INDIAN INSTITUTE OF MANAGEMENT KOZHIKODE

Working Paper

IIMK/WPS/219/ITS/2017/03

January 2017

Portfolio choice decision making with NBP-effSAMWMIX: A Stochastic Multi-Armed Bandit Algorithm using Naïve Bandit Portfolio Approach

Boby Chaitanya Villari[1]
Mohammed Shahid Abdulla[2]

[1] Doctoral Student, IT & Systems Area, Indian Institute of Management Kozhikode, IIMK Campus P.O, Kerala – 673570, India, E-mail: Bobycv06fpm@iimk.ac.in
[2] Associate Professor, IT & Systems Area, Indian Institute of Management Kozhikode, IIMK Campus P.O, Kerala – 673570, India, E-mail: shahid@iimk.ac.in, Phone: +91 - 495 - 2809254

IIMK WORKING PAPER

Portfolio choice decision making with *NBP-effSAMWMIX*: A Stochastic Multi-Armed Bandit Algorithm using Naïve Bandit Portfolio Approach

Boby Chaitanya Villari
Doctoral Student of IT & Systems Area, IIM Kozhikode

Mohammed Shahid Abdulla
Associate Professor of IT & Systems Area, IIM Kozhikode

Abstract

Portfolio Selection Problem (PSP) is actively discussed in financial research. The choice of available assets poses the need for exploration and the objective to maximize the portfolio payoffs makes the PCP an explore-exploit decision-making problem. Multi-armed bandit algorithms (MAB) suit well for such problems when applied as the decision engines in Naïve Bandit Portfolio algorithms (NBP). An NBP's performance varies by varying the MAB inside the algorithm. In this work we test a Stochastic Multi-Armed Bandit (SMAB) named *effSAMWMIX*, which we proposed in a previous work of ours, to solve the PSP. We compare the performance of effSAMWMIX vis-à-vis KL-UCB, Thompson Sampling algorithm and the benchmark Market Buy & Hold strategy. We tested the algorithms on simulated and real-world market datasets. We report our results where effSAMWMIX, applied as the decision-making engine of NBP, has achieved better cumulative wealth for all portfolios when compared to the competing SMAB algorithms.

Keywords: Portfolio Selection Problem, Multi-Armed Bandit, Geometric Brownian Motion

1 Introduction

Decision making under uncertainty has always been a challenge to a decision maker. A Portfolio Selection Problem (PSP) often encounters such uncertainty due to the changing economic and political environments. The fast proliferation of the information, on Internet-based modern day economies, imposes a need to make a quick decision based on the available but limited information. Many business problems including the Online Portfolio Selection Problem (OPSP) could require simultaneous optimization and the best choice identification. OPSP has been widely discussed in Computational Finance(Borodin and El-Yaniv 2005; Fiat 1998; Li and Hoi 2014; Mohr and Schmidt 2013; Schmidt, Mohr and Kersch 2010).

To solve any PSP, the investor decides on a strategy to allocate the available (finite) wealth among the available choice of assets. Every asset is a diverse investment opportunity and the realization of the asset allocation strategy builds a portfolio. An asset is risky if the prices of the asset are uncertain and such riskiness needs to be incorporated into the portfolio allocation process. The time between any two portfolio allocation decisions is called a period. If there is only one decision during the whole investment period, it is called a Single-Period PSP.A multi-period PSP requires sequential decision-making over the time horizon of investment and thus is proposed as an online decision-making problem. Investor's during the decision making is to optimize an objective decision which could be the Return on Investment(RoI) or Risk of losing wealth (Risk) or a combination of both (Risk & Return). Thus portfolio decision making could also involve the management of Risk and maximize the RoI. The following section briefly discusses the PSP and introduces a Machine Learning(ML) perspective of the OPSP problem.

2 OPSP & Machine Learning Algorithms

In the academic literature, OPSP is addressed in two ways. The first one considers the risk management into the objective function and thus the performance measure will be quantified by the Cumulative Wealth(CW) achieved or the net risk of the decision with respect to the achievable wealth. Such performance measures are seen from Markowitz's seminal work (Markowitz 1952) and also in Sharpe's work(Sharpe 1966).A few other works followed similar risk management measures until recently in 2011(Lisi 2011),(Rockafellar and Uryasev 2000).It can be observed that these ideas are characterized by building statistical models of the asset prices in the market. The input to these statistical models requires a forecasting model in some form(DeMiguel, Martín-Utrera and Nogales 2015).The forecasting model, in turn, requires a calibration based on historical data of asset prices(Sharpe 1963) or market capitalization data (Fama and French 1992).

The second way of addressing the OPSP is based on utilizing the modern day computing infrastructure along with intelligent ML techniques that include Neural Networks or Reinforcement Learning Algorithms(Shen, Wang, Jiang and Zha 2015).ML Algorithms are solely based on the empirical observations motivated by dynamic rise and fall of the asset prices. Algorithms (alternatively Strategies) like Follow-the-Winner(Agarwal, Hazan, Kale and Schapire 2006; Li and Hoi 2014), Follow-the-Loser(Li and Hoi 2012) etc. are a couple of those which make use of such dynamic price changes. In brief, an ML Algorithm's approach to OPSP is to concretely explore the available information of past asset prices and based on its indigenous technique, provide a suggestion as to how the portfolio allocation be done for the next period. The algorithm typically intends to maximize the cumulative wealth at the end of the multi-period investment horizon.

