Optimal online algorithms for the portfolio selection problem, bi-directional trading and -search with interrelated prices

Pascal Schroeder, Imed Kacem, Günter Schmidt

RAIRO-Oper. Res. 53 (2019) 559–576
https://doi.org/10.1051/ro/2018064

RAIRO Operations Research
www.rairo-ro.org

OPTIMAL ONLINE ALGORITHMS FOR THE PORTFOLIO SELECTION PROBLEM, BI-DIRECTIONAL TRADING AND -SEARCH WITH INTERRELATED PRICES

Pascal Schroeder[1,*], Imed Kacem[1] and Günter Schmidt[2]

Abstract. In this work we investigate the *portfolio selection problem* (P1) and *bi-directional trading* (P2) when prices are interrelated. Zhang *et al.* (*J. Comb. Optim.* **23** (2012) 159–166) provided the algorithm UND which solves one variant of P2. We are interested in solutions which are optimal from a worst-case perspective. For P1, we prove the worst-case input sequence and derive the algorithm *optimal portfolio for interrelated prices* (OPIP). We then prove the competitive ratio and optimality. We use the idea of OPIP to solve P2 and derive the algorithm called *optimal conversion for interrelated prices* (OCIP). Using OCIP, we also design optimal online algorithms for *bi-directional search* (P3) called *bi-directional UND* (BUND) and *optimal online search for unknown relative price bounds* (RUN). We run numerical experiments and conclude that OPIP and OCIP perform well compared to other algorithms even if prices do not behave adverse.

Mathematics Subject Classification. 90C27.

Received June 25, 2018. Accepted September 4, 2018.

1. Introduction

The *portfolio selection problem* (P1) is one of the most prominent examples of making decisions which are only optimal – in monetary terms – with knowledge about the future. The problem itself consists in allocating wealth to a set of stocks in a capital market; for this allocation to be optimal, knowledge about the performance of the assets in the next trading period – a year, a month, a day, or even a second – is needed. Since a correct prediction about the behavior of the market is often not realizable, online (sometimes also written as on-line) algorithms have been developed. Based on the past movements of the market, they attempt to maximize the ratio of the value of the portfolio in the very last trading period n and the invested money in the very first one.

P1 is one of many problems in the field of the so called "online optimization"; this field in turn is an instance in the much broader area of combinatorial optimization in dynamic environments (see [2]). The literature in the field of online optimization focuses mostly on a competitive analysis, *i.e.* the design of algorithms whose performance is guaranteed to be within a certain bound compared to the best solution. Sleator and Tarjan [17] introduced the performance measure called "competitive ratio" in order to compare the competitiveness of such algorithms. In simple terms, this ratio is the relative difference of an online algorithm and the optimal solution

Keywords. Competitive analysis, portfolio selection, two-way trading, online algorithms, bi-directional search.

[1] LCOMS EA 7306, Université de Lorraine, Metz, France.
[2] Max-Planck Institute for Informatics, Saarbrücken, Germany.
*Corresponding author: pascal.schroeder@univ-lorraine.fr

in a worst-case scenario. Solutions that incur the lowest ratio are called optimal. Recently, Gangolf *et al.* [8] applied the concept of competitive ratio to automated credit rating prediction algorithms in order to evaluate their performance in worst-case scenarios. Schroeder and Kacem [15] derived optimal online algorithms for the cash-management problem with uncertain and interrelated demands.

One way of evaluating online portfolio selection algorithms is the comparison of the achieved value of the portfolio in the last period using such an algorithm to the one of an offline benchmark like *best constant rebalanced portfolio* (BCRP) and *best asset* (BA). While BA selects that asset with the relatively highest increase in value, BCRP constantly selects that portfolio allocation vector which yields the highest increase in value when applied constantly. Borodin *et al.* [3] define a third offline benchmark called OPT. OPT solves P1 optimally because it invests all wealth in the best performing asset of each period. Clearly, it is not possible to perform better than OPT. All three offline benchmarks have complete knowledge about the future, *i.e.* the input of the problem, and they differ only in their feasible set of decision possibilities, *i.e.* the allowed output.

In the P1 literature, it is common practice to employ BCRP as an offline benchmark. A substantial number of algorithms, like the *universal portfolio* (UP) [6] and *exponential gradient* (EG) [9] algorithm, provide a performance guarantee with respect to BCRP. Further, they attempt to track the portfolio allocation vector which BCRP employs.

Online algorithms that are capable of achieving a high empirical portfolio wealth like the *passive aggressive mean reversion* (PAMR) [12] or *Anticor* (AC) [4] algorithm are denoted as heuristic algorithms. These online algorithms are often capable of substantially surpassing the performance of BCRP (see [10]). However, they fail to guarantee a non-trivial upper bound compared to any offline benchmark.

Closely associated with P1 is the *bi-directional trading problem* (P2). In literature, P2 is often referred to as two-way trading problem. Here one trades (or converts) back and forth between two assets and aims at maximizing the amount of units of one asset. Dochow [7] showed that P2 is a special case of P1. The achieved wealth is the main performance measure in P1; in P2, however, it is the competitive ratio. In literature, there are many optimal algorithms, when P2 is restricted to only one direction, *i.e.* when trading is uni-directional. With this restriction, Chen *et al.* [5] and Zhang *et al.* [19] developed optimal online algorithms called UND and DIV for P2 with interrelated prices. According to Zhang *et al.* [19], the motivation for modeling prices this way can be found in the stock market of China mainland; here the prices of each stock are merely allowed to drop or increase by 10% per day. Recently, Schroeder *et al.* [14, 16] developed optimal uni-directional online algorithms for selected variants of P2, notably the algorithm *profit UND* (PUND) and *profit DIV* (PDIV) that solve the restricted P2 optimally with a profit function and interrelated prices. Another variant of P2 is the so called *bi-directional search problem* (P3). P3 is a special instance of P2; here an investor is allowed to trade back and forth between two assets; however, the trade is "all or nothing". Recently, Schmidt [13] solved P3 to optimality when global price bounds and number of price runs are given.

The motivation for this work is the following:

(1) The concept of interrelated prices has not yet been applied to P1. Consequently, there exists no optimal online algorithm with given interrelated price bounds,
(2) even though there are some non-optimal solutions for P2 with unknown interrelated price bounds, there exists no optimal solution for P2 with known or unknown bounds,
(3) there exists no optimal solution for P3 with known or unknown bounds.

In this work we develop a competitive portfolio selection algorithm called *optimal portfolio for interrelated prices* (OPIP) with the offline benchmark being OPT. OPIP solves P1 optimally when maximal upward and downward relative price changes of each asset are known. In order to derive OPIP, we first apply the concept of competitive analysis to P1. OPIP is competitive and optimal. Based on the ideas of OPIP, we derive the new online algorithm *optimal conversion for interrelated prices* (OCIP) which solves P2 optimally online when an investor is allowed to be arbitrarily invested in two assets; note that OCIP can also be applied, when the interrelated price bounds are unknown. Based on OCIP, we derive the *bi-directional UND* (BUND) and *optimal*

online search for unknown relative price bounds (RUN) algorithm for P3. We then carry out a numerical testing to evaluate the performance of these algorithms. The contributions of this work are the following:

(1) The portfolio selection problem with interrelated prices is optimally solved online,
(2) bi-directional trading with known or unknown interrelated prices is optimally solved online,
(3) bi-directional search with known or unknown interrelated prices is optimally solved online,
(5) numerical experiments indicate which online algorithm is best for P2 and given interrelated price bounds,
(6) numerical experiments confirm the good performance of OPIP for P1 compared to other online algorithms.

This work is organized in six sections. In Section 2 P1 and the concept of competitive analysis is presented. The section closes with offline benchmarks for P1. In Section 3 we derive OPIP. Subsequently, we prove its competitiveness and optimality. In Section 4 we present OCIP and demonstrate the competitive ratio and optimality. We then present the BUND and RUN algorithm and provide detailed proofs for competitive ratio and optimality. In Section 5 we conduct numerical experiments to show the performance of the proposed algorithms. The paper ends with a conclusion in Section 6.

2. Preliminaries

2.1. The portfolio selection problem

P1 is described as follows. Consider a market of m assets with n trading periods with an arbitrary period $i = 1, \ldots, n$. Each asset $j = 1, \ldots, m$ has a price $p_{j,i}$ at the end of period i, with $\mathbf{p}_i = (p_{1,i}, \ldots, p_{m,i})$ denoting the vector containing the price of every asset at the end of period i. The price vector $\mathbf{p}_0$ comprises the initial prices of all assets before the first trading period. The relative price change $x_{j,i}$ of an asset j during period i is denoted as $x_{j,i} = p_{j,i}/p_{j,i-1}$; $\mathbf{x}_i = (x_{1,i}, \ldots, x_{m,i})$ contains all relative price changes in period i. All relative price change vectors of all periods are contained in the sequence $\mathbf{x}$, formally $\mathbf{x} = \mathbf{x}_1, \ldots, \mathbf{x}_n$.

