 with 1511 observations. In particular, the years 2014, 2015, 2016 have 252 observations each, the years 2017, 2018 have 251 observations each and 2019, 2020 have 253 observation together. So, we have $noep = [252, 252, 252, 251, 251, 253]$ inside $tspan = [0\ 6]$. From Algorithm 8 it follows that

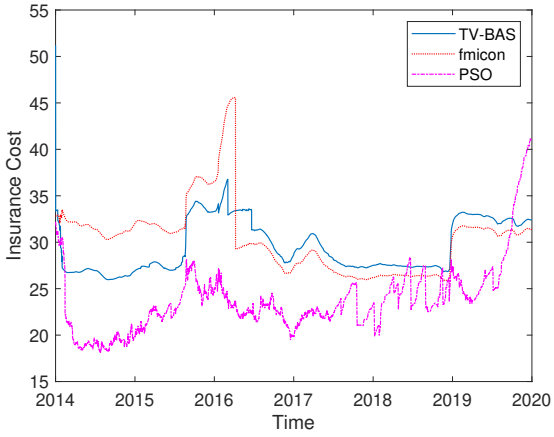
$$\omega = \begin{cases} 252 - 1, & t \in [0, 3) \\ (252 \cdot 3 - 1 + 251 \cdot (t-3))/t, & t \in [3, 5) \\ (252 \cdot 3 + 251 \cdot 2 - 1 + 253 \cdot (t-5))/t, & t \in [5, 6] \end{cases}$$



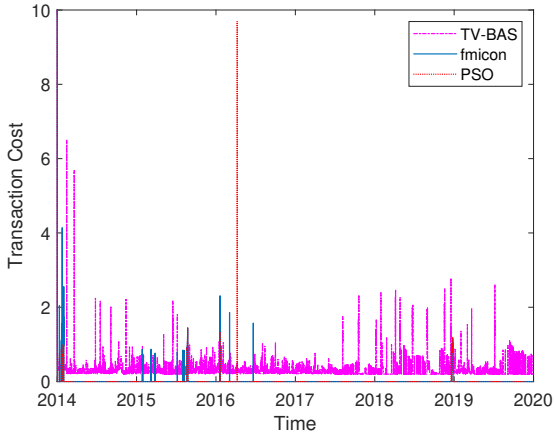
(a) Convergence of Portfolios.



(b) Payoff of Portfolios.



(c) Insurance Costs of Portfolios.



(d) Transaction Costs of Portfolios.

Figure 6: The convergence, the payoff, the insurance costs and the transaction costs for a portfolio consisting of 6 stocks, in numerical example B.

where the 1511 observations are divided in terms of the year which they belong. Also, we use the piecewise cubic Hermite data interpolation in order to convert p and X into the functions of time $f_p(t)$ and $f_X(t)$, respectively, through Algorithm 7. Given a portfolio $\eta_p = [7, 3, 4, 5, 5, 3]^{\mathrm{T}}$, we set $\gamma = 1000$, $R = 1e5$ and a constant floor $\phi = 840$.

We present the results in Figures 6a-6d where:

- Figure 6a shows the outcome $\eta(t)$ of TV-BAS and the outcomes of `fmincon` and `particleswarm` MATLAB functions;
- Figure 6b shows the payoff of the portfolio $\eta(t)$, which is $f_X(\omega t)\eta(t)$, compared with the outcomes of `fmincon` and `particleswarm` MATLAB functions;
- Figure 6c shows the variance of the portfolio $\eta(t)$ compared with the outcomes of `fmincon` and `particleswarm` MATLAB functions;
- Figure 6d shows the transaction costs of the portfolio $\eta(t)$ compared with the outcomes of `fmincon` and `particleswarm` MATLAB functions.

