Meta Optimization and its Application to Portfolio Selection

Puja Das
Dept of Computer Science & Engg
Univ of Minnesota, Twin Cities
pdas@cs.umn.edu

Arindam Banerjee
Dept of Computer Science & Engg
Univ of Minnesota, Twin Cities
banerjee@cs.umn.edu

ABSTRACT

Several data mining algorithms use iterative optimization methods for learning predictive models. It is not easy to determine upfront which optimization method will perform best or converge fast for such tasks. In this paper, we analyze Meta Algorithms (MAs) which work by adaptively combining iterates from a pool of base optimization algorithms. We show that the performance of MAs are competitive with the best convex combination of the iterates from the base algorithms for online as well as batch convex optimization problems. We illustrate the effectiveness of MAs on the problem of portfolio selection in the stock market and use several existing ideas for portfolio selection as base algorithms. Using daily S&P500 data for the past 21 years and a benchmark NYSE dataset, we show that MAs outperform existing portfolio selection algorithms with provable guarantees by several orders of magnitude, and match the performance of the best heuristics in the pool.

1. INTRODUCTION

Several data mining algorithms use iterative update methods for learning predictive models. Typically, there are several choices for iterative update methods including gradient based or Newton step based optimization routines, stochastic gradient descent algorithms, domain specific methods, evolutionary and genetic algorithms, or plain heuristics. It is not easy to determine upfront which method will converge fast or perform the best.

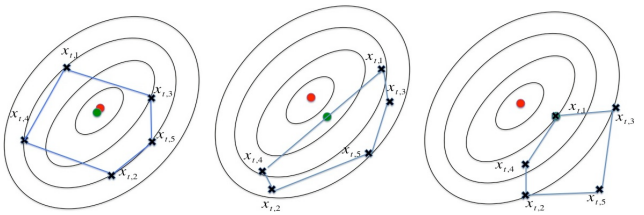
While multiple iterative update methods can be run in an embarrassingly parallel manner, it is unclear if iterates from multiple algorithms for the same problem can be meaningfully combined to guarantee good optimization performance. Ideally, one would like the combined iterates to outperform the best algorithm in the pool, noting that the best algorithm may be different for different problem settings and domains. Such a desideratum is related to ensemble methods for prediction problems, where one expects the ensemble prediction to outperform the single best predictor in the pool [13, 6, 7]. In this paper, we investigate a related question in the contest of iterative optimization: Can iterates from multiple iterative update algorithms for the same problem be combined in a way so as to outperform the single best algorithm in the pool in terms of optimization performance? Related questions have been investigated in certain other contexts, including online learning [9, 22] and genetic programming [24, 27].

The setting we consider is fairly general: Given a canonical convex optimization problem $\min_{x \in P} \ \phi(x)$ and a set of k different base algorithms which generate an iterate $x_{t,h} \in P, 1 \leq h \leq k$ in every iteration, can we form an adaptive convex combination of the iterates $x_t^{w_t} = \sum_{h=1}^{k} w_{t,h} x_{t,h}$ whose performance is at least as good as the best single algorithm. There is no requirement from the base algorithms other than producing a feasible $x_{t,h} \in P$ in every iteration. In particular, the base algorithms need not guarantee monotonic improvements in the objective function, and may be based on a heuristic without any guarantees. To make our analysis general, we even allow the convex function to change over time. Using advances in online learning and online convex optimization [8, 22, 20, 9, 15], we develop two algorithms for adaptively combining iterates which are guaranteed to be as good as the best convex combination of iterates, and hence the best algorithm.