The fraction of wealth invested in an asset j during period i is denoted as $b_{j,i}$; $\mathbf{b}_i = (b_{1,i}, \ldots, b_{m,i})$ is the vector that includes all segments of the invested wealth in period i. The sequence which contains all portfolio allocation vectors is $\mathbf{b}$, formally $\mathbf{b} = \mathbf{b}_1, \ldots, \mathbf{b}_n$. There are restrictions as to the instances of an arbitrary $\mathbf{b}_i$. First, consumption of wealth is excluded over all trading periods, *i.e.* the investor is always invested in the market with all her wealth. Second, short-selling of assets is prohibited; the investor is always non-negatively invested in all assets of the market. Consequently, the allowed domain Δ_m for all $\mathbf{b}_i$ is

$$\Delta_n = \left\{ \mathbf{b}_i \in \mathbb{R}^n : b_{j,i} \geq 0, \text{ for } \forall j \text{ and } \sum_{j=1}^{m} b_{j,i} = 1, \forall i \right\}. \tag{2.1}$$

The wealth after n periods with the market sequence $\mathbf{x}$ and the portfolio allocation sequence $\mathbf{b}$ is

$$W_n(\mathbf{x}, \mathbf{b}) = W_0 \prod_{i=1}^{n} \sum_{j=1}^{n} b_{j,i} x_{j,i}, \tag{2.2}$$

with W_0 being the initial wealth at the start of the first period. As in the literature, we set $W_0 = 1$.

One common objective in P1 is the maximization of $W_n(\mathbf{x}, \mathbf{b})$. One needs to decide on the allocation of wealth at the beginning of period i; however, the needed information about $\mathbf{x}_i$ is revealed at the end of that same period, when it is no longer useful for optimization. Hence, a maximization of equation (2.2) is only possible, if the sequence of price changes $\mathbf{x}$ is known in hindsight.

2.2. Competitive analysis

Sleator and Tarjan [17] designed the competitive analysis in order to evaluate the performance of an online algorithm. There are two parameters that affect the competitive ratio c^{ALG} of an arbitrary algorithm ALG, one

being the selected $\mathbf{b}$, and the other being the feasible sequence $\mathbf{x}$. The family of sequences we permit is denoted as $\mathbf{X}$ with $\mathbf{x} \in \mathbf{X}$.

$$c^{\mathrm{ALG}} = \max_{\mathbf{x}\in\mathbf{X}} \frac{W_n\left(\mathbf{x}, \mathbf{b}^{\mathrm{OPT}}\right)}{W_n\left(\mathbf{x}, \mathbf{b}^{\mathrm{ALG}}\right)}. \tag{2.3}$$

In this work we restrict $\mathbf{X}$ in the following way. For all $\mathbf{x} \in \mathbf{X}$ it holds: the maximal (minimal) relative change from price $p_{j,i}$ to $p_{j,i+1}$ is bounded by u_j (d_j) for an arbitrary asset j (first introduced by Zhang *et al.* [19]). For our setting we have

$$d_j \leq \min_{i=1,\ldots,n} x_{j,i} \leq \max_{i=1,\ldots,n} x_{j,i} \leq u_j \tag{2.4}$$

for all $j = 1, \ldots, m$.

One desired quality of an online algorithm ALG is the one of optimality, *i.e.* the fact that there exists no other algorithm that guarantees a smaller competitive ratio than the one guaranteed by ALG for all $\mathbf{x} \in \mathbf{X}$. Hence, if an online algorithm is optimal for a given P1 with arbitrary information about $\mathbf{X}$, then it is optimally solved online. Formally, an optimal algorithm ALG must always ensure that

$$c^{\mathrm{ALG}} = \min_{\mathbf{b}} \max_{\mathbf{x}\in\mathbf{X}} \frac{W_n\left(\mathbf{x}, \mathbf{b}^{\mathrm{OPT}}\right)}{W_n\left(\mathbf{x}, \mathbf{b}\right)}. \tag{2.5}$$

2.3. Offline benchmarks

Borodin *et al.* [3] present three offline benchmarks that are optimally solving the portfolio selection problem to a different extent. The first one is called *Best Asset* (BA); it invests all wealth into the one asset which will have the highest relative price increase in sequence $\mathbf{x}$, formally

$$\mathbf{b}_i^{\mathrm{BA}} = \arg\max_{\mathbf{b}_1\in\Delta_n} \mathbf{b}_1 \prod_{i_1=1}^{n} \mathbf{x}'_{i_1}, \forall i. \tag{2.6}$$

Note that $\mathbf{x}'_{i_1}$ is the transposed vector of $\mathbf{x}_i$. BA yields the optimal solution over the vertex of Δ_n. The *Best Constant Rebalanced Portfolio* (BCRP) constantly invests all wealth into that allocation which will have the highest relative increase in sequence $\mathbf{x}$, formally

$$\mathbf{b}_i^{\mathrm{BCRP}} = \arg\max_{\mathbf{b}_1\in\Delta_n} \prod_{i_1=1}^{n} \sum_{j=1}^{m} b_{j,1} x_{j,i_1}, \forall i. \tag{2.7}$$

BCRP yields the optimal solution over the full simplex Δ_n. The benchmark OPT invests all wealth in every trading period i into that asset which will have the highest relative increase in all $\mathbf{x}_i$, formally

$$\mathbf{b}_i^{\mathrm{OPT}} = \arg\max_{\mathbf{b}_1\in\Delta_n} \mathbf{b}_1 \mathbf{x}'_i, \forall i \tag{2.8}$$

and resulting $\mathbf{b}^{\mathrm{OPT}} = \mathbf{b}_1^{\mathrm{OPT}}, \ldots, \mathbf{b}_n^{\mathrm{OPT}}$. OPT yields the optimal solution over the vertex of Δ_n in every period i. Clearly, the terminal wealth incurred by OPT is the highest possible wealth and it is at least as high as the one incurred by BCRP and that in turn is at least as high as the one of BA.

3. Optimal portfolio for interrelated prices (OPIP)

We first introduce a function $F\left(\mathbf{x}_i, \mathbf{b}_i\right)$ which calculates the ratio of relative wealth increase gained by OPT and the wealth increase incurred by an online player using $\mathbf{b}_i$. Formally, we have

$$F\left(\mathbf{x}_i, \mathbf{b}_i\right) = \frac{\max_{j=1,\ldots,m} x_{j,i}}{\sum_{j=1}^{m} b_{j,i} x_{j,i}}. \tag{3.1}$$

Intuitively, we know that

$$\prod_{i=1}^{n} F\left(\mathbf{x}_i, \mathbf{b}_i\right) = \frac{W_n\left(\mathbf{x}, \mathbf{b}^{\mathrm{OPT}}\right)}{W_n\left(\mathbf{x}, \mathbf{b}\right)}. \tag{3.2}$$

Therefore, maximizing $F\left(\mathbf{x}_i, \mathbf{b}_i\right)$ in every period i is equal to selecting that market sequence $\mathbf{x}$ which achieves the competitive ratio.

We are interested in maximizing the function $F\left(\mathbf{x}_i, \mathbf{b}_i\right)$; this can be achieved by raising an arbitrary asset j to the maximal feasible price $(p_{j,i-1}u_j)$ and dropping all other assets $j' \neq j$ to the minimum feasible price $(p_{j',i-1}d_{j'})$. Now we only need to evaluate m relative price change vectors for the function, formally

$$F_{j,i}\left(\mathbf{b}_i\right) = \frac{u_j}{b_{j,i}\left(u_j - d_j\right) + \sum_{j'=1}^{m} b_{j',i} d_{j'}}, \tag{3.3}$$

being the adapted function; since $F_{j,i}\left(\mathbf{b}_i\right)$ does only depend on the selected $\mathbf{b}_i$, we erase the time component and set $F_{j,i}\left(\mathbf{b}_i\right) = F_j\left(\mathbf{b}_i\right)$.

Since there are many assets to choose from, we need to make sure, that we only consider those ones which are not dominated by any other asset. We state that one asset j dominates an asset j' $(j \succ j')$ if $d_j \geq u_{j'}$. Clearly, OPT does not invest anything into j' at any period. Therefore, we ought to first eliminate all assets which are dominated in order to avoid unnecessary calculations.