To analyze the performance of an OPSP strategy, the algorithm is run on a simulated data such as that obtained by simulating stock prices using Geometric Brownian Motion (GBM)(Marathe and Ryan 2005) and then on real-time market data obtained from standard datasets available from various sources(Bruni, Cesarone, Scozzari and Tardella 2016). It is to be noted that though it is fast to obtain the results by running the algorithm and thus easier to compare the performance with other strategies, the quality of the results heavily depends on the quality of the input data. Hence to avoid the dependency on the same, it is common that the performance of an algorithm is compared to a benchmark algorithm (Mohr, Ahmad and Schmidt 2014) which could be optimal in the hindsight. The Buy-and-Hold(Li and Hoi 2014) could be one such benchmark algorithm which the performance of an ML based OPSP algorithm is compared with.

This work considers using a Reinforcement Learning algorithm called effSAMWMIX which is a Stochastic Multi-Armed Bandit(SMAB) algorithm. We employ effSAMWMIX to build a Naïve Bandit Portfolio(NBP) similar to Shen et.al's work(Shen, Wang, Jiang and Zha 2015) and compare the same with NBPs that implement standard SMABs like UCB(Auer and Ortner 2010) ,KL-UCB(Garivier and Cappé 2011) and Thompson Sampling(Agrawal and Goyal 2012; Kaufmann, Korda and Munos 2012; Thompson 1933). We compare the performances over a multitude of simulated and benchmark datasets to analyze the results. The following section introduces SMAB and how an NBP is constructed.

3 Stochastic Multi-Armed Bandits & Naïve Bandit Portfolio Algorithm

An SMAB typically deals with an explore-exploit problem scenario where there are a set of available choices and the decision maker has to decide on only one of the choices can be opted. This is similar to that of a Portfolio Selection problem except that in a typical MAB, the payoffs of

the action other than the chosen action in that round remain unknown. But for a PSP, the data is available as the asset prices for the previous time period are known to the decision maker.

3.1 A Stochastic Multi-Armed Bandit

A Multi-Armed Bandit(MAB) problem is a sequential decision-making problem which spans over a horizon of decision-making rounds(T).In each round t the decision maker chooses an action a_t from among a set of K action choices that are available and obtains a reward $r_t^{a_t} \epsilon (0,1)$ for choosing arm a_t in round t .The choice is based on an objective function that maximizes the overall reward or cumulative wealth(CW) from the decision making process. Alternatively the objective function could be to minimize the Regret(R_t) which is the difference between the highest possible reward and the CW obtained by the algorithm up to that round of decision making.(Lai and Robbins 1985; Robbins 1985).

Thus, the cumulative reward of a MAB is $CW = \sum_{t=1}^{T} E(r_t^{a_t})$ (1)

Let the maximum possible reward be v^* and hence maximum reward for such an oracle policy is $T * v^*$.

Then the regret of the MAB algorithm is given by $R_T = Tv^* - \sum_{t=1}^{T} E(r_t^{a_t})$ (2)

A general MAB has no restrictions on the reward distribution. But a Stochastic Multi-Armed Bandit (SMAB) problem imposes an assumption that the rewards of each of the available action choices(arms) follow a fixed distribution $v_i on\ [0,1]$ unknown to the SMAB algorithm. Also the reward of each action i is independent of the rewards it obtained from any other time horizon (or pull) and independent of rewards of other actions (Robbins 1985).This means that the rewards $\{X_{i,t}\}_{t \epsilon N}$ are assumed to be independent and identically distributed (i.i.d) from v_i and all the rewards of K choices(arms) are also independent of each other.While there are a few other MAB settings like the adversarial MAB(Auer, Cesa-Bianchi, Freund and Schapire 1995) where the environment chooses the rewards so as to minimize the CW, we deal with SMABs with the rewards following the assumption stated above.

3.2 A Naïve Bandit Portfolio strategy

A Naïve Bandit Portfolio (NBP) implements an SMAB inside as a decision-making engine. The inputs to the SMAB are the time horizon T, number of available arms K where each arm represents a portfolio or asset, the time period between the decisions Δt which could be a day for daily returns or a week for weekly returns or month or an year. $r_t^{a_t}$is the reward obtained when an asset $a_t$is chosen in time period t and CW_t is the cumulative wealth obtained until the round t.
The gross return on i^{th}asset in round t is denoted by r_t^i and is obtained as given below

$$r_t^i = P_{t,i}/P_{t-1,i} \quad (3)$$

where $P_{t,i}$ is the price of the asset i at time t. Thus the returns in round t $\epsilon (1, T)$ over the time for the portfolio with K arms can be given as $\boldsymbol{R_t} = \left(r_{t,1}, r_{t,2}, r_{t,3} \dots \dots r_{t,K}\right)^T$. The portfolio investment decision over is the determination of the weights proportionate to which each the investment is budgeted over the assets available to the investor. Thus the weights vector at time t which is represented as

$$\omega_t = \left(\omega_{t,1}, \omega_{t,2,}, \omega_{t,3} \dots \dots \omega_{t,K}\right)^T \quad (4)$$