We extensively evaluate the proposed methodology in an important problem in financial data mining—portfolio selection [23, 10, 16, 1]. The goal is to adaptively update a portfolio over a set of stocks so that the returns over time are maximized. The problem can be posed as an online convex optimization problem, where the convex function gets determined by market movements on each day [1, 16, 10]. Due to its importance, the portfolio selection problem has been widely studied for six decades [23, 19, 10, 9], and numerous algorithms and heuristics exist on how to pick the next days portfolio which forms the iterate $x_{t,h}$ in our setting. We use a pool of these existing algorithms for portfolio selection, and focus on creating a portfolio by adaptively combining the portfolios suggested by the base algorithms. Through our analysis and algorithms, we establish theoretical results and illustrate strong empirical performance. In particular, we show that the meta algorithms for portfolio selection will be universal, i.e., competitive with the best constant rebalanced portfolio (CRP) chosen in hindsight [10, 18, 4], if any base algorithm in the pool is universal. Note that universal portfolios are guaranteed to be as good as the best stock even in adversarial settings. Our experiments show that the meta algorithms outperform all existing universal algorithms by orders of magnitude, by suitably leveraging good heuristics in the pool. For example, trading on S&P500 stocks over the past 21 years (1990-2010), the meta algorithms multiply the starting wealth by 10^3 times even with two major financial meltdowns. Further, the proposed meta algorithms clearly outperform other simplistic approaches to combining portfolios.

The rest of the paper is organized as follows. We present a general framework and two algorithms for meta optimization in Section 2. In Section 3, we specialize the analysis and algorithms to

KDD'11, August 21–24, 2011, San Diego, California, USA.



(a) (b) (c)

Figure 1: The best convex combination x_t^w of the iterates from the base algorithms is always better than individual iterates $x_{t,h}$(the red dot is the global minimum and the green dot is the best point in the convex hull of iterates): (a) x_t^w achieves the global minimum, (b) x_t^w is on an edge of the hull, and (c) x_t^w overlaps with the best iterate.

the problem of portfolio selection. We present comprehensive experimental results in Section 4, and conclude in Section 5.

2. ONLINE META OPTIMIZATION

Consider the following generic convex optimization problem which shows up while building models for a variety of data mining tasks [25]:

$$\min_{x \in P} \phi(x) \,, \tag{1}$$

where ϕ is a convex function and $P \in \mathbb{R}^d$ determines the convex feasible set. For the meta-optimization setting, we assume access to k different iterative algorithms $A_1, \ldots, A_k$, referred to as *base algorithms*, which attempt to solve the above problem. In particular, A_h is assumed to generate a feasible $x_{t,h} \in P$ at every iteration. The analysis we present does not depend on any other properties of the base algorithms or the iterates. The iterates may be coming from a iterative convex optimization routines based on gradient or Newton methods, from domain specific heuristics, or even entirely arbitrary guesses. The proposed meta-algorithm picks a suitable iterate from the convex hull of the iterates at any time, given by:

$$\text{Co}(X_t) = \left\{ X_t w = \sum_{h=1}^{k} w_h x_{t,h} \,\middle|\, \sum_{h=1}^{k} w_h = 1, w_h \geq 0 \right\} \,,$$

where $X_t = [x_{t,1} \cdots x_{t,k}] \in \mathbb{R}^{d \times k}$ is the matrix of iterates. Let Δ_k denote the k-dimensional simplex. Then, it is easy to see that the best point $x_t^w = X_t w = \sum_h w_h x_{t,h} \in \text{Co}(X_t)$ will always achieve a lower (better) objective function value than any of the individual iterates, i.e.,

$$\min_{w \in \Delta_k} \phi \left(\sum_{h=1}^{k} w_h x_{t,h} \right) \leq \phi(x_{t,h}), \;\; \forall h \,.$$

Figure 1 shows examples to illustrate the above point. In Figure 1(a), the best point in the convex hull of the iterates achieves the global minimum of the function; in Figure 1(b), it is nearest to the global minimum; and in Figure 1(c), the best point in the convex hull is an iterate itself, i.e., a corner of the hull.