Assume there is no dominated asset. We first need to prove the worst-case price changes for our selected $\mathbf{b}_i$.

Lemma 3.1. *The worst-case relative price change vector* $\mathbf{x}_i^{wc}$ *during* i *is*

$$\mathbf{x}_i^{wc} = \left(d_1, \ldots, d_{j^*-1}, u_{j^*}, d_{j^*+1}, \ldots, d_n\right), \tag{3.4}$$

with

$$j^* = \arg \max_{j=1,\ldots,m} F_j\left(\mathbf{b}_i\right).$$

Proof. Assume that we select an arbitrary allocation $\mathbf{b}_i$ at i. Using equation (3.3) for all m relative price change vectors, that asset j^* which maximizes $F_j\left(\mathbf{b}_i\right)$ raises by the factor u_{j^*} and all other assets fall by the factor d_j. This finishes the proof. □

We now present the first new online algorithm *optimal portfolio for interrelated prices* (OPIP).

Algorithm OPIP. At each period of time, allocate the wealth according to vector $\mathbf{b}_i^{\mathrm{OPIP}}$ with

$$\mathbf{b}_i^{\mathrm{OPIP}} = \arg \min_{\mathbf{b}_1 \in \Delta_m} \max_{j=1,\ldots,m} F_j\left(\mathbf{b}_1\right). \tag{3.5}$$

Theorem 3.2. *For arbitrary* u_j *and* d_j*, OPIP's competitive ratio is*

$$c^{\mathrm{OPIP}} = \left(\max_{j=1,\ldots,m} F_j\left(\mathbf{b}_1^{\mathrm{OPIP}}\right)\right)^n. \tag{3.6}$$

Proof. Assume we use $\mathbf{b}_1^{\mathrm{OPIP}}$ for period 1. Thus, the ratio of OPT and us using $\mathbf{b}_1^{\mathrm{OPIP}}$ is $\max_{j=1,\ldots,m} F_j\left(\mathbf{b}_1^{\mathrm{OPIP}}\right)$. In $i = 2$ we use $\mathbf{b}_2^{\mathrm{OPIP}} = \mathbf{b}_1^{\mathrm{OPIP}}$ and find the same ratio again (since nothing changed compared to period 1). This repeats until n. Thus, the competitive ratio is

$$c^{\mathrm{OPIP}} = \left(\max_{j=1,\ldots,m} F_j\left(\mathbf{b}_1^{\mathrm{OPIP}}\right)\right)^n. \tag{3.7}$$

This finishes the proof. □

Theorem 3.3. *The OPIP algorithm is optimal, i.e. there exists no other online algorithm which guarantees a smaller competitive ratio than* c^{OPIP}.

Proof. Assume there exists an online algorithm ALG* which uses $\mathbf{b}_i^*$ at every i and incurs a smaller competitive ratio c^{ALG^*}. Thus, we have

$$\begin{aligned} c^{\mathrm{ALG}^*} &< c^{\mathrm{OPIP}} \\ &< \left(\max_{j=1,\ldots,m} F_j\left(\mathbf{b}_1^{\mathrm{OPIP}}\right)\right)^n \\ &< \left(\min_{\mathbf{b}_1\in\Delta_n} \max_{j=1,\ldots,m} F_j\left(\mathbf{b}_1\right)\right)^n. \end{aligned} \tag{3.8}$$

This is not possible by definition. Hence, there exists no such algorithm ALG* and OPIP incurs the lowest possible competitive ratio. This finishes the proof. □

We now consider the case that all assets share the same u and d. In other words, $u_1 = u_2 = \cdots = u_m = u$ and $d_1 = d_2 = \cdots = d_m = d$. $F_j(\mathbf{b}_i)$ then simplifies to

$$\hat{F}_j\left(\mathbf{b}_i\right) = \frac{u}{b_{j,i}\left(u-d\right)+d}. \tag{3.9}$$

Proposition 3.4. *For* $u_1 = \cdots = u_m = u$ *and* $d_1 = \cdots = d_m = d$, *the competitive ratio of OPIP is* $\left(\frac{m}{1+(m-1)\,du^{-1}}\right)^n$. *If* u *and* d *are unknown, then it is* m^n *(as proven in [3]).*

Proof. For $u_1 = \cdots = u_m = u$ and $d_1 = \cdots = d_m = d$, the following holds. All assets can increase by the factor u and decrease by the factor d, in other words, they are identical. Therefore, the wealth must be uniformly distributed among those m assets, formally

$$\mathbf{b}_i^{\mathrm{OPIP}} = \left(\frac{1}{m},\ldots,\frac{1}{m}\right)\forall i. \tag{3.10}$$

Using Theorem 3.2 we have

$$\begin{aligned} c^{\mathrm{OPIP}} &= \left(\max_{j=1,\ldots,m} \hat{F}_j\left(\mathbf{b}_1^{\mathrm{OPIP}}\right)\right)^n \\ &= \left(\max_{j=1,\ldots,m} \frac{u}{b_{j,i}\left(u-d\right)+d}\right)^n \\ &= \left(\frac{mu}{u+(m-1)\,d}\right)^n \\ &= \left(\frac{m}{1+(m-1)\,du^{-1}}\right)^n. \end{aligned} \tag{3.11}$$

For unknown u and d (notably, $du^{-1} \to 0$), c^{OPIP} is at most m^n. This finishes the proof. □

4. Competitive algorithms for P2 and P3 with interrelated prices

We now turn to P2. In P2 we start with one asset (a Dollar (D)) and we are offered a price in the beginning of period i. In the end of every period i with $i = 1,\ldots,n$ a price $p_{D,i}$ is announced which contains an amount of another asset (a Yen (Y)). In the beginning of every period i with $i = 1,\ldots,n$, we can trade fractions of D for Y (receiving $p_{D,i-1}$ units of Y for one D) and the other way around ($p_{Y,i-1} = {p_{D,i-1}}^{-1}$ units of D for one Y). The allowed domain of P2 is Δ_2. The goal is to maximize the wealth in D after the second to last period $n-1$. In n we must again be fully invested in D and there is no price movement. Clearly, maximizing the wealth

of a portfolio is equivalent to maximizing the obtainable amount of any asset. Thus, OPT still uses the same portfolio allocation vector and invests all wealth in that asset which will rise highest during an arbitrary period i.

Having used $\mathbf{b}_{i-1}^{\mathrm{ALG}}$ of an arbitrary online algorithm ALG, we denote the wealth of our portfolio in terms of units of D at the end of period $i-1$ as $D_{i-1}\left(\mathbf{b}_{i-1}^{\mathrm{ALG}}\right)$. At the beginning of i, our wealth in D consists of

$$D_{i-1}\left(\mathbf{b}_{i-1}^{\mathrm{ALG}}\right) = b_{D,i} D_{i-1}\left(\mathbf{b}_{i-1}^{\mathrm{ALG}}\right) + \left(1 - b_{D,i}\right) D_{i-1}\left(\mathbf{b}_{i-1}^{\mathrm{ALG}}\right) p_{D,i-1}. \tag{4.1}$$

At the end of period i, our wealth $D_i\left(\mathbf{b}_i^{\mathrm{ALG}}\right)$ changes to

$$\begin{aligned} D_i\left(\mathbf{b}_i^{\mathrm{ALG}}\right) &= b_{D,i} D_{i-1}\left(\mathbf{b}_{i-1}^{\mathrm{ALG}}\right) + \left(1 - b_{D,i}\right) D_{i-1}\left(\mathbf{b}_{i-1}^{\mathrm{ALG}}\right) \tfrac{p_{D,i-1}}{p_{D,i}} \\ &= D_{i-1}\left(\mathbf{b}_{i-1}^{\mathrm{ALG}}\right)\left(b_{D,i} + \left(1 - b_{D,i}\right) x_{D,i}^{-1}\right). \end{aligned} \tag{4.2}$$

Consequently, after n periods, the amount of D is

$$\begin{aligned} D_n\left(\mathbf{b}_n^{\mathrm{ALG}}\right) &= D_{n-1}\left(\mathbf{b}_{n-1}^{\mathrm{ALG}}\right)\left(b_{D,n}^{\mathrm{ALG}} + \left(1 - b_{D,n}^{\mathrm{ALG}}\right) x_{D,n}^{-1}\right) \\ &= D_0 \prod_{i=1}^{n} b_{D,i}^{\mathrm{ALG}} + \left(1 - b_{D,i}^{\mathrm{ALG}}\right) x_{D,i}^{-1}. \end{aligned} \tag{4.3}$$

We now proceed to solve P2.