Thus $\omega_{t,i}$ which is the invested wealth percentage on the i^{th} asset in round t is determined by the SMAB algorithm.

$$\sum_{i=1}^{K} \omega_{t,i} = 1 \quad (5)$$

Also for an SMAB to operate, $r_t^i \epsilon (0,1)$ which means that the rewards are to be normalized and then given as the input to the algorithm. This is a critical condition to be imposed on the dataset in order to use any SMAB on an OPSP. Another critical aspect is the introduction of a parameter τ indicating the rolling-horizon settings(Shen, Wang and Ma 2014), which is the number of periods of data ,prior to the current decision making period, the algorithm should consider while calculating the necessary input parameters like the mean return, standard deviation and the Sharpe ratios(SR).Thus for an NBP, the return for an asset i in round t is given as $r_t^i = SR_t^i$ (6)

Where $$SR_t^i = \left(\frac{\mu_t^i}{\sigma_t^i}\right)_\tau . \quad (7)$$

The τ denotes the rolling-horizon time period.For example, $\tau = 120$ indicates that the mean(μ) and standard deviations(σ) of the past 120 time periods is taken in to consideration for calculating SR_t^i.Then to get the $SR_t^i \epsilon [0,1]$ normalize the SR_t^i for based on data for every asset i in period t.

$$nSR_t^i = \frac{SR_t^i - \min(SR_t^{(i=1\ to\ K)})}{\max(SR_t^{(i=1\ to\ K)}) - \min(SR_t^{(i=1\ to\ K)})} \quad (8)$$

The Naïve Bandit Portfolio algorithm's pseudocode is put below.

Inputs : K (assets), T (time horizon), Δt, $\mathbf{R_t}, \tau$
for $i = 1$ *to* $(T - \tau)$ ***do***
Step-1: *Estimate the averate return* $\mu_t^i = E(R_t)$ *based on previous* τ *periods*
Step-2: *Estimate the standard deviation* $\sigma_t^i = \sqrt{\lambda_t^i}$ *where*
λ_t^i *is the variance of asset* i*'s returns for the past* τ *periods*
Step-3: Using (7) Calculate the SR_t^i matrix for every asset for rounds $t = \tau$ *to* T.
Step-4: Using (8) Normalize the SR_t^i matrix so that every $nSR_t^i \epsilon [0,1]$ for round t = τ *to* T. *This* nSR_t^i *is the adjusted Reward function*
Step-5: *Provide* nSR_t^i *as the input for the SMAB algorithm*
which could be $UCB1, KL-UCB, Thompson\ Sampling\ or\ effSAMWMIX$
to compute $\omega_t = (\omega_{t,1}, \omega_{t,2,}, \omega_{t,3} \ldots\ldots \omega_{t,K})^T$
Output:
The portfolio weight vector $\omega_t = (\omega_{t,1}, \omega_{t,2,}, \omega_{t,3} \ldots\ldots \omega_{t,K})^T$
The Cumulative Wealth $CW = \prod^{T} (\omega_{t,1}, \omega_{t,2,}, \omega_{t,3} \ldots\ldots \omega_{t,K})^T * R_t$

Figure 1: A Naïve Bandit Portfolio (NBP) algorithm

From the Step 5 of NBP algorithm (Figure 1), it is seen that an SMAB is implemented to compute the weights vector for any round $\epsilon [1, T - \tau]$. Current work compares the performance of the proposed effSAMWMIX SMAB with UCB1, KL-UCB and, Thompson Sampling (TS) algorithms. The functioning of each of these algorithms is put below.

3.2.1 The UCB1 algorithm

UCB1 is among those first generation algorithms that update and consider both exploration and exploitation components in a same surrogate UCB parameter(Auer, Cesa-Bianchi and Fischer 2002). Consider the following UCB1 parameter to be updated in every iteration for every arm

$ucb_t^i = \max\left(\overline{x_t^i} + \sqrt{\frac{2\ln(t)}{n_i}}\right)$ where $\overline{x_t^i}$ is the empirical mean of observed reward of arm i until round t and n_i is the total number of times arm i is played until round t .This UCB estimate is similar to $ucb_i = \text{argmax}(\overline{\mu_i}) + \mathcal{P}_i$ for any round i. While $\text{argmax}(\overline{\mu_i}) = \max\left(\overline{x_i^k}\right)$ is the exploitation component, $\mathcal{P}_i = \sqrt{\frac{2\ln(k)}{n_i}}$ is the exploration bonus. Thus the optimistic guess parameter will get updated simultaneously with knowledge related to both exploration and exploitation. The UCB1 algorithm can be written as follows

1. Initialization : Play (execute) each arm once and obtain rewards x_i
2. Further , play the arm that satisfies $\text{argmax}\left(\overline{x_t^i} + \sqrt{\frac{2\ln(t)}{n_i}}\right)$ in that round
    a. n_i is number of times that particular arm is played so far
    b. t is the current round
    c. $\overline{x_t^i}$ is the average reward for arm i at round t

Figure 2:UCB1 algorithm

3.2.2 The KL-UCB algorithm

KL stands for Kullback-Leibler divergence and KL-UCB(Garivier and Cappé 2011) differs from UCB1 in the padding function $\mathcal{P}_i$ which is derived by employing KL divergence. The authors reported improved regret bounds for KL-UCB where $\mathcal{P}_i$ incorporates the distance between estimated reward distributions for the arms when calculating the UCB parameter. The algorithm is put below.