In general, the best point $x_t^w = X_t w \in \text{Co}(X_t)$ inside the convex hull or equivalently the best convex combination $w \in \Delta_k$ cannot be obtained in closed form. One can design optimization algorithms to find the best point inside the convex hull. Note that such computations have to be repeated at every iteration, since corners of the hull, determined by X_t, changes in every iteration. In this section, we develop algorithms which adaptively pick $w_t \in \Delta_k$ based on X_{t-1}, and show that the iterates $x_t^{w_t} = X_t w_t = \sum_h w_{t,h} x_{t,h}$ of the meta-algorithm are competitive with any fixed convex combination $w \in \Delta_k$ used over iterations, i.e., $\forall w \in \Delta_k$ we have

$$\sum_{t=1}^{T} \phi(X_t w_t) \leq \sum_{t=1}^{T} \phi\left(X_t w\right) + o(T) \,. \tag{2}$$

In particular, if any $w^* \in \Delta_k$ achieves the global minimum, the adaptive approach will find the global minimum as well. Indeed, instead of simply being competitive with the single best iterate, the adaptive $x_t^{w_t}$ will be competitive with any convex combinations of them (Figure 1). To present our analysis in its full generality, we consider the online convex optimization (OCO) setting [28], where the convex function itself can change over time. We denote the convex function at time t to be ϕ_t. Note that we can recover the batch case analysis for a fixed ϕ as a special case by simply setting $\phi_t = \phi, \forall t$. In the OCO setting, we intend to get a set of adaptive iterates $x_t^{w_t}$ such that the following form of *regret bounds* are satisfied:

$$\sum_{t=1}^{T} \phi_t(X_t w_t) \leq \min_{w \in \Delta_k} \sum_{t=1}^{T} \phi_t\left(X_t w_t\right) + o(T) \,. \tag{3}$$

Algorithm 1 Online Gradient Update (OGU) for Meta Optimization

1: Initialize $w_{1,h} = \frac{1}{k}, h = 1, \ldots, k$
2: For $t = 1, \ldots, T$
3: Receive $X_t = [x_{t,1} \cdots x_{t,k}]$ from base algorithms
4: Compute $x_t^{w_t} = \sum_{h=1}^{k} w_{t,h} x_{t,h}$
5: Receive convex function ϕ_t from nature
6: Update distribution

$$w_{t+1,h} = w_{t,h} \exp(-\eta \ell_t(h))/Z_t$$

where $\ell_t(h)$ is as in (6) and Z_t is the partition function.

2.1 Online Gradient Updates

Our meta algorithm and analysis for Online Gradient Updates (OGU) involves suitably reducing the Online Meta Optimization (OMO) problem to an online learning problem over k experts [22, 9], where each expert corresponds to a corner for meta-optimization. We start by recalling a standard result from the online learning literature [22, 12, 3]:

Lemma 1 *Let $\ell_t \in [0,1]^k, t = 1, \ldots, T$, be an arbitrary sequence of loss vectors over the k experts. If one maintains an adaptive distribution over the experts using multiplicative updates given by $p_{t+1}(h) = p_t(h) \exp(-\eta \ell_t(h))/Z_t$, where $\eta > 0$ and Z_t is the partition function, then for any $w \in \Delta_k$, the following inequality holds:*

$$\sum_{t=1}^{T} p_t^T \ell_t \leq \frac{\eta \sum_{t=1}^{T} w^T \ell_t + \log k}{1 - \exp(-\eta)} \,. \tag{4}$$

Variants of the above result form the basis of much work in online learning, boosting, game theory, and numerous other developments in the past two decades [22, 12, 11, 2, 3, 9]. We now outline a transformation of the OMO problem to the above online learning setting.