4.1. Optimal conversion for interrelated prices (OCIP)

As in P1, we restrict the price by the upward and downward factor u and d. P2 can be seen as an instance of P1, in which the upward factor u of asset D is the inverse downward factor of asset Y; the downward factor d of D is the inverse upward factor of asset Y, formally

$$u_D = \frac{1}{d_Y} \text{ and } d_D = \frac{1}{u_Y}. \tag{4.4}$$

For this special case, we denote the worst-case ratio $G_i\left(\mathbf{b}_i^{\mathrm{ALG}}\right)$ of OPT and an online algorithm ALG in terms of relative dollar increase as

$$\begin{aligned} G_i\left(\mathbf{b}_i^{\mathrm{ALG}}\right) &= \max_{x_{D,i} \in [d_D, u_D]} \left(\frac{\max\left(1, x_{D,i}^{-1}\right)}{b_{D,i}^{\mathrm{ALG}} + \left(1 - b_{D,i}^{\mathrm{ALG}}\right) x_{D,i}^{-1}} \right) \\ &= \max\left(\frac{\max\left(1, u_D^{-1}\right)}{b_{D,i}^{\mathrm{ALG}} + \left(1 - b_{D,i}^{\mathrm{ALG}}\right) u_D^{-1}}, \frac{\max\left(1, d_D^{-1}\right)}{b_{D,i}^{\mathrm{ALG}} + \left(1 - b_{D,i}^{\mathrm{ALG}}\right) d_D^{-1}} \right) \\ &= \max\left(\frac{\max\left(1, u^{-1}\right)}{b_{D,i}^{\mathrm{ALG}} + \left(1 - b_{D,i}^{\mathrm{ALG}}\right) u^{-1}}, \frac{\max\left(1, d^{-1}\right)}{b_{D,i}^{\mathrm{ALG}} + \left(1 - b_{D,i}^{\mathrm{ALG}}\right) d^{-1}} \right). \end{aligned} \tag{4.5}$$

We simplify the notation by shortening u_D to u and d_D to d. Knowing that prices do not change in period n, *i.e.* $G_n\left(\mathbf{b}_n^{\mathrm{ALG}}\right) = 1$, we find that

$$\begin{aligned} \prod_{i=1}^{n} G_i\left(\mathbf{b}_i^{\mathrm{ALG}}\right) &= \prod_{i=1}^{n-1} G_i\left(\mathbf{b}_i^{\mathrm{ALG}}\right) 1 \\ &= \frac{D_{n-1}\left(\mathbf{b}_{n-1}^{\mathrm{OPT}}\right)}{D_{n-1}\left(\mathbf{b}_{n-1}^{\mathrm{ALG}}\right)} \\ &= c^{\mathrm{ALG}}. \end{aligned} \tag{4.6}$$

We observe that $G_i\left(\mathbf{b}_i^{\mathrm{ALG}}\right)$ is independent of the current prices of the assets. Therefore, $G_i\left(\mathbf{b}_i^{\mathrm{ALG}}\right) = G\left(\mathbf{b}_i^{\mathrm{ALG}}\right)$ and we have

$$\mathbf{b}_1^* = \arg\min_{\mathbf{b}_1 \in \Delta_2} G\left(\mathbf{b}_1\right). \tag{4.7}$$

Considering equations (4.5) and (4.7), we can state the following: If $d \geq 1$, then we set $b_{D,i} = 1$; this minimizes $G_i\left(\mathbf{b}_i^{\mathrm{ALG}}\right)$ to 1. Along the same lines, we find that if $u \leq 1$ then we set $b_{D,i} = 0$ and minimize $G_i\left(\mathbf{b}_i^{\mathrm{ALG}}\right)$ to 1.

We now consider all other constellations of u and d and their implications for $b_{D,i}$. Notably, $G\left(\mathbf{b}_i^{\mathrm{ALG}}\right)$ further simplifies to

$$G\left(\mathbf{b}_i^{\mathrm{ALG}}\right) = \max\left(\frac{1}{b_{D,i}^{\mathrm{ALG}} + \left(1 - b_{D,i}^{\mathrm{ALG}}\right) u^{-1}}, \frac{d^{-1}}{b_{D,i}^{\mathrm{ALG}} + \left(1 - b_{D,i}^{\mathrm{ALG}}\right) d^{-1}}\right). \tag{4.8}$$

We equate both elements of the max function and receive

$$b_{D,i} = \frac{1 - u^{-1}}{2 - u^{-1} - d}. \tag{4.9}$$

We now present the new online algorithm *optimal conversion for interrelated prices* (OCIP) for P2.

Algorithm OCIP. In n, select $b_{D,n}^{\mathrm{OCIP}} = 1$. In each period of time $i = 1, \ldots, n-1$, invest the fraction $b_{D,i}^{\mathrm{OCIP}}$ in D and the fraction $b_{Y,i}^{\mathrm{OCIP}}$ in Y, with

$$b_{D,i}^{\mathrm{OCIP}} = \begin{cases} 1 & \text{if } d \geq 1, \\ 0 & \text{if } u \leq 1, \\ \dfrac{1 - u^{-1}}{2 - u^{-1} - d} & \text{else,} \end{cases} \tag{4.10}$$

and $b_{Y,i}^{\mathrm{OCIP}} = 1 - b_{D,i}^{\mathrm{OCIP}}$. We now proceed with the proof of the competitive ratio.

Theorem 4.1. *For arbitrary u and d, OCIP's competitive ratio is*

$$c^{\mathrm{OCIP}} = \begin{cases} 1 & \text{if} \quad u \leq 1 \vee d \geq 1, \\ \left(1 + \dfrac{1}{1 - du^{-1}}\right)^{n-1} & \text{else.} \end{cases} \tag{4.11}$$

For unknown u and d it is at most 2^{n-1}.

Proof. The trivial if case has already been verified in equation (4.7) and the consecutive text. Hence, we focus on the non-trivial else case. Applying the OCIP algorithm to equation (4.6), we have

$$\begin{aligned} c^{\mathrm{OCIP}} &= \frac{D_{n-1}\left(\mathbf{b}_{n-1}^{\mathrm{OPT}}\right)}{D_{n-1}\left(\mathbf{b}_{n-1}^{\mathrm{OCIP}}\right)} \\ &= \frac{D_0 \prod_{i=1}^{n-1} b_{D,i}^{\mathrm{OPT}} + \left(1 - b_{D,i}^{\mathrm{OPT}}\right) u^{-1}}{D_0 \prod_{i=1}^{n-1} b_{D,i}^{\mathrm{OCIP}} + \left(1 - b_{D,i}^{\mathrm{OCIP}}\right) u^{-1}} \\ &= \frac{1}{\left(b_{D,i}^{\mathrm{OCIP}} + \left(1 - b_{D,i}^{\mathrm{OCIP}}\right) u^{-1}\right)^n} \\ &= \left(\frac{2 - du^{-1}}{1 - du^{-1}}\right)^{n-1} \\ &= \left(1 + \frac{1}{1 - du^{-1}}\right)^{n-1}. \end{aligned} \tag{4.12}$$

We cannot prevent $d, u^{-1} \to 0$ for unknown u and d. In this case, we have $c^{\mathrm{OCIP}} = 2^{n-1}$. □

Theorem 4.2. *The OCIP algorithm is optimal, i.e. there exists no other online algorithm which guarantees a smaller competitive ratio than* c^{OCIP}.

Proof. Using Theorem 4.1, we find that $c^{\mathrm{OCIP}} = 1$ if $u \leq 1 \vee d \geq 1$. Thus, OCIP is optimal for these combinations because it is not possible to perform better than OPT. For all other parameter combinations, OCIP provides that $\mathbf{b}_1^*$ which minimizes equation (4.7) and is thus minimizing the competitive ratio (see Eq. (4.6)). This finishes the proof. □

We now shortly consider the case of unknown u and d. Notably, we cannot prevent that $u^{-1} \to 0$ and $d \to 0$. Using equation (4.10), we find $b_{D,i}^{\mathrm{OCIP}} = b_{Y,i}^{\mathrm{OCIP}} = 0.5$ for $i = 1, \ldots, n-1$. Thus, if u and d are unknown, then OCIP suggests to be equally invested in both assets.

4.2. Bi-directional UND (BUND)

In P2 we have $m = 2$. We now restrict the allowed domain Δ_2 in the following way (see Eq. (2.1)). We are merely allowed to trade all D(Y) for Y(D), formally

$$\Delta_2 = \{\mathbf{b}_i \in \{(1,0), (0,1)\} \forall i\}. \tag{4.13}$$

We will solve this variant of P2 (denoted as *bi-directional search problem* (P3)) with knowledge about u, d and n. Clearly, according to Lemma 3.1, whenever we trade all D for Y in i for price $p_{D,i-1} = p_{i-1}$, the price will increase by $p_{i-1}u$. Likewise, if we convert all of Y into D in i for price p_{i-1}, then the price will decrease by $p_{i-1}d$. We now present the algorithm *bi-directional UND* (BUND).