The results that are depicted in Figure 6a show that the TV-BAS solves the TV-MCPITC problem and produces its online solution $\eta(t)$. The solution of the TV-BAS is similar to the solution of the MATLAB functions `fmincon` and `particleswarm`. The insurance costs of the portfolio $\eta(t)$ are shown in Figure 6c and its payoff is shown in Figure 6b. It can be observed that increase of the insurance costs of the portfolio initiates increase of the portfolio's payoff. The transaction costs are shown in Figure 6d, wherein it is observable that the TV-BAS solution requires on average the least transaction costs compared with the solutions of `fmincon` and `particleswarm`. We also observe that when the transaction costs of the portfolio increase, the portfolio's insurance costs reduce. The time consumption recorded in this example is presented in Table 4. The results arranged in Table 4 show that the TV-BAS is on average faster than the `fmincon` and `particleswarm` MATLAB functions and the `particleswarm` is the slowest. Note that we do not compare the `GA` MATLAB function because of its extremely slow execution time. Overall, the TV-BAS worked excellently in solving the TV-MCPITC problem.

6.3 Numerical Example C

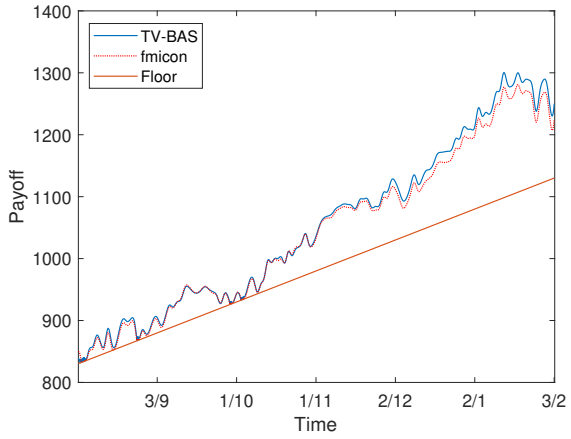
In this example, we shall verify the effectiveness of the proposed Algorithm 10 for solving the TV-MCPITC problem in three different groups of portfolios with bigger size, consisting of 20 stocks, 40 stocks and 60 stocks. Table 3 includes the stocks used in our portfolios. In this example, we find the monthly optimal portfolio for a period of six months. Each table is divided into three groups and each group comprises from twenty ticker symbols. The experimental results obtained in this section confirm the reliability of Algorithm 10 on the real-world dataset and demonstrate its efficacy in practical scenarios, even for large data sets.

Group A		**Group B**		**Group C**	
AAL	CMCSA	HPQ	MS	PFE	TEVA
AAPL	CSCO	INFY	MSFT	S	TSLA
ABEV	CZR	INTC	MU	SAN	TWTR
AMD	ET	ITUB	FB	SCHW	VALE
AUY	F	JPM	NLY	SIRI	VZ
BABA	FCX	KGC	NOK	SLB	WMB
BAC	GE	KMI	NTNX	SMFG	WPX
BBD	GILD	KO	NVDA	SNAP	XOM
C	GOLD	M	OXY	SQ	DAL
CCL	HBAN	BMY	PBR	T	DIS