For our analysis, we assume that the sequence of convex functions ϕ_t can be arbitrary, but satisfies $\|\nabla \phi_t(x)\|_\infty \leq g_\infty$ for $x \in P$. Further, we assume $x \in P$ satisfies $\|x\|_1 \leq c$. For the portfolio

selection application in Section 3, we will obtain specific values for g_∞ and c. Let

$$f_t(w) = \phi_t\left(X_t w\right) \ . \tag{5}$$

Since $\phi_t : P \mapsto \mathbb{R}$, where $P \subseteq \mathbb{R}^d$, is a convex function, the function $f_t : \Delta_k \mapsto \mathbb{R}$ is also convex. To see this, first note that the Hessian $\nabla^2 f_t(w) = X_t^T \nabla^2 \phi_t(X_t w) X_t$. Since ϕ_t is convex, $\nabla^2 \phi(X_t w)$ is positive semi-definite. Hence, $\nabla^2 f_t(w)$ is positive semi-definite, implying convexity of f_t. Define loss vector

$$\ell_t = \frac{1}{2}\left(\frac{\nabla f_t(w_t)}{c g_\infty} + e\right) \in \mathbb{R}^k \ , \tag{6}$$

where e is the all ones vector. Based on this definition of loss, Algorithm 1 presents an adaptive algorithm for Online gradient update for meta optimization. We establish the following regret bound for OGO for this algorithm:

Theorem 1 *For any sequence of convex functions ϕ_t such that $\|\nabla \phi_t(x)\|_\infty \le g_\infty$, and any sequence of iterates $X_t = [x_{t,1} \ \cdots \ \cdots \ x_{t,k}]$ such that $\|x_{t,h}\|_1 \le c$, for $\eta = \log\left(1 + \sqrt{\frac{2\log k}{T}}\right)$ in Algorithm 1, we have*

$$\begin{aligned} \sum_{t=1}^T f_t(w_t) - \min_{w \in \Delta_k} \sum_{t=1}^T f_t(w) \\ \le 2 c g_\infty \left(\sqrt{2T \log k} + \log k\right) \ . \end{aligned} \tag{7}$$

PROOF. Since $\nabla f_t(w_t) = X_t^T \nabla \phi_t(X_t w_t)$, $\|\nabla f_t(w_t)\|_\infty = \max_h |x_{t,h}^T \nabla \phi_t(x_t^{w_t})|$. From Hölder's inequality [26, 14, 21],

$$|x_{t,h}^T \nabla \phi_t(X_t w_t)| \le \|x_{t,h}\|_1 \|\nabla \phi_t(X_t w_t)\|_\infty \le c g_\infty \ .$$

Hence $\frac{\nabla f_t(w_t)}{c g_\infty} \in [-1, 1]^k$, so that $\ell_t \in [0, 1]^k$. From Lemma 1, Algorithm 1 will satisfy (4). Let $\epsilon = 1 - \exp(-\eta)$ so that from Lemma 1 we have

$$\sum_{t=1}^T w_t^T \ell_t - \sum_{t=1}^T w^T \ell_t \le \epsilon T + \frac{1}{\epsilon} \log k \ ,$$

where we have used $\sum_{t=1}^t w^T \ell_t \le T$. Choosing $\epsilon = \frac{\sqrt{2\log k}}{\sqrt{2\log k} + \sqrt{T}}$, a direct calculation shows

$$\sum_{t=1}^T \ell_t^T (w_t - w) \le \sqrt{2T \log k} + \log k \ . \tag{8}$$

Now, since f_t is convex, we have

$$f_t(w_t) - f_t(w) \le \nabla f_t(w_t)^T (w_t - w) = 2 c g_\infty \ell_t^T (w_t - w) \ ,$$

where the last equality follows since $e^T(w_t - w) = 0$ as $w_t, w \in \Delta_k$. Adding over all t and using (8), we have

$$\begin{aligned} \sum_{t=1}^T f_t(w_t) - \sum_{t=1}^T f_t(w) &\le 2 c g_\infty \sum_{t=1}^T \ell_t^T (w_t - w) \\ &\le 2 c g_\infty \left(\sqrt{2T \log k} + \log k\right) \ . \end{aligned}$$

Noting that the above inequality holds for any $w \in \Delta_k$ completes the proof. □

Since $2 c g_\infty \left(\sqrt{2T \log k} + \log k\right) = o(T)$, we have a desired form of the bound. Further, assuming $\phi_t = \phi$ gives the corresponding bound for the batch optimization case.