Algorithm BUND. At n, $b_{D,n}^{\mathrm{OCIP}} = 1$. In periods $i = 1, \ldots, n-1$ be invested in D according to $b_{D,i}^{\mathrm{BUND}}$ with

$$b_{D,i}^{\mathrm{BUND}} = \begin{cases} 1 & \text{if } 1 \leq d \leq u \vee ud \geq 1, \\ 0 & \text{else.} \end{cases} \tag{4.14}$$

We now prove the competitive ratio of BUND and its optimality.

Theorem 4.3. *For arbitrary* u *and* d*, BUND's competitive ratio is*

$$c^{\mathrm{BUND}} = \begin{cases} 1 & \text{if } d \leq u \leq 1 \vee 1 \leq d \leq u, \\ \min\left(u^{n-1}, d^{1-n}\right) & \text{else.} \end{cases} \tag{4.15}$$

Proof. The trivial if case has already been verified in equation (4.7) and the consecutive text ($b_{D,i} = 1$ and $b_{D,i} = 0$ are still within the reduced Δ_2). Hence, we focus on the non-trivial else case. First, assume $ud \geq 1$; consequently, we do not convert at all, *i.e.* $\mathbf{b}_i^{\mathrm{BUND}} = (1,0), \forall i$. According to equations (4.8) and (4.6), we have

$$\begin{aligned} c^{\mathrm{BUND}} &= \prod_{i=1}^{n-1} G\left(\mathbf{b}_i^{\mathrm{BUND}}\right) \\ &= \max\left(\frac{1}{1}, \frac{d^{-1}}{1}\right)^{n-1} \\ &= \min\left(u^{n-1}, d^{1-n}\right). \end{aligned} \tag{4.16}$$

Now assume $ud < 1$; we trade all D for Y at $i = 1$ and all of Y to D at $i = n$, *i.e.* $\mathbf{b}_i^{\mathrm{BUND}} = (0,1)$ for $i = 1, \ldots, n-1$ and $\mathbf{b}_n^{\mathrm{BUND}} = (1,0)$. According to equations (4.8) and (4.6), we have

$$\begin{aligned} c^{\mathrm{BUND}} &= \prod_{i=1}^{n-1} G\left(\mathbf{b}_i^{\mathrm{BUND}}\right) \\ &= \max\left(\frac{1}{u^{-1}}, \frac{d^{-1}}{d^{-1}}\right)^{n-1} \\ &= \min\left(u^{n-1}, d^{1-n}\right). \end{aligned} \tag{4.17}$$

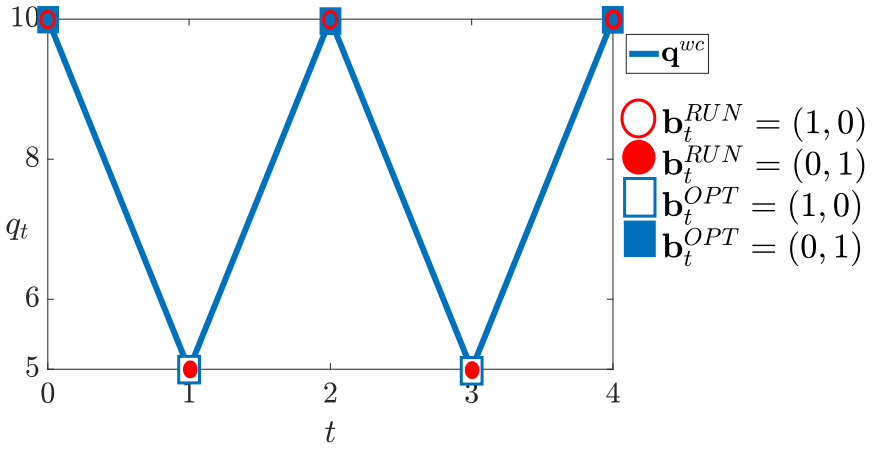

FIGURE 1. Worst-case sequence $\mathbf{p}^{wc}$ for the RUN algorithm.

Using equations (4.16) and (4.17), we have $c^{\mathrm{BUND}} = \min\left(d^{1-n}, u^{n-1}\right)$ for the else case. This finishes the proof. □

Theorem 4.4. *The BUND algorithm is optimal, i.e. there exists no other online algorithm which guarantees a smaller competitive ratio than* c^{BUND}.

Proof. For an arbitrary i we can only chose between the allocation vectors $(1,0)$ (completely invested in D) and $(0,1)$ (completely invested in Y). Assume there exists an arbitrary algorithm ALG' which proposes k times to be fully invested in D, *i.e.* $\mathbf{b}_i = (1,0)$, and $n-k-1$ times to be fully invested in Y, *i.e.* $\mathbf{b}_i = (0,1)$ (with $0 \leq k \leq n-1$). The order of the selected allocations is of no importance. Thus, we set $\mathbf{b}^{\mathrm{ALG}'}$ to contain k consecutive times $(1,0)$ followed by $n-1-k$ times $(0,1)$ and $\mathbf{b}_n^{\mathrm{ALG}'} = (1,0)$. Thus we have

$$\begin{aligned} c^{\mathrm{ALG}'} &= \textstyle\prod_{i=1}^{k} G\left(\mathbf{b}_i^{\mathrm{ALG}'}\right) \prod_{i=k+1}^{n-1} G\left(\mathbf{b}_i^{\mathrm{ALG}'}\right) \\ &= d^{-k} u^{n-k-1} \\ &\geq \min\left(d^{1-n}, u^{n-1}\right). \end{aligned} \tag{4.18}$$

Thus, ALG' is at best capable of achieving c^{BUND} (when setting $k=0$ or $k=n-1$). However, a lower competitive ratio is not obtainable. This finishes the proof. □

4.3. RUN

We now design an algorithm for the case of unknown u and d. The algorithm RUN solves this problem. The main idea of RUN is to convert D(Y) to Y(D) right after a local price maximum (minimum).

Algorithm RUN. In $i=1,n$ set $\mathbf{b}_i^{\mathrm{RUN}} = (1,0)$. For $i=2,\ldots,n-1$: if $p_i < p_{i-1}$, then $\mathbf{b}_i^{\mathrm{RUN}} = (0,1)$; else, $\mathbf{b}_i^{\mathrm{RUN}} = (1,0)$.

Figure 1 illustrates the worst-case price sequence $\mathbf{p}^{wc}$ for $u = 2 = d^{-1}, n = 5, p_0 = 10$ and the output of OPT and RUN. In $i=1$ RUN selects $\mathbf{b}_1^{\mathrm{RUN}} = (1,0)$. OPT therefore sets $\mathbf{b}_1^{\mathrm{OPT}} = (0,1)$, and the price drops to $p_0 d$ for $i=1$. In $i=2$ we have $\mathbf{b}_2^{\mathrm{RUN}} = (0,1)$. OPT sets $\mathbf{b}_2^{\mathrm{OPT}} = (1,0)$ and the price increases to $p_0 du$. This pattern continues until $n-1$.

We now prove the competitive ratio of RUN and its optimality for the given P3.

Theorem 4.5. *For arbitrary* u *and* d *(notably* $du^{-1} \to 0$*), RUN's competitive ratio is*

$$c^{\mathrm{RUN}} = u^{\left\lfloor \frac{n-1}{2} \right\rfloor} d^{-\left\lceil \frac{n-1}{2} \right\rceil}.$$

Proof. The worst-case price sequence $\mathbf{p}^{wc}$ starts with a local maximum p_0, followed by a local minimum $p_1 = p_0 d$, followed by a local maximum $p_2 = p_1 u$, etc.. We apply RUN and trade all D for Y in $i = 2$ and all of Y for D in $i = 3$. We trade all D for Y in all local minimum and all Y for D in all local maximum; OPT trades all D for Y in all local maximum and all Y for D in all local minimum. This forms the worst-case price sequence $\mathbf{p}^{wc}$, formally

$$\mathbf{p}^{wc} = p_0, p_0 d, p_0 du, \ldots, p_0 d^{\left\lceil \frac{n-1}{2} \right\rceil} u^{\left\lfloor \frac{n-1}{2} \right\rfloor} \tag{4.19}$$

and a corresponding $\mathbf{x}^{wc}$.