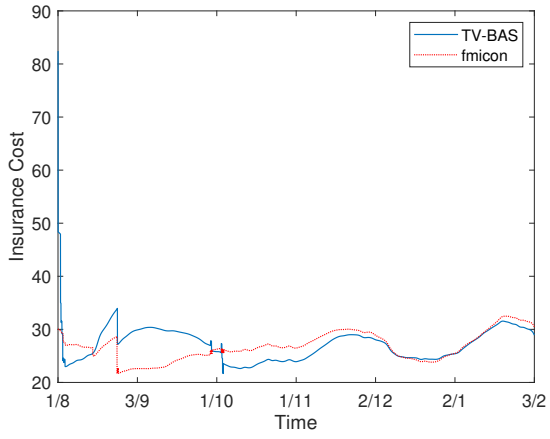
Table 3: Market vector stocks

6.3.1 Portfolio of 20 stocks

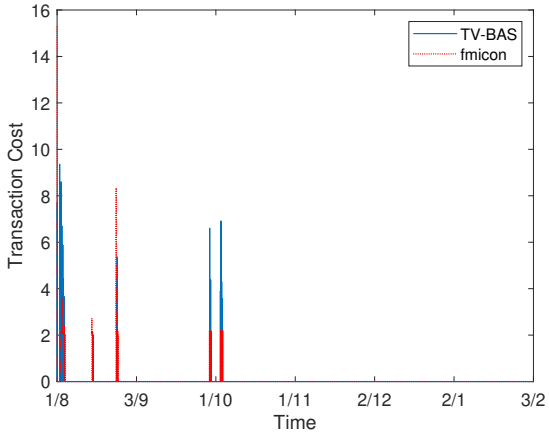
Let $X = [x_1, \ldots, x_{20}]$, where X comprises the daily close prices of Group A stocks of Table 3 from 5/6/2019 to 3/2/2020 into $x_1, \cdots, x_{20}$, respectively. We use the first 40 prices from the aforementioned time series to calculate the insurance prices p of Algorithm 4. Consequently, we set $\tau = 40$, $\zeta = 2$ and $\theta = 1e3$ in Algorithm 4. The rest of our data is the period from 1/8/2019 to 3/2/2020 with 128 observations. In particular, August comprises 22 observations, September and November comprise 20 observations each, October 21 and December with January have 22 observations together. So, $noep = [22, 20, 23, 20, 21, 22]$ at $tspan = [0\ 6]$.



(a) Payoff of Portfolios.



(b) Insurance Costs of Portfolios.



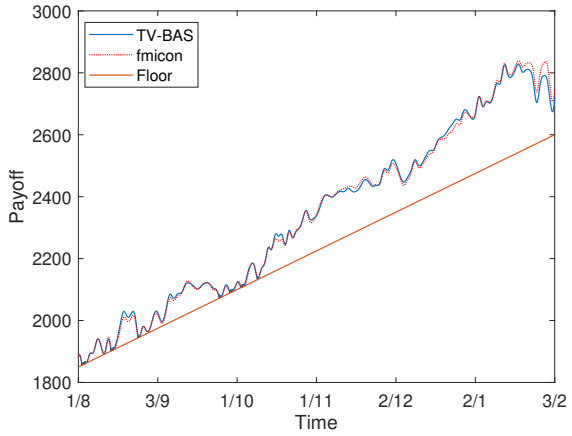
(c) Transaction Costs of Portfolios.

Figure 7: The payoff, the insurance costs and the transaction costs for a portfolio consisting of 20 stocks, in numerical example C.

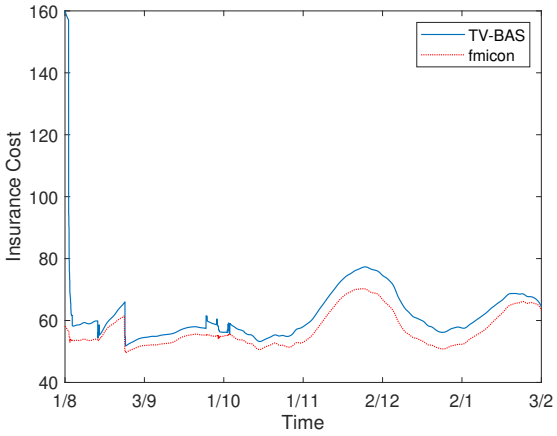
From Algorithm 8 it follows that

$$\omega = \begin{cases} 22-1, & t \in [0,1) \\ (22{\cdot}1-1+20\cdot(t-1))/t, & t \in [1,2) \\ (22{\cdot}1+20\cdot 1-1+23\cdot(t-2))/t, & t \in [2,3) \\ (22{\cdot}1+20\cdot 1+23\cdot 1-1+20\cdot(t-3))/t, & t \in [3,4) \\ (22{\cdot}1+20\cdot 2+23\cdot 1-1+21\cdot(t-4))/t, & t \in [4,5) \\ (85+21{\cdot}1-1+22\cdot(t-5))/t, & t \in [5,6] \end{cases}$$

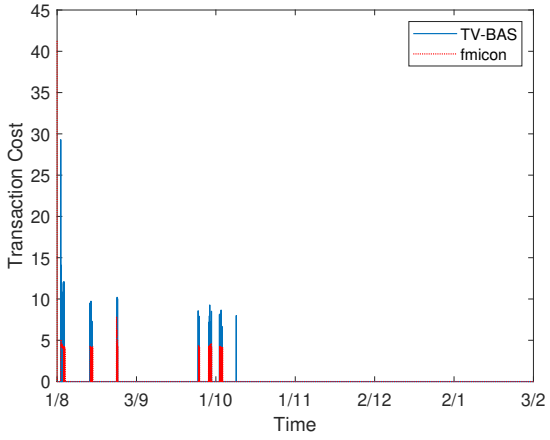
where the 128 observations have been divided in terms of the month which they belong. Also, we use cubic spline data interpolation in order to convert p and X into time-varying functions $f_p(t)$ and $f_X(t)$, respectively, through Algorithm 6. Given a portfolio $\eta_p = ones(20,1)$; we set $\gamma = 1000$, $R = 1e8$ and a floor $\phi = @(t)830 + 50t$. The results are as shown in Figure 7 and the execution time in Table 4.