1. Initialization : Play (execute) each arm once and obtain rewards x_i
2. Further , play the arm that satisfies $\text{argmax}_i\left(n_i \,.\, d(\mu_i, M)\right) \leq \log t + c \log\log t$ in that round
    a. n_i is number of times that particular arm is played so far
    b. i is the arm
    c. $d(\mu_i, M) = \mu_i \log\left(\frac{\mu_i}{M}\right) + (1-\mu_i)\log\left(\frac{1-\mu_i}{1-M}\right)$

Figure 3: KL-UCB algorithm

3.2.3 The Thompson Sampling algorithm

Thompson Sampling(TS) heuristic was proposed by Thompson (Thompson 1933) in 1933 but remained less popular compared to other MAB algorithms for the lack of proofs on the regret bounds which were given very recently(Agrawal and Goyal 2012). Also, the proof for logarithmic regret to Thompson sampling has come only recently(Kaufmann, Korda and Munos 2012).It can also be argued that TS cannot be ignored for the lack of proofs on the regret for its empirical performance (in simulated environments) outperformed a few well known MAB algorithms for example UCB. Hence this work considers TS as a competent algorithm to compare with the effSAMWMIX. The TS algorithm is given below

1. Initialization:
    a. Set α, β which are the prior parameters for Beta distribution
    b. Set $S_i = 0, F_i = 0\ \forall_i$ where S_i *is success counter and* F_i *is Failure counter*
2. Loop: For every round $t = 1, \ldots . T$ do
    a. For every arm k= $1, \ldots . K$ do
        i. Draw θ_i according to $Beta(S_i + \alpha, F_i + \beta)$
    b. Draw an arm $i = argmax_i\ \theta_i$ and observe the reward r
    c. If $r = 1$ then $S_i = S_i + 1$
    d. Else $F_i = F_i + 1$

Figure 4: Thompson sampling algorithm

3.2.4 The effSAMWMIX algorithm

effSAMWMIX is based on SAMWMIX(Abdulla and Bhatnagar 2016).It differs from upper confidence bound like algorithms since it avoids searching for a maximum of a resulting parameter. Instead, it picks a 'soft-maximum' using a Boltzmann Exploration structure. It calculates a probability vector ϕ_k over the set of K arms.This ϕ_k is iteratively updated to obtain a ϕ^*which associates maximum probability of 1 to the best arm a^* and 0 to the rest of the arms. The equation for ϕ is put below

$$\phi_{t+1}^{j} = (1 - \gamma_t)\ \frac{e^{\Sigma \eta_t \widehat{X_t^j}}}{\sum_{j=1}^{N} e^{\Sigma \eta_t \widehat{X_t^j}}} + \frac{\gamma_t}{N} \qquad (9)$$

Where η_t is similar to that of SAMWMIX except that it is parameterized by a d_k which is obtained by the utilization of a heuristic. Equation (9), put above, is the same as that of equation (10) in SAMWMIX's original proposition. effSAMWMIX obtains the superiority on how the learning parameter γ_t is calculated as put below.

$$\gamma_k = \frac{N\left(4+(d+d_k)\right)}{k(d+d_k)^2-(d+d_k-2d^2)} \qquad (10)$$

Further, the parameter $d_k$is obtained by using a heuristic to quickly converge without requiring a closed form of an equation. The effSAMWMIX algorithm is put below.

Input : Rewards Vector G_t,set of Arms N,number of rounds T

1. Using G_t Calculate d=min $\Delta(\mu_1, \mu_2 \ldots . \mu_N)$ where μ_i is the reward mean of Arm i.
2. Calculate
    a. $C_0 = N + 1;\ \sigma^2 = 2 * N;$
    b. $\eta_0 = \frac{1}{C_0} \log\left(\frac{1+C_p * d}{\sigma^2}\right)$
    c. $startIter = ((4 + d) * N + d)/d^2$
3. $for\ i\ =\ 1, \ldots, N\ do$
    a. Obtain reward $X_{t=i}^{i}$
    b. Initialize $\phi_t^i\ = \eta_0 * \left(\frac{startIter}{N}\right) * \left(\frac{X_{t=i}^{i}}{\frac{1}{N}}\right)$
    c. Initialize pull count for $arm\ a^i\ as\ p_i$=1
4. $for\ t\ = (startIter + 1 + N), \ldots, (startIter + T) do$
    a. Obtain random probability r
    b. Choose an arm i as winner if $\sum \phi_t^i\ > r$ and store reward $G_{t-startIter}^{i} = a_t^*$ and normalize the reward using its probability $\widehat{X}$=a_t^*/ϕ_t^i

c. Update $p_i = p_i + 1$

d. $for\ d_t\ =\ 1, \dots, \frac{t-startIter}{T-N}\ in\ steps\ of\ d_{tstep}\ do$

i. Calculate $k_t = d + d_t$;

ii. $\gamma_{ttemp} = ((4 + k_t) * N + k_t)/(t * k_t^2)$

iii. $C_t = \left(\frac{N}{\gamma_{ttemp}}\right) + 1$ and $\sigma_{temp}^2 = 2 * N/\gamma_{ttemp}$