Algorithm 2 Online Newton Update (ONU) for Meta Optimization

1: Initialize $w_1 \in \Delta_k$
2: Let $\beta = \min\left\{\frac{1}{8 c g_\infty}, \alpha\right\}$
3: For $t = 1, \ldots, T$
4: Receive $X_t = [x_{t,1} \ \cdots \ x_{t,k}]$ from base algorithms
5: Compute $x_t^{w_t} = \sum_{h=1}^k w_{t,h} x_{t,h}$
6: Receive convex function ϕ_t from nature
7: Update distribution

$$w_{t+1,h} = \prod_{\Delta_k}^{A_t}\left(w_t - \frac{2}{\beta} A_t^{-1} \nabla f_t\right) \ ,$$

where A_t and $\prod_{\Delta_k}^{A_t}$ are as in (9) and (10).

2.2 Online Newton Updates

Our analysis for Online Newton Updates (ONU) build on recent advances in Online Convex Optimization (OCO) [15, 1]. The analysis of ONU differs from the standard analysis of online Newton step [15] due to two reasons: first, our analysis focuses on the derived convex function $f_t : \Delta_k \mapsto \mathbb{R}$ instead of the original convex function $\phi_t : P \mapsto \mathbb{R}$, and second, our bounds are based on the L_∞ norm of ϕ_t instead of the L_2 norm, which can be substantially larger for high-dimensional problems.

Following [15], we consider convex functions ϕ_t which satisfy the α-exp-concavity property: there is a $\alpha > 0$ such that for $x \in P$, $\exp(-\alpha \phi_t(x))$ is a concave function . Note that α-exp-concave functions ϕ_t are more general than ones which have bounded gradients and Hessians which are strictly bounded away from 0, i.e., $\nabla^2 \phi_t \succeq H \mathbb{I}$ for some constant $H > 0$. As before, we assume that L_∞ norm of the gradient of ϕ_t are bounded above, i.e., $\|\nabla \phi_t\|_\infty \le g_\infty$.

With these assumptions, Algorithm 2 presents the Online Newton Update (ONU) algorithm for Online Meta Optimization [15]. In essence, the algorithm takes a Newton-like step from the current iterate w_t, and then projects the vector to the feasible set Δ_k to obtain w_{t+1}. Note that the algorithm does not use the actual Hessian of f_t, but a matrix based on the outer product of the gradients defined as:

$$A_t = \sum_{\tau=1}^t \nabla f_t \nabla f_t^T + \epsilon \mathbb{I} \ , \tag{9}$$

where $\epsilon = \frac{k}{\beta^2 c^2}$. Further $\prod_{\Delta_k}^{A_t}$ is the projection onto Δ_k using the Mahalanobis distance induced by A_t, i.e.,

$$\prod_{\Delta_k}^{A_t}(\tilde{w}) = \operatorname*{argmin}_{w \in \Delta_k} (w - \tilde{w})^T A_t^{-1} (w - \tilde{w}) \ . \tag{10}$$

We start our analysis by showing that if ϕ_t is α-exp-concave for $x \in P$, then f_t is α-exp-concave for $w \in \Delta_k$ for the same (set of) α.

Lemma 2 *If ϕ_t is α-exp-concave for some $\alpha > 0$, then f_t as defined in (5) is also α-exp-concave.*

PROOF. Let $h_t(w) = \exp(-\alpha f_t(w))$. The Hessian is given by

$$\begin{aligned} \nabla^2 h_t(w) &= [\alpha^2 (\nabla f_t)(\nabla f_t)^T - \alpha \nabla^2 f_t] h_t(w) \\ &= X_t^T [\alpha^2 (\nabla \phi_t)(\nabla \phi_t)^T - \alpha \nabla^2 \phi_t] X_t h_t(w) \end{aligned}$$