(a) Payoff of Portfolios.



(b) Insurance Costs of Portfolios.



(c) Transaction Costs of Portfolios.

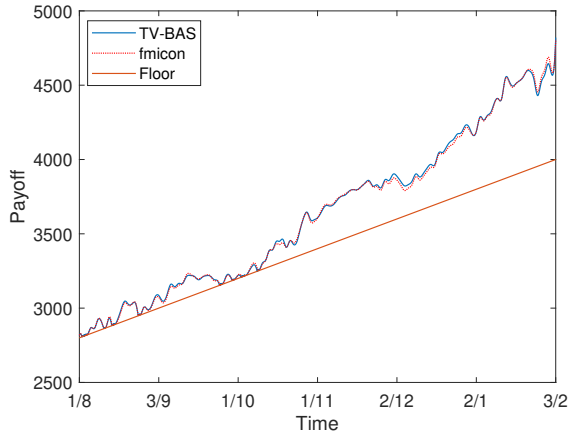
Figure 8: The payoff, the insurance costs and the transaction costs for a portfolio consisting of 40 stocks, in numerical example C.

6.3.2 Portfolio of 40 stocks

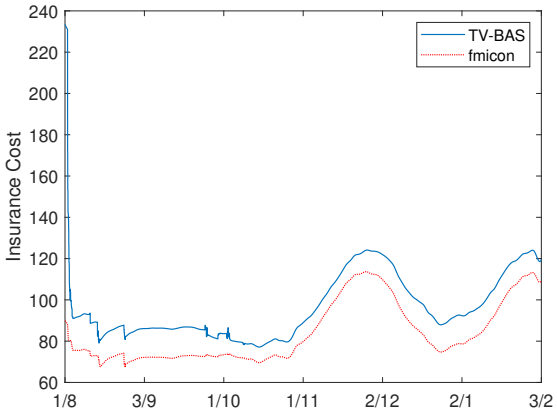
In this example we take numerical example 6.3.1 exactly as it is and we just add another twenty stocks in our portfolio. So, we set $X = [x_1, \ldots, x_{40}]$, where X comprises the daily close prices of Group A and Group B stocks of Table 3 from 1/8/2019 to 3/2/2020. Given a portfolio $\eta_p = ones(40, 1)$; we set $\gamma = 1000$, $R = 1e8$ and the floor $\phi = @(t)1850 + 125t$. The results are as shown in Figure 8 and the execution time is arranged in Table 4.

6.3.3 Portfolio of 60 stocks

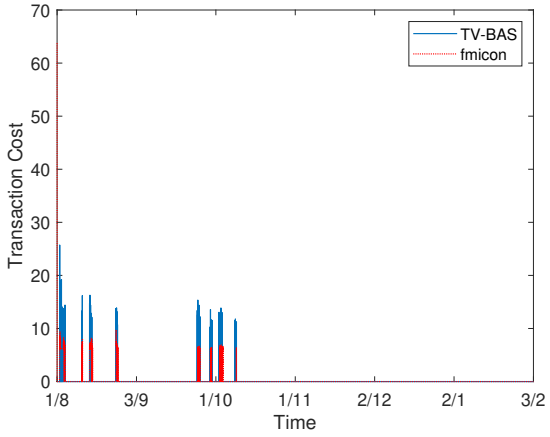
In this example we take numerical example 6.3.2 exactly as it is and we just add another twenty stocks in our portfolio. So, we set $X = [x_1, \ldots, x_{60}]$, where X comprises the daily close prices of groups A, B and C stocks of Table 3 from 1/8/2019 to 3/2/2020. Given a portfolio $\eta_p = ones(60, 1)$, we set $\gamma = 1000$, $R = 1e8$ and the floor $\phi = @(t)2800 + 200t$. The results are as shown in Figure 9 and the execution time in Table 4.