iv. $\eta_t = \frac{1}{C_t} \log\left(\frac{1+C_t * k_t}{\sigma_{temp}^2}\right)$

v. $\sum \phi_{ttemp}^i = \sum \phi_t^i + e^{\sum \eta_t \widehat{X_t^i}}$

vi. If $e^{\sum \eta_t * (d_t * \widehat{X_t^i}) + \eta_t * (d_t * \widehat{X_t^{a^t}})} > \sum \phi_{ttemp}^i$ then assign $d_{ttrue} = d_t - d_{tstep}$

e. Assign

i. $K_t = d_t + d_{ttrue}$

ii. $\gamma_t = ((4 + k_t) * N + k_t)/(t * k_t^2)$

iii. $C_t = \left(\frac{N}{\gamma_{ttemp}}\right) + 1$ and $\sigma_t^2 = 2 * N/\gamma_{ttemp}$

iv. $\eta_t = \frac{1}{C_t} \log\left(\frac{1+C_t K_t}{\sigma_t^2}\right)$

f. Now update ϕ_{t+1}^j using (1) which is

$$\phi_{t+1}^j = (1 - \gamma_t) \frac{e^{\sum \eta_t \widehat{x_t^j}}}{\sum_{j=1}^N e^{\sum \eta_t \widehat{x_t^j}}} + \frac{\gamma_t}{N}$$

Output : ϕ vector

Figure 5: effSAWMIX algorithm

The performance of the four algorithms in the current context are compared against simulated GBM portfolios and real-world benchmark datasets as explained in the following sections.

4 Experiments

Experiments are conducted on both simulated datasets and real-world datasets. The simulated datasets are obtained by using Geometric Brownian Motion prediction techniques as explained below

4.1 Stock prediction on Simulated Geometric Brownian Motion Datasets

Geometric Brownian Motion (GBM) is also known as Wiener Process in which the logarithm of a quantity that varies at random will follow a Brownian Motion(Wilmott 2000).GBM is formally a mathematical modeling technique that is often used to predict the short-term stock price movement(Ladde and Wu 2009).Since the stock price movement is often unpredictable the GBM's random walk model tends to predict the stock prices with reasonable accuracy(Fama 1995).This work considers the GBM technique to build a synthetic dataset to test the performance of the NBP algorithm that utilizes effSAMWMIX,UCB1,KL-UCB and TS as the SMAB engine for the NBP. The GBM dataset is generated using the daily closing prices which are the input for the GBM model. The returns are calculated for each asset using the following equation

$$R_i = \left(\frac{P_{i+1} - P_i}{P_i}\right) \quad (11)$$

Where P_i is the closing price of the asset on day i.If T is the total number of periods where the returns are calculated then the mean return μ is calculated as follows

$$\mu = \frac{1}{T} \sum_{i-1}^{T} R_i \quad (12)$$

Also the standard deviation of all the returns σ is calculated as

$$\sigma = \sqrt{\frac{1}{T-1} \sum_{i-1}^{T} (R_i - \mu)} \quad (13)$$

If the price of stock at time t is $S(t)$ and a random value generated at that time is denoted by $X(t)$ then the $S(t)$ is calculated using GBM as follows.

$$S(t) = S(0) * e^{[\mu - 0.5\sigma^2]t - \sigma[X(t) - X(0)]} \quad (14)$$

4.1.1 Experimental Settings & Simulation results

From the S&P 500 Stock dataset(Bruni, Cesarone, Scozzari and Tardella 2016) we have randomly picked n stocks from the same so that the SMABs will have n choices(arms) to choose from. Each of the stocks have the periodic closing prices from November 2004 to April 2016.The μ & σ are calculated for each of the 10 stocks and a new set of closing prices are predicted using GBM prediction equation (14). This newly generated stock price closing data will now be the dataset on which the NBP's performance is evaluated when the NBP uses a different SMAB for decision making process. The naming convention for the GBM simulated portfolio dataset with 5 assets $n = 5$ is GBM05 and that with 15 assets $n = 15$ is GBM15.The results of the experiments on GBM05 and GBM15 datasets are put below. Also effSAMWMIX employed inside NBP is henceforth addressed as NBP-effSAMWMIX. Similarly, NBP-UCB1 indicates that UCB1 is employed inside NBP. The other two algorithms are denoted as NBP-KLUCB and NBP-TS where KL-UCB and TS are employed inside NBP respectively.

Table 1: Terminal Cumulative Wealth on GBM Datasets.