Let $\psi_t(x) = \exp(-\alpha \phi_t(x))$. Since ϕ_t is α-exp-concave, the Hessian $\nabla^2 \psi_t \preceq 0$, so that

$$[\alpha^2 (\nabla \phi_t)(\nabla \phi_t)^T - \alpha \nabla^2 \phi_t] \psi_t(x) \preceq 0 \ .$$

Let $B_t = [\alpha^2(\nabla\phi_t)(\nabla\phi_t)^T - \alpha\nabla^2\phi_t]$. Since $\psi_t(x) \geq 0$, we have

$$B_t \preceq 0 \quad \Rightarrow \quad X_t^T B_t X_t \preceq 0 \,,$$

so that $\nabla^2 h_t \preceq 0$ since $h_t(w) \geq 0$, implying h_t is α-exp-concave. $\square$

We now establish a result, similar to Lemma 3 in [15], but using the L_∞ bound g_∞ and the fact that $\|x\|_1 \leq c$ for $x \in P$.

Lemma 3 *For $\beta \leq \min\{\frac{1}{8cg_\infty}, \alpha\}$, for any $w, w_t \in \Delta_k$, we have*

$$f_t(w) \geq f_t(w_t) + \nabla f_t(w_t)^T(w - w_t) + \frac{\beta}{4}(w - w_t)^T \nabla f_t(w_t) \nabla f_t(w_t)^T (w - w_t) \,. \tag{11}$$

PROOF. Since $\beta \leq \alpha$, following the proof of Lemma 3 in [15] we have

$$f_t(w) \geq f_t(w_t) - \frac{1}{\beta}\log[1 - \beta\nabla f_t(w_t)^T(w - w_t)] \,.$$

Now, by Hölder's inequality,

$$|\beta\nabla f_t(w_t)^T(w - w_t)| \leq \beta\|\nabla f_t(w_t)\|_\infty \|w - w_t\|_1 \leq 2\beta c g_\infty \leq \frac{1}{4} \,.$$

Since $-\log(1 - z) \geq z + \frac{1}{4}z^2$ for $|z| \leq \frac{1}{4}$, using it for $z = \beta\nabla f_t(w_t)^T(w - w_t)$ completes the proof. $\square$

We now present the main result for ONU:

Theorem 2 *For any sequence of α-exp-concave functions ϕ_t such that $\|\nabla\phi_t\|_\infty \leq g_\infty$ for $x \in P$ where $\|x\|_1 \leq c$, for $T \geq 2a$ where $a = \frac{32 g_\infty}{c^2}$, we have the following regret bound:*

$$\sum_{t=1}^T f_t(w_t) - \min_{w \in \Delta_k} \sum_{t=1}^T f_t(w) \leq k\left(8cg_\infty + \frac{1}{\alpha}\right)\log\frac{eT}{a} \,. \tag{12}$$

PROOF. Using Lemma 3 and using the proof of Theorem 2 in [15], we have

$$\sum_{t=1}^T R_t \leq \frac{1}{\beta}\sum_{t=1}^T \nabla_t^T A_t^{-1}\nabla_t + \frac{\beta}{4}(w_t - w)^T(A_1 - \nabla_1\nabla_1^T)(w_1 - w) \,,$$

where $R_t = f_t(w_t) - f_t(w)$ for any $w \in \Delta_k$, $\nabla_t = \nabla f_t$, and $A_t = \sum_{\tau=1}^t \nabla_\tau\nabla_\tau^T + \epsilon\mathbb{I}$ as in (9). Since $A_1 - \nabla_1\nabla_1^T = \epsilon\mathbb{I}$, $\|w_1 - w\|_2^2 \leq 4c^2$, and $\epsilon = \frac{k}{\beta^2 c^2}$, we have

$$\begin{aligned}\sum_{t=1}^T R_t &\leq \frac{1}{\beta}\sum_{t=1}^T \nabla_t^T A_t^{-1}\nabla_t + \frac{\beta}{4}\epsilon\|w_1 - w\|_2^2 \\ &\leq \frac{1}{\beta}\sum_{t=1}^T \nabla_t^T A_t^{-1}\nabla_t + \frac{k}{\beta} \,.\end{aligned}$$