For odd n, OPT's terminal amount of D is

$$D_{n-1}\left(\mathbf{b}^{\mathrm{OPT}}\right) = d^{\frac{1-n}{2}}, \tag{4.20}$$

while we receive

$$D_{n-1}\left(\mathbf{b}^{\mathrm{RUN}}\right) = u^{\frac{1-n}{2}}. \tag{4.21}$$

For even n, OPT's terminal amount of D is

$$D_{n-1}\left(\mathbf{b}^{\mathrm{OPT}}\right) = d^{\frac{-n}{2}}, \tag{4.22}$$

while we receive

$$D_{n-1}\left(\mathbf{b}^{\mathrm{RUN}}\right) = u^{\frac{2-n}{2}}. \tag{4.23}$$

For arbitrary n, OPT receives

$$D_{n-1}\left(\mathbf{b}^{\mathrm{OPT}}\right) = d^{-\left\lceil \frac{n-1}{2} \right\rceil}, \tag{4.24}$$

while we receive

$$D_{n-1}\left(\mathbf{b}^{\mathrm{RUN}}\right) = u^{-\left\lfloor \frac{n-1}{2} \right\rfloor}. \tag{4.25}$$

The resulting competitive ratio is

$$\begin{aligned} c^{\mathrm{RUN}} &= \frac{D_{n-1}\left(\mathbf{b}^{\mathrm{OPT}}\right)}{D_{n-1}\left(\mathbf{b}^{\mathrm{RUN}}\right)} \\ &= u^{\left\lfloor \frac{n-1}{2} \right\rfloor} d^{-\left\lceil \frac{n-1}{2} \right\rceil}. \end{aligned} \tag{4.26}$$

This finishes the proof. □

Theorem 4.6. *The RUN algorithm is optimal, i.e. there exists no other online algorithm which guarantees a smaller competitive ratio than* c^{RUN}.

Proof. We use algorithm ALG' as described in Theorem 4.4. Notably, the competitive ratio of ALG' is

$$\begin{aligned} c^{\mathrm{ALG}'} &= \frac{u^{n-k-1}}{d^k} \\ &= d^{-k} u^{n-k-1}. \end{aligned} \tag{4.27}$$

We denote the ratio of $c^{\mathrm{ALG}'}$ and c^{RUN} as ϕ, formally

$$\begin{aligned}\phi &= \frac{d^{-k}u^{n-k-1}}{d^{-\left\lceil\frac{n-1}{2}\right\rceil}u^{\left\lfloor\frac{n-1}{2}\right\rfloor}} \\ &= d^{\left\lceil\frac{n-1}{2}\right\rceil-k}u^{n-k-1-\left\lfloor\frac{n-1}{2}\right\rfloor} \\ &= d^{\left\lceil\frac{n-2k-1}{2}\right\rceil}u^{\left\lceil\frac{n-2k-1}{2}\right\rceil} \\ &= (ud)^{\left\lceil\frac{n-2k-1}{2}\right\rceil}.\end{aligned} \tag{4.28}$$

If RUN is not optimal, then $\phi < 1$ must hold. However, without knowledge about u and d we cannot assure that the selected k implies $\phi < 1$. For instance, if we set k such that the exponent is at least zero, then we cannot prevent that (ud) is smaller than or equal to one; this implies $\phi \geq 1$. Likewise, if we set k such that the exponent is at best zero, then we risk $(ud) \geq 1$ and thus $\phi \geq 1$. This finishes the proof. □

5. Numerical experiments

The OPIP algorithm is executed using IBM ILOG CPLEX 12.7.1 and illustrated in Matlab. The algorithms OCIP, RUN and BUND are executed using VBA and also illustrated in Matlab. All experiments are tested on a personal computer with an Intel Core i5-6300 2.4 GHz CPU and 16 GB of RAM. We also include the *naïve diversification* (ND) which equally distributes the wealth between all assets, *i.e.* $\mathbf{b}_i = n^{-1}, \ldots, n^{-1}$ for all i.

5.1. OPIP

In P1 it is common practice to compare the performance of an algorithm on several empirical datasets containing daily prices in several stock markets. We use the four stock market datasets used in [4, 12] called NYSE, TSE, DIJA and SP500; details concerning the different datasets are given in Table 1 (see also [12]).

We compare OPIP with the two offline benchmark algorithms BA and BCRP, the learning algorithms Universal Portfolio (UP) according to [11], Exponential Gradient (EG) with $\eta = 0.05$ as suggested by Helmbold *et al.* [9] and Online Newton Step (ONS) with $\eta = 0, \beta = 1$ and $\gamma = 0.125$ as suggested by Agarwal *et al.* [1]. The wealth achieved by those five benchmarks in those four datasets can be found in [12]. We also add the so called market method, *i.e.* at $i = 1$ we are equally invested in all assets and do not trade later on. The highest terminal wealth achieved in each dataset by an online benchmark is highlighted in bold font. The terminal wealth for the different methods on the datasets is shown in Table 2.

We see that OPIP is performing quite well compared to other learning algorithms. Except for the DIJA dataset OPIP's wealth always increases over time and is often capable of surpassing EG and UP (except for SP500). One must note however, that while EG and ONS have their respectively best parameter setting in terms of terminal wealth, OPIP does not influence its distinct parameters (they are given by the dataset itself);

Table 1. Summary of the four real datasets in our numerical experiments.

Dataset	Region	Time frame	n	m
NYSE	US	7/3/1962 – 12/31/1984	5651	36
TSE	CA	1/4/1994 – 12/31/1998	1259	88
SP500	US	1/2/1998 – 1/31/2003	1276	25
DIJA	US	1/14/2001 – 1/14/2003	507	30

TABLE 2. Terminal wealth achieved by various portfolio selection strategies on the four datasets.

Method/Dataset	NYSE	TSE	SP500	DIJA
Market	14.50	1.61	1.34	0.76
BA	54.14	6.28	3.78	1.19
BCRP	250.60	6.78	4.07	1.24
UP	26.68	1.60	1.62	0.81
EG	27.09	1.59	1.63	0.81
ONS	**109.19**	1.62	**3.34**	**1.53**
ND	27.08	1.54	1.65	0.81
OPIP	74.62	**1.95**	1.05	0.93

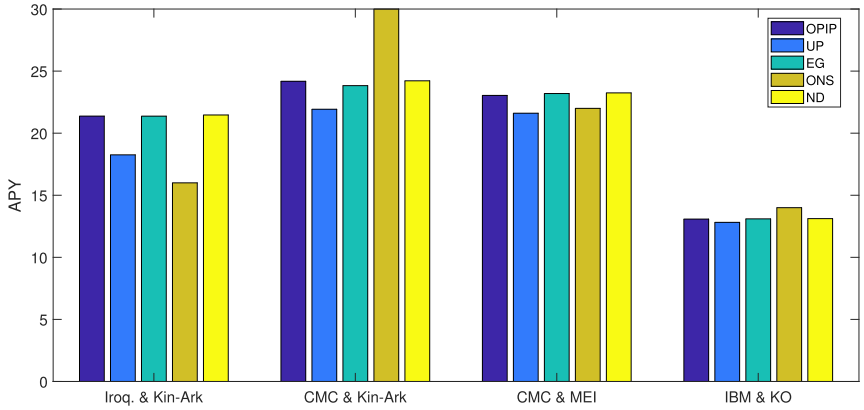

FIGURE 2. APY of selected algorithms on four pairs of stocks tested by Cover [6], Helmbold *et al.* [9] and Agarwal *et al.* [1].

also, UP does not use any parameter and its performance is therefore not depending on any knowledge about the future input nor any parameter optimization. We must keep in mind that UP, EG and ONS are algorithms that aim at tracking BCRP, while OPIP aims at minimizing its relative distance to OPT in worst-case input sequences.

We now consider four pairs of assets in the NYSE dataset, that is Iroquois and Kin-Ark (Iro & Kin-Ark), Commercial Metals Company and Kin-Ark (CMC & Kin-Ark), CMC and Meicco Corp. (CMC & MEI) and IBM and Coca Cola (IBM & KO). Cover [6] used these pairs to illustrate the behavior of the UP algorithm and later on Helmbold *et al.* [9] and Agarwal *et al.* [1] also used them to demonstrated the behavior of EG and ONS. We compare the *annual percentage yield* (APY) of each algorithm on each pair; formally, the APY for a period of 22 years is

$$\text{APY} = W_n(\mathbf{x}, \mathbf{b})^{22^{-1}} - 1. \tag{5.1}$$

The respective APY of each algorithm is illustrated in Figure 2.

As we can see OPIP is never performing worse than all other algorithms. Except for the pair CMC & Kin-Ark, OPIP's APY is among the highest ones possible.

We conclude that OPIP is not only providing the lowest competitive ratio possible, but it is also performing empirically well compared to other algorithms that have a worse performance guarantee.