	Cumulative Wealth($)				
	Market B&H Strategy	NBP-UCB1	NBP-KLUCB	NBP-TS	NPB-effSAMWMIX
GBM05 Dataset	1.173561174	1.3789034	0.862313301	1.51668599	1.621277794
GBM15 Dataset	1.288151551	1.402418615	1.559731636	1.704466302	1.786662446

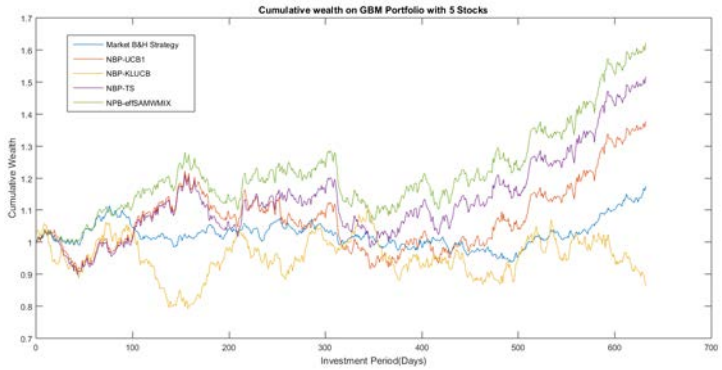

Figure 6: Cumulative wealth curves across the investment periods on GBM05 dataset

NBP-effSAMWMIX performed better than when NBP-UCB1, NBP-KLUCB, NBP-TS. Also, NBP-effSAMWMIX has acquired a better CW than the Market Buy & Hold Strategy(Li and Hoi 2014).Results are similarly favorable for NBP-effSAMWMIX when simulated portfolio consisted of 15 assets (see Fig.7.).The terminal cumulative wealth acquired per a unit investment is shown in Table 1.

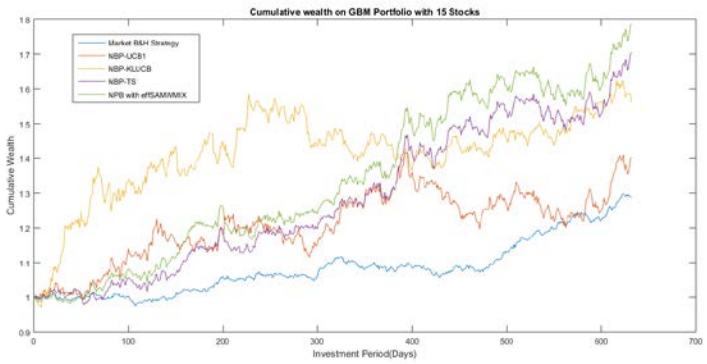

Figure 7: Cumulative wealth curves across the investment periods on the GBM15 dataset.

4.2 Stock prediction on real-world benchmark datasets

We choose benchmark datasets from (Bruni, Cesarone, Scozzari and Tardella 2016) and (Li, Sahoo and Hoi 2016) where the datasets are validated for the comparative performance of portfolio selection models. These datasets are generated using real-world price values obtained from major stock markets. They are reported to contain error-free cleaned data of weekly return values which are adjusted for dividends and stock splits. These publicly available datasets help in an unbiased comparison of the different NBP-SMAB portfolio selection strategies that are tested in this work. We chose these datasets to get a variety of data in terms of region, market type, the number of assets and the number of periods. For example, MSCI measures the equity market performance of global emerging markets and DJIA gives the stock market data from the USA which is a developed economy. Table 2. provides the details of the datasets under consideration for this work.

Table 2: Summary of the four benchmark datasets from real markets.

Dataset	**Market**	**Region**	**Time Frame**	**# Periods**	**# Assets**	**Reference**
DJIA	Stock	USA	01/14/2001 - 01/14/2003	507	30	(Li, Sahoo and Hoi 2016)
TSE	Stock	CANADA	01/04/1994 - 12/31/1998	1259	88	(Li, Sahoo and Hoi 2016)
NASDAQ100	Stock	USA	06/2002 - 04/2016(weekly)	596	82	(Bruni, Cesarone, Scozzari and Tardella 2016)
MSCI	Index	Global	01/14/2001 - 01/14/2003	507	30	(Li, Sahoo and Hoi 2016)

We report (Table 3.) the terminal cumulative wealth achieved by each of these algorithms over the four benchmark datasets mentioned above. Apparently, NBP-effSAMWMIX has achieved the highest cumulative wealth when compared to other NBP algorithms. Except in the case of NASDAQ100 dataset, NBP-effSAMWMIX has performed better than the Market Buy & Hold strategy as well.

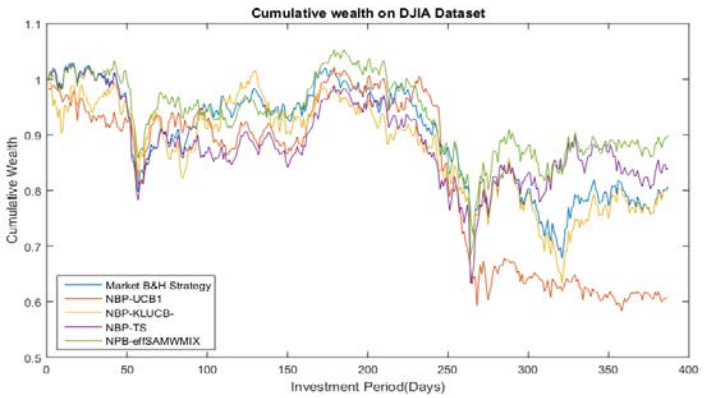

Figure 8: Cumulative wealth curves across the investment periods on DJIA dataset.