Since $\|\nabla f_t\| \leq \sqrt{k}\|\nabla f_t\|_\infty \leq \sqrt{k}cg_\infty$, from Lemma 11 in [15], we have

$$\sum_{t=1}^T \nabla_t^T A_t^{-1}\nabla_t \leq k\log\left(\frac{kc^2 g_\infty^2 T}{\epsilon} + 1\right) \leq k\log\left(\frac{T}{2a} + 1\right) \,,$$

where we have used $\epsilon = \frac{k}{\beta^2 c^2}, \beta \leq \frac{1}{8cg_\infty}$, and $a = \frac{32 g_\infty}{c^2}$. For $T \geq 2a$, $\frac{T}{2a} + 1 \leq \frac{T}{a}$. Plugging everything back, we have

$$\sum_{t=1}^T R_t \leq \frac{k}{\beta}\left(\log\frac{T}{a} + 1\right) = \frac{k}{\beta}\log\frac{eT}{a} \,.$$

Since $\beta = \min\{\frac{1}{8cg_\infty}, \alpha\}$, we have

$$\frac{1}{\beta} = \max\left\{8cg_\infty, \frac{1}{\alpha}\right\} \leq 8cg_\infty + \frac{1}{\alpha} \,.$$

Plugging this upper bound back completes the proof. $\square$

3. META OPTIMIZATION FOR PORTFOLIO SELECTION

We consider a stock market consisting of n stocks $\{s_1, \ldots, s_n\}$ over a span of T periods. For ease of exposition, we will consider a period to be a day, but the analysis presented in the paper holds for any valid definition of a 'period,' such as an hour or a month. Let $r_t(i)$ denote the *price relative* of stock s_i in day t, i.e., the multiplicative factor by which the price of s_i changes in day t. Hence, $r_t(i) > 1$ implies a gain, $r_t(i) < 1$ implies a loss, and $r_t(i) = 1$ implies the price remained unchanged. We assume $r_t(i) > 0$ for all i, t. Let $r_t = \langle r_t(1), \ldots, r_t(n)\rangle$ denote the vector of price relatives for day t, and let $r_{1:t}$ denote the collection of such price relative vectors upto and including day t. A portfolio $x_t = \langle x_t(1), \ldots, x_t(n)\rangle$ on day t can be viewed as a probability distribution over the stocks that prescribes investing $x_t(i)$ fraction of the current wealth in stock $s_t(i)$. Note that the portfolio x_t has to be decided before knowing r_t which will be revealed only at the end of the day. The multiplicative gain in wealth at the end of day t, is then simply $r_t^T x_t = \sum_{i=1}^n r_t(i)x_t(i)$. Given a sequence of price relatives $r_{1:t-1} = \{r_1, \ldots, r_{t-1}\}$ upto day $(t-1)$, the sequential portfolio selection problem in day t is to determine a portfolio x_t based on past performance of the stocks. At the end of day t, r_t is revealed and the actual performance of x_t gets determined by $r_t^T x_t$. Over a period of T days, for a sequence of portfolios $x_{1:T} = \{x_1, \ldots, x_t\}$, the multiplicative gain in wealth is then

$$S(x_{1:T}, r_{1:T}) = \prod_{t=1}^T \left(r_t^T x_t\right) \,. \tag{13}$$

The above problem can be viewed as an Online Convex Optimization (OCO), where the convex function $\phi_t(x_t) = -\log(r_t^T x_t)$, and the cumulative loss over T iterations is

$$\sum_{t=1}^T \phi_t(x_t) = -\sum_{t=1}^T \log(r_t^T x_t) = -\log S(x_{1:T}, r_{1:T}) \,. \tag{14}$$

There are numerous algorithms in the literature for picking the portfolio x_