(a) Payoff of Portfolios.



(b) Insurance Costs of Portfolios.



(c) Transaction Costs of Portfolios.

Figure 9: The payoff, the insurance costs and the transaction costs for a portfolio consisting of 60 stocks, in numerical example C.

6.3.4 Comparative Results and Discussion

The results from the numerical examples in subsections 6.3.1 - 6.3.3 can be summarized as follows:

- Table 4 shows the average execution time of TV-BAS and `fmincon` in numerical examples 6.3.1-6.3.3, by using linear, cubic spline and piecewise cubic Hermite data interpolation;
- Figures 7a, 8a and 9a show the payoff of the portfolios consisting of 20, 40 and 60 stocks, respectively, between the outcome of TV-BAS and the outcome of `fmincon`;
- Figures 7b, 8b and 9b show the insurance costs of the portfolio $\eta(t)$ compared with the outcome of `fmincon`;
- Figures 7c, 8c and 9c show the transaction costs of the portfolio $\eta(t)$ compared with the outcome of `fmincon` MATLAB functions.

The solutions $\eta(t)$ of the TV-BAS are similar to the solutions of the MATLAB function `fmincon`. The insurance costs of the portfolios $\eta(t)$ are shown in Figures 7b, 8b and 9b. Also, their payoff values $f_X(\omega t)\eta(t)$ are shown in Figures 7a, 8a and 9a for a portfolio consisting of 20, 40 and 60 stocks, respectively. The transaction

costs are shown in Figures 7c, 8c and 9c for a portfolio consisting of 20, 40 and 60 stocks, respectively. It is observable that the TV-BAS solutions include on average about the same transaction costs as the solutions of `fmincon`. We also observe that when the transaction costs of the portfolio increase, the insurance costs of the portfolio reduce. The time consumptions of numerical example C are presented in Table 4 and show that the TV-BAS is on average much faster than the `fmincon` MATLAB function. Furthermore, in order to make our approach more realistic, we consider the parameter ω which is very helpful in the case where we want to combine different time periods with a different number of observations in each one of them. Also, we do not compare the `GA` and `particleswarm` MATLAB functions because of their slow execution time and the constraints violation in the TV-MCPITC problem which is caused by the portfolio's large size. Overall, the TV-BAS worked properly in solving the TV-MCPITC problem.

6.4 Time Comparison of TV-BAS, fmincon, GA and PSO

Example	**Interpolation**	**TV-BAS**	**fmincon**	**GA**	**PSO**
6.1	Linear	$55.6s$	$81s$	$2522s$	$94s$
	Cubic Spline	$57.2s$	$84.3s$	$2480s$	$92.8s$
	P.C.Hermite	$57.3s$	$81.7s$	$2569s$	$95.2s$
6.2	Linear	$83.8s$	$85.8s$	–	$148.4s$
	Cubic Spline	$84.3s$	$87.9s$	–	$151.3s$
	P.C.Hermite	$85.2s$	$89s$	–	$156s$
6.3.1	Linear	$90.4s$	$131.6s$	–	–
	Cubic Spline	$91.1s$	$131.4s$	–	–
	P.C.Hermite	$89.5s$	$127.2s$	–	–
6.3.2	Linear	$105.5s$	$132s$	–	–
	Cubic Spline	$107.5s$	$134s$	–	–
	P.C.Hermite	$107s$	$134.7s$	–	–
6.3.3	Linear	$113.6s$	$132.7s$	–	–
	Cubic Spline	$114.7s$	$137.2s$	–	–
	P.C.Hermite	$114.1s$	$137.1s$	–	–

Table 4: Examples 6.1, 6.2 and 6.3 execution time

We compare the performance of TV-BAS with the proposed MATLAB functions of Subsection 4.1 against the `fmincon`, `GA` and `particleswarm` MATLAB functions. Corresponding numerical values are presented in Table 4, which shows the average execution time of numerical examples 6.1, 6.2 and 6.3 by using linear, cubic spline and P.C. Hermite (Piecewise Cubic Hermite) interpolation functions.