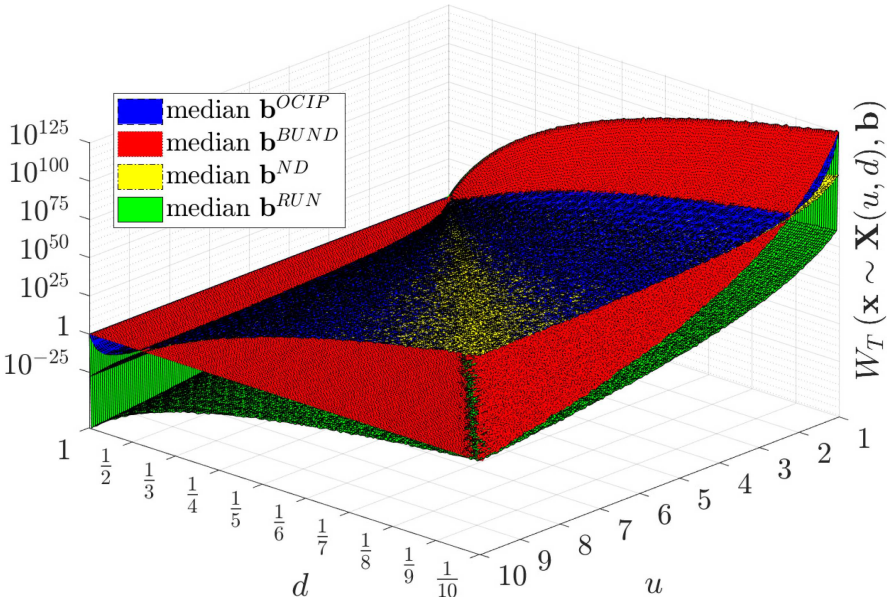

FIGURE 3. Median of OCIP, BUND,ND and RUN for various u and d on the experimental setting.

5.2. OCIP

In P2 it is not common to conduct numerical experiments. The main literature focuses on finding optimal algorithms and rarely conducts any sort of performance testing. We consider P2 for $n = 250$, various u, d and compare OCIP with ND, BUND and RUN. Our research question we want to answer is as follows. Given a parameter combination of u and d, which algorithm performs best in terms of some selected performance measures. These insights are particularly relevant for practitioners.

The sequence of all considered upward factors u starts at 1 and increases absolutely by 0.05, the last value being 10. The decrease factor also starts at 1 and decreases relatively as much as u increases, the last value being 0.1. For every pair of u and d we conduct 1000 experiments. In an arbitrary experiment we use the random variable $x \in X \sim \mathcal{U}(0,1)$ which originates from the standard uniform distribution to derive future price changes with

$$x_{D,i} = u^{2\max(x,0.5)-1} d^{1-2\min(x,0.5)}. \tag{5.2}$$

The thereby constructed sequence $\mathbf{x}$ comes from the family of all feasible sequences that use u and d denoted as $\mathbf{X}(u,d)$.

The parameter p_0 is of no importance for the testing of OCIP (we therefore set it equal to 1). After all experiments for a pair u and d have been calculated, we derive the median, the 0.5% (5 permille) and the 99.5% (995 permille) of the incurred wealth in terms of D of each algorithm. While the median gives us an idea of the general performance of the algorithms, the permille measures illustrate experimental worst and best-case performance of each algorithm.

We proceed with the testing of OCIP, BUND, ND and RUN. In Figure 3 we see the median performance of all four algorithms.

Starting with ND, we see that as u increases and d remains close to 1, ND looses wealth (for $d = 1$ and $u = 10$ the wealth achieved is 3.20×10^{-21}). This is not surprising; when prices increase by far more than they fall, a shifting from D to Y means that the accumulated Y tend to lose their value; since ND proposes to be always equally invested, the amount of D in the portfolio diminishes, while the amount of Y increases. When all wealth needs to be traded for D, the price for one unit of that asset is so high, the investor does hardly

get any; this leads to a massive loss in wealth. OCIP also looses wealth (the lowest wealth being 1.88×10^{-7}) but recognizes that the wealth must be increasingly invested in D. As $u \to 10$ the wealth invested in D reaches 100%. The result of BUND is not surprising since all pairs of u and d with $u \geq d^{-1}$ lead BUND to not trade at all. Thus the investor does never loose the invested D if this relation between u and d holds true. As $d \to 0.1$ and $u \to 10$ we observe that OCIP and ND are well performing (with $D_n(\mathbf{b}_n) = 3.16 \times 10^{21}$ for both). In this area (especially around the space diagonal) often times ND wins, while OCIP is only insignificantly worse. Both profit from the volatility of both assets. However, as this area is left, OCIP starts to outperform ND (left and right side of the space diagonal) because it puts an increasing weight on the one asset which falls less strong (and increases sharper). For all pairs $u \leq d^{-1}$, BUND trades all wealth for Y at the very beginning and trades back at the very end. Therefore, as $d \to 0.1$ and $u \leq d^{-1}$, BUND starts to increase in wealth, until it surpasses the wealth generated by OCIP. The highest possible wealth achieved by BUND is 5.35×10^{62} for $u = 1$ and $d = 10.00^{-1}$. Again, this is not surprising for BUND; when prices tend to fall extremely, then it is reasonable to trade D for Y as soon as possible; the value of Y compared to D then increases intensely and the one Y is traded back for a large amount of D. Fortunately, OCIP also recognizes that the portfolio must be shifted toward Y and is thus able to stay close to the result of BUND. We note that ND also increases in wealth; this is because of ND tending to trade Y for D at increasingly high prices. However, ND trades too many Y too early and therefore fails to achieve a wealth close to OCIP. Concerning RUN, we observe that it is clearly outperformed by all other algorithms. Only if $d = 1$ or $u = 1$, it performs as good as BUND and OPIP. For $u = d^{-1}$ RUN has a higher median performance than does BUND. We derive the following findings for the median experimental performance of the considered online algorithms:

(1) ND outperforms all other algorithms when $u \approx d$,
(2) BUND outperforms all other algorithms when u is close to 1 or d is close to 1,
(3) OCIP outperforms all other algorithms for all other parameter combinations,
(4) RUN never outperforms all algorithms at once.

This finishes the interpretation of Figure 3.

In Figure 4 we see the 5 permille performance of all four algorithms.

We first observe that the general level of terminal wealth dropped compared to Figure 3. Looking at the space diagonal, we observe that left to it, OCIP still outperforms the other algorithms; however, ND achieves a higher terminal wealth right to the space diagonal when prices behave adversely. Compared to OCIP, the observed "too much too early" trading behavior of ND (see the interpretation of Fig. 3) is advantageous for adverse prices. OCIP keeps more Y in the portfolio (betting on low prices) and suffers bigger losses when prices are rising compared to ND. This advantage of ND over OCIP fades out as $u \to 1$ and $d \to 0.1$. Left to the space diagonal, we observe that OCIP outperforms ND when prices show adverse behavior. This is because OCIP does keep a higher fraction of the initial D in the portfolio, thereby reducing the potential value losses that can occur when trading between D and Y. ND, on the other side, keeps too many units of Y in the portfolio and suffers larger value losses when prices increase. Concerning RUN, we note that it performs better than BUND for some pairs of $u < d^{-1}$. Concerning BUND, we observe that it outperforms all other algorithms as long as $u < 2d^{-1}$; the amount of pairs of u and d left to the space diagonal for which BUND outperforms all other algorithms increased. Right to the space diagonal, the amount of dominating combinations for BUND declined. For all other pairs, we observe the prior findings of the median performance. We derive the following findings for the 5 permille experimental performance of the considered online algorithms:

(1) ND outperforms all other algorithms when $u \leq d^{-1}$ and $u \geq (3d)^{-1}$ holds true,
(2) BUND outperforms all other algorithms when u is close to 1 or $u > 2d^{-1}$,
(3) OCIP outperforms all other algorithms for all other parameter combinations,
(4) RUN never outperforms any algorithm.

This finishes the interpretation of Figure 4.

In Figure 5 we see the 995 permille performance of all four algorithms.