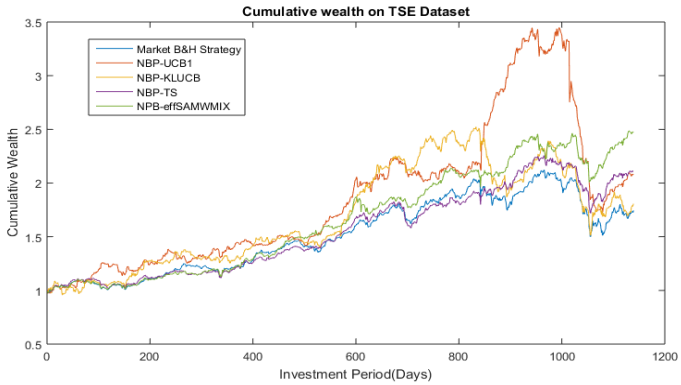

Figure 9: Cumulative wealth curves across the investment periods on TSE dataset.

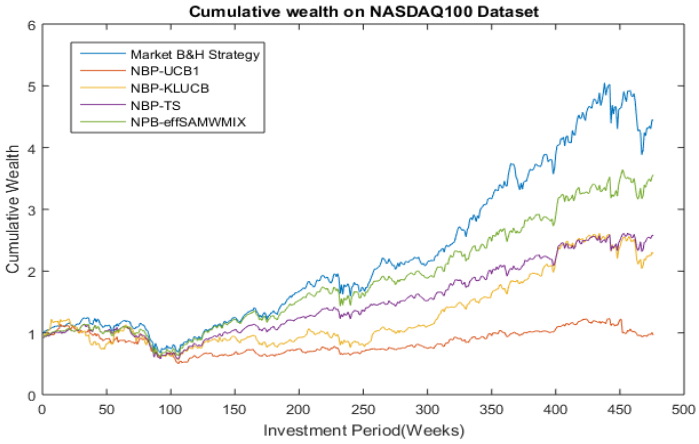

Figure 10: Cumulative wealth curves across the investment periods on NASDAQ100 dataset.

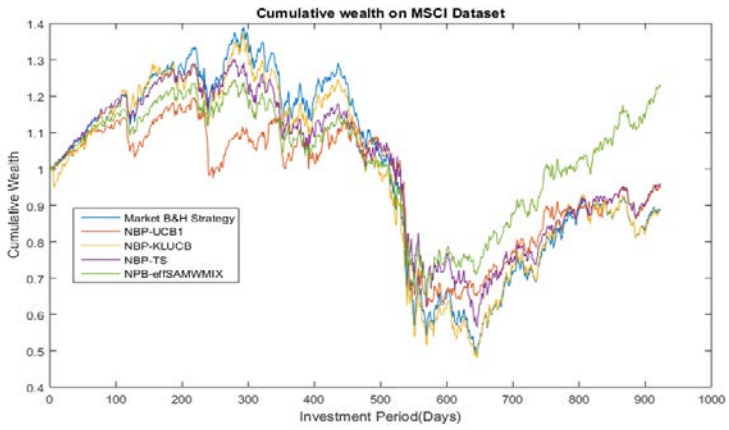

Figure 11: Cumulative wealth curves across the investment periods on MSCI dataset.

Figures 8-11 show the time series curves of the CW achieved over the investment periods. NBP-effSAMWMIX has performed comparatively with NBP-TS on DJIA and TSE datasets but has a distinguishably better performance on MSCI dataset. On the NASDAQ100 dataset, the Market Buy & Hold strategy is a clear winner from the early investment periods and none of the NBP algorithms could match its performance. Except for this one case, NBP-effSAMWMIX achieved the highest wealth level in all the datasets including the simulated GBM datasets.

Table 3: Terminal Cumulative Wealth on Benchmark Datasets.

	Cumulative Wealth($)				
	Market B&H Strategy	NBP-UCB1	NBP-KLUCB	NBP-TS	NPB-effSAMWMIX
DJIA	0.807213277	0.607567206	0.804983186	0.8409141	**0.898913847**
TSE	1.744441602	2.073821019	1.807159136	2.105704114	**2.468557904**
NASDAQ100	***4.436306307***	0.964635451	2.280205468	2.578221404	**3.559058958**
MSCI	0.890717018	0.945641382	0.886583483	0.955732983	**1.226302272**

5 Conclusion

In this work, we report the implementation of the effSAMWMIX inside of a Naïve Bandit Portfolio algorithm. effSAMWMIX which has a regret of $O(\log T)$ where T(we will put our working paper reference here) is shown to perform better than KL-UCB and Thompson Sampling on a few popular reward distributions. This work intends to exploit this advantage (of effSAMWMIX's performance) over competing SMAB algorithms reported in the literature. Along with NBP-effSAMWMIX, the NBP versions of KL-UCB and Thompson Sampling are evaluated for the first time in literature. We report the cumulative wealth data on both simulated and benchmark real-world datasets so as to concretely report the empirical performance of the proposed algorithm. The performance on NASDAQ100 dataset opens up the need to further asses the dataset so as to analyze why none of the NBP algorithms could beat the Market Buy & Hold strategy while they could in the rest of the cases. This could be because NBP does not consider asset correlations while making the decision. To further this work, we intend to do an Orthogonalization of the portfolios in order to remove the correlation and evaluate the performance of effSAMWMIX in such a scenario.

6 References

Abdulla, Mohammed Shahid, and Shalabh Bhatnagar, 2016. Multi-armed bandits based on a variant of Simulated Annealing, *Indian Journal of Pure and Applied Mathematics* 47, 195-212.