We also monitor the performance of TV-BAS with the corresponding MATLAB function (namely `interp1`) in Table 5. That is, we set $f_p = @(t)interp1(p, t+1, 'method')'$ and $f_X = @(t)interp1(X(\tau+1 : end, :), t+1, 'method')$ which produce the interpolation of the insurance price and the close prices, respectively. The steps created by the parameter γ in the Table 5 are the same with the corresponding cases of Table 4. Hence, the tables 4 and 5 are suitable to be compared with each other.

All numerical experiments are performed using the MATLAB R2018b environment on an Intel® Core™ i5-6600K CPU 3.50 GHz, 16 GB RAM, running on Windows 10 64 bit Operating System.

The conclusion arising from Table 4 is that the linear, the cubic spline and the piecewise cubic Hermite interpolation have the same level of efficiency when the proposed MATLAB functions of Subsection 4.1 are used. On the contrary, the conclusion arising from Table 5 is that the piecewise cubic Hermite interpolation does not have the same level of efficiency as the linear and the cubic spline interpolation when the MATLAB function `interp1` is used.

Example	**Interpolation**	**TV-BAS**	**fmincon**
6.1	'linear'	58.4s	88.7s
	'spline'	60.5s	89s
	'pchip'	60.1s	87s
6.2	'linear'	88.7s	89.8s
	'spline'	91.8s	94.2s
	'pchip'	92.4s	95.5s
6.3.1	'linear'	97.9s	141.1s
	'spline'	100.5s	143.4s
	'pchip'	106.5s	148.4s
6.3.2	'linear'	107.1s	155.3s
	'spline'	108.6s	155.2s
	'pchip'	121.5s	150s
6.3.3	'linear'	115.7s	135.4s
	'spline'	118.2s	139.8s
	'pchip'	131.1s	153.1s

Table 5: Examples 6.1, 6.2 and 6.3 execution time

The general conclusion arising from tables 4 and 5 is that when we apply the proposed MATLAB functions from subsection 4.1, the TV-BAS, `fmincon`, `GA` and `particleswarm` produce faster results than the function `interp1`, and with the same level of efficiency among the linear and the cubic spline interpolation. Note that as the value of the γ parameter increases, the difference in time consumption between the proposed MATLAB functions of Subsection 4.1 and the MATLAB function `interp1` becomes significant.

7 Conclusion

This paper introduces the TV-MCPITC problem and presents its online solution. The efficiency of the TV-BAS algorithm in time-varying financial NLP problems has been demonstrated by a number of numerical examples. Conforming to our numerical simulations, we deduced that the TV-BAS approach provides online solutions of a time-varying version of the minimum-cost portfolio insurance under transaction costs problem. It is also a highly competitive, or even better alternative to the `fmincon`, `GA` and `particleswarm` MATLAB functions. Nonetheless, as the value of the γ parameter increases, the performance of the TV-BAS algorithm

improves. Experimental results show the reliability of the TV-BAS algorithm on the real-world datasets in different and reasonable portfolios setup, and demonstrate its efficacy in practical scenarios, even for large sets of data. The online solution of a time-varying financial problem is a great technical analysis tool as well as an important financial analysis tool. A possible future research direction should be to apply similar techniques to other financial problems, such as asset allocation, risk management, option pricing, model calibration, etc., which first require proper modification of them as realistic time-varying models, and then to define the appropriate time-varying method for their solution.

References

[1] S. Maier, J. W. Polak, D. M. Gann, Valuing portfolios of interdependent real options using influence diagrams and simulation-and-regression: A multi-stage stochastic integer programming approach, Computers & Operations Research (2018) 104505.

[2] H.-Q. Li, Z.-H. Yi, Portfolio selection with coherent Investors expectations under uncertainty, Expert Systems with Applications 133 (2019) 49–58.

[3] C. E. Kountzakis, On efficient portfolio selection using convex risk measures, Mathematics and Financial Economics 4 (3) (2011) 223–252.

[4] J.-S. Chen, C.-L. Chang, J.-L. Hou, Y.-T. Lin, Dynamic proportion portfolio insurance using genetic programming with principal component analysis, Expert systems with applications 35 (1-2) (2008) 273–278.

[5] A. H. Khan, X. Cao, V. N. Katsikis, P. Stanimirovic, I. Brajevic, S. Li, S. Kadry, Y. Nam, Optimal Portfolio Management for Engineering Problems Using Nonconvex Cardinality Constraint: A Computing Perspective, IEEE Access (2020) 1–1.