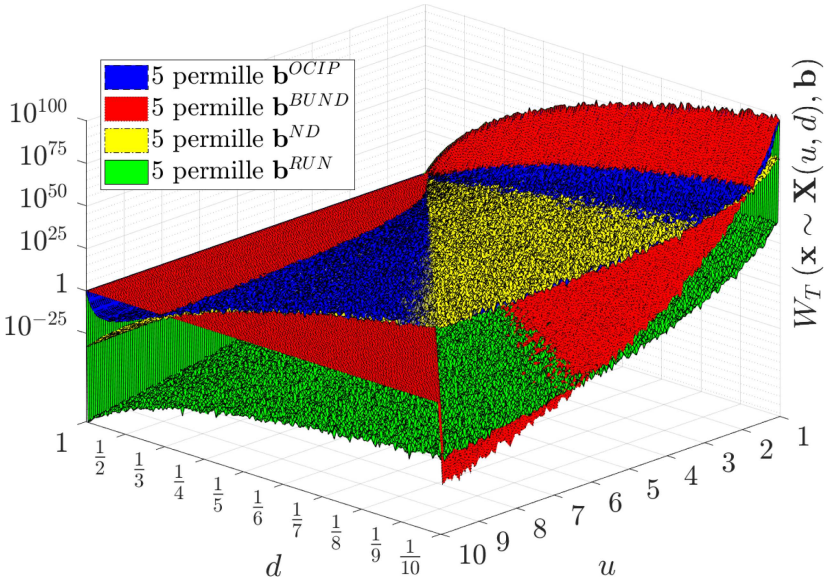

FIGURE 4. 5 permille of OCIP, BUND, ND and RUN for various u and d on the experimental setting.

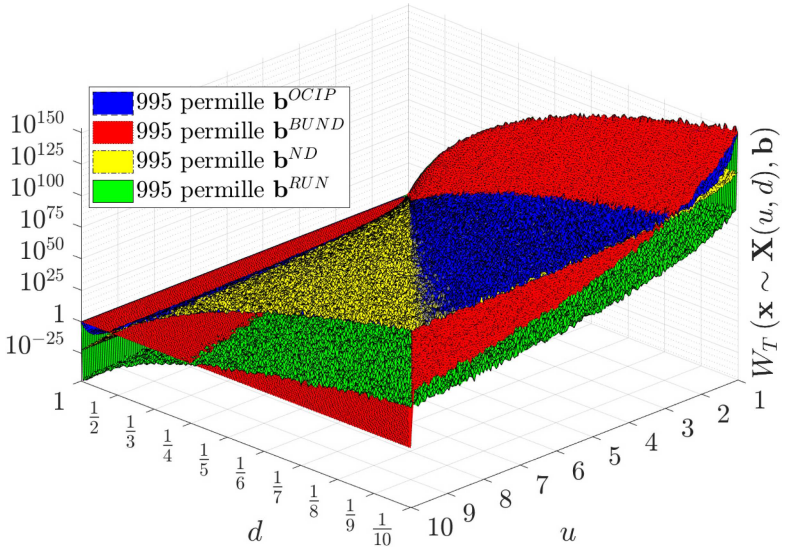

FIGURE 5. 995 permille of OCIP, BUND,ND and RUN for various u and d on the experimental setting.

We first observe that the general terminal wealth level surpassed that of Figure 3. More importantly, we interpret Figure 5 to be the counterpart of Figure 4. In Figure 4, we observed that ND's outperforming combinations of u and d are right to the space diagonal; in Figure 5, we see that an equal-size area of outperforming combinations for ND of u and d is left to the space diagonal. The same holds true for OCIP; its former outperforming area of combinations moved from the left to the right side of the space diagonal. In addition, the

small strip where OCIP is better than all other algorithms has moved from the right side to the left side of the space diagonal. BUNDs large (small) area of outperforming combinations of u and d has moved from the left (right) side of the space diagonal to the right (left) side. We derive the following findings for the 995 permille experimental performance of the considered online algorithms:

(1) ND outperforms all other algorithms when $u \geq d^{-1}$ and $d \geq 3\,(u)^{-1}$ holds true,
(2) BUND outperforms all other algorithms when d is close to 1 or $u < (2d)^{-1}$,
(3) OCIP outperforms all other algorithms for all other parameter combinations,
(4) RUN never outperforms any algorithm.

This finishes the interpretation of Figure 5.

Summing up the results, we find that ND is performing well given the fact that it has no knowledge about future price sequences (ND does not take u and d into consideration as opposed to BUND and OCIP). Furthermore, there exist price sequences and parameter combinations for which it is more profitable to trade all or nothing (which BUND does) than to trade fractions of the initial wealth. OCIP unites both the ability to trade in fractions back and forward between D and Y as well as the knowledge about the bounds of the interrelated relative price changes. RUN performs poorly in all three performance measures and is therefore not very suitable for practical application. This concludes this section.

6. Conclusion

To the best of our knowledge, it is the first time that the portfolio selection problem has been optimally solved online on markets where prices are interrelated. We developed OPIP, which solves this problem and provided proofs for the competitive ratio and optimality. From this solution, we considered bi-directional trading and used ideas of OPIP to solve the problem optimally online. This resulted in the new algorithm OCIP; from this, we then developed the solutions for bi-directional search called BUND and RUN. All three algorithms address currently unsolved cases in the field of online conversion problems; competitiveness and optimality were proven for all three algorithms. We ran numerical experiments to gain further insights regarding their performance.

Acknowledgements. This work has been supported by FP7-EYE. The short version of this work has been partially presented at the 3rd international conference on Control, Decision and Information Technologies (CoDIT16), St. Julian's, 2016.

References

[1] A. Agarwal, E. Hazan, S. Kale and R.E. Schapire, Algorithms for portfolio management based on the newton method. In: *Proceedings of the 23rd International Conference on Machine Learning*. ACM (2006) 9–16.
[2] N. Boria and V.T. Paschos, A survey on combinatorial optimization in dynamic environment. *RAIRO: OR* **45** (2011) 241–294.
[3] A. Borodin, R. El-Yaniv and V. Gogan, On the competitive theory and practice of portfolio selection (Extended Abstract). In: *Proceedings of the 4th Latin American Symposium on Theoretical Informatics* (2000) 173–196.
[4] A. Borodin, R. El-Yaniv and V. Gogan, Can we learn to beat the best stock. *J. Artif. Intell. Res. AI Access Found.* **21** (2004) 579–594.
[5] G.-H. Chen, M.-Y. Kao, Y.-D. Lyuu and H.-K. Wong, Optimal buy-and-hold strategies for financial markets with bounded daily returns. *SIAM J. Comput.* **31** (2001) 447–459.
[6] T. Cover, Universal portfolios. *Math. Finance* **1** (1991) 1–29.
[7] R. Dochow, Online Algorithms for the Portfolio Selection Problem. Springer, Berlin (2016).
[8] C. Gangolf, R. Dochow, G. Schmidt and T. Tamisier, Automated credit rating prediction in a competitive framework. *RAIRO: OR* **50** (2016) 749–765.
[9] D. Helmbold, R. Schapire, Y. Singer and M. Warmuth, On-line portfolio selection using multiplicative updates. *Math. Finance* **8** (1998) 325–347.
[10] D. Huang, J. Zhou, B. Li, S.C. Hoi and S. Zhou, Robust median reversion strategy for on-line portfolio selection. In: *Proceedings of the 23rd International Joint Conference on Artificial Intelligence* (2013) 2006–2012.
[11] A. Kalai and S. Vempala, Efficient algorithms for universal portfolios. *J. Mach. Learn. Res.* **3** (2002) 423–440.
[12] B. Li, P. Zhao, S. Hoi and V. Gopalkrishnan, PAMR: passive aggressive mean reversion strategy for portfolio selection. *Mach. Learn.* **87** (2012) 221–258.

[13] G. Schmidt, Competitive analysis of bi-directional non-preemptive conversion. *J. Comb. Optim.* **34** (2017) 1096–1113.

[14] P. Schroeder, R. Dochow and G. Schmidt, Conversion algorithms with a reward function and interrelated conversion rates. In: *Proceedings of the 45th International Conference on Computers and Industrial Engineering*. France (2015) 536–543.

[15] P. Schroeder and I. Kacem, Optimal cash management with uncertain and interrelated demands. In: *Proceedings of the 47th International Conference on Computers and Industrial Engineering. Portugal* (2017) 502–508.

[16] P. Schroeder, R. Dochow and G. Schmidt, Optimal solutions for the online time series search and one-way trading problem with interrelated prices and a profit function. *Comput. Ind. Eng.* **119** (2018) 465–471.

[17] D.D. Sleator and R.E. Tarjan, Amortized efficiency of list update and paging rules. *Commun. ACM* **28** (1985) 202–208.

[18] R. El-Yaniv, A. Fiat, R. Karp and G. Turpin, competitive analysis of financial games. In: *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science* (1992) 327–333.

[19] W. Zhang, Y. Xu, F. Zheng and Y. Dong, Optimal algorithms for online time series search and one-way trading with interrelated prices. *J. Comb. Optim.* **23** (2012) 159–166.