Agarwal, Amit, Elad Hazan, Satyen Kale, and Robert E Schapire, 2006. *Algorithms for portfolio management based on the newton method*(ACM).

Agrawal, Shipra, and Navin Goyal, 2012. *Analysis of Thompson Sampling for the Multi-armed Bandit Problem*.

Auer, Peter, Nicolo Cesa-Bianchi, and Paul Fischer, 2002. Finite-time analysis of the multiarmed bandit problem, *Machine Learning* 47, 235-256.

Auer, Peter, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire, 1995. *Gambling in a rigged casino: The adversarial multi-armed bandit problem*(IEEE).

Auer, Peter, and Ronald Ortner, 2010. UCB revisited: Improved regret bounds for the stochastic multi-armed bandit problem, *Periodica Mathematica Hungarica* 61, 55-65.

Borodin, Allan, and Ran El-Yaniv, 2005. *Online computation and competitive analysis*(cambridge university press).

Bruni, Renato, Francesco Cesarone, Andrea Scozzari, and Fabio Tardella, 2016. Real-world datasets for portfolio selection and solutions of some stochastic dominance portfolio models, *Data in Brief* 8, 858-862.

DeMiguel, Victor, Alberto Martín-Utrera, and Francisco J Nogales, 2015. Parameter uncertainty in multiperiod portfolio optimization with transaction costs, *Journal of Financial and Quantitative Analysis* 50, 1443-1471.

Fama, Eugene F, 1995. Random walks in stock market prices, *Financial Analysts Journal* 51, 75-80.

Fama, Eugene F, and Kenneth R French, 1992. The cross - section of expected stock returns, *The Journal of finance* 47, 427-465.

Fiat, Amos, 1998. Online Algorithms: The State of the Art (Lecture Notes in Computer Science).

Garivier, Aurélien, and Olivier Cappé, 2011. *The KL-UCB Algorithm for Bounded Stochastic Bandits and Beyond*.

Kaufmann, Emilie, Nathaniel Korda, and Rémi Munos, 2012. *Thompson sampling: An asymptotically optimal finite-time analysis*(Springer).

Ladde, GS, and Ling Wu, 2009. Development of modified geometric Brownian motion models by using stock price data and basic statistics, *Nonlinear Analysis: Theory, Methods & Applications* 71, e1203-e1208.

Lai, Tze Leung, and Herbert Robbins, 1985. Asymptotically efficient adaptive allocation rules, *Advances in applied mathematics* 6, 4-22.

Li, Bin, and Steven CH Hoi, 2012. On-line portfolio selection with moving average reversion, *arXiv preprint arXiv:1206.4626*.

Li, Bin, and Steven CH Hoi, 2014. Online portfolio selection: A survey, *ACM Computing Surveys (CSUR)* 46, 35.

Li, Bin, Doyen Sahoo, and Steven CH Hoi, 2016. OLPS: a toolbox for on-line portfolio selection, *Journal of Machine Learning Research* 17, 1-5.

Lisi, Francesco, 2011. Dicing with the market: randomized procedures for evaluation of mutual funds, *Quantitative Finance* 11, 163-172.

Marathe, Rahul R, and Sarah M Ryan, 2005. On the validity of the geometric Brownian motion assumption, *The Engineering Economist* 50, 159-192.

Markowitz, Harry, 1952. Portfolio selection, *The Journal of finance* 7, 77-91.

Mohr, Esther, Iftikhar Ahmad, and Günter Schmidt, 2014. Online algorithms for conversion problems: a survey, *Surveys in Operations Research and Management Science* 19, 87-104.

Mohr, Esther, and Günter Schmidt, 2013. How much is it worth to know the future in online conversion problems?, *Discrete Applied Mathematics* 161, 1546-1555.

Robbins, Herbert, 1985. *Some aspects of the sequential design of experiments*(Springer).

Rockafellar, R Tyrrell, and Stanislav Uryasev, 2000. Optimization of conditional value-at-risk, *Journal of risk* 2, 21-42.

Schmidt, Günter, Esther Mohr, and Mike Kersch, 2010. Experimental analysis of an online trading algorithm, *Electronic Notes in Discrete Mathematics* 36, 519-526.

Sharpe, William F, 1963. A simplified model for portfolio analysis, *Management science* 9, 277-293.

Sharpe, William F, 1966. Mutual fund performance, *The Journal of Business* 39, 119-138.

Shen, Weiwei, Jun Wang, Yu-Gang Jiang, and Hongyuan Zha, 2015. *Portfolio choices with orthogonal bandit learning*(AAAI Press).

Shen, Weiwei, Jun Wang, and Shiqian Ma, 2014. *Doubly Regularized Portfolio with Risk Minimization*.

Thompson, William R, 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples, *Biometrika* 25, 285-294.

Wilmott, P, 2000. *Quantitative Finance (vols. I and II)*(Wiley, Chichester, West Sussex, England).

Research Office

Indian Institute of Management Kozhikode

IIMK Campus P. O.,

Kozhikode, Kerala, India,

PIN - 673 570

Phone: +91-495-2809238

Email: research@iimk.ac.in

Web: https://iimk.ac.in/faculty/publicationmenu.php