[6] A. H. Khan, X. Cao, S. Li, C. Luo, Using Social Behavior of Beetles to Establish a Computational Model for Operational Management, IEEE Transactions on Computational Social Systems (2020) 1–11.

[7] J. Zhang, Y. Huang, G. Ma, B. Nener, Multi-objective beetle antennae search algorithm, arXiv preprint arXiv:2002.10090.

[8] S. Xie, X. Chu, M. Zheng, C. Liu, Ship predictive collision avoidance method based on an improved beetle antennae search algorithm, Ocean Engineering 192 (2019) 106542.

[9] J. Wang, H. Chen, BSAS: Beetle swarm antennae search algorithm for optimization problems, arXiv preprint arXiv:1807.10470.

[10] C. D. Aliprantis, D. J. Brown, J. Werner, Minimum-cost portfolio insurance, Journal of Economic Dynamics & Control 24 (2000) 1703–1719.

[11] V. N. Katsikis, Matlab - Modelling, Programming and Simulations, IntechOpen, 2010, Ch. Computational and mathematical methods in portfolio insurance - A MATLAB-based approach.

[12] V. N. Katsikis, S. D. Mourtas, A heuristic process on the existence of positive bases with applications to minimum-cost portfolio insurance in C[a, b], Applied Mathematics and Computation 349 (2019) 221–244.

[13] V. N. Katsikis, Computational methods in lattice-subspaces of C[a,b] with applications in portfolio insurance, Applied Mathematics and Computation 200 (2008) 204–219.

[14] V. N. Katsikis, S. D. Mourtas, ORPIT: A Matlab Toolbox for Option Replication and Portfolio Insurance in Incomplete Markets, Computational Economics (2019) 1.
URL `http://dx.doi.org/10.1007/s10614-019-09936-5`

[15] V. N. Katsikis, Computational methods in portfolio insurance, Applied Mathematics and Computation 189 (1) (2007) 9–22.

[16] H. Maalej, J.-L. Prigent, On the stochastic dominance of portfolio insurance strategies, Journal of Mathematical Finance 6 (01) (2016) 14.

[17] J. George, W. Trainor, Portfolio insurance using leveraged ETFs, Available at SSRN 3055199.

[18] D. Pain, J. Rand, Recent developments in portfolio insurance, Recent Developments in Portfolio Insurance.

[19] M. S. Lobo, M. Fazel, S. Boyd, Portfolio optimization with linear and fixed transaction costs, Annals of Operations Research 152 (2007) 341–365.

[20] K. Deb, Optimization for Engineering Design: Algorithms and Examples, 2nd Edition, PHI, 2013.

[21] X. Jiang, S. Li, BAS: Beetle Antennae Search Algorithm for Optimization Problems, arXiv preprint abs/1710.10724.
URL http://arxiv.org/abs/1710.10724

[22] A. H. Khan, X. Cao, S. Li, V. N. Katsikis, L. Liao, BAS-ADAM: an ADAM based approach to improve the performance of beetle antennae search optimizer, IEEE/CAA Journal of Automatica Sinica 7 (2) (2020) 461–471.

[23] Z. Zhu, Z. Zhang, W. Man, X. Tong, J. Qiu, F. Li, A new beetle antennae search algorithm for multiobjective energy management in microgrid, in: Proc. 13th IEEE Conf. Industrial Electronics and Applications (ICIEA), 2018, pp. 1599–1603.

[24] Q. Wu, X. Shen, Y. Jin, Z. Chen, S. Li, A. H. Khan, D. Chen, Intelligent Beetle Antennae Search for UAV Sensing and Avoidance of Obstacles, Sensors 19 (2019) 1758.

[25] X. Xu, K. Deng, B. Shen, A beetle antennae search algorithm based on Lvy flights and adaptive strategy, Systems Science & Control Engineering 8 (2020) 35–47.

[26] C. De Boor, C. De Boor, E.-U. Mathématicien, C. De Boor, C. De Boor, A practical guide to splines, Vol. 27, springer-verlag New York, 1978.

[27] F. N. Fritsch, R. E. Carlson, Monotone piecewise cubic interpolation, SIAM Journal on Numerical Analysis 17 (2) (1980) 238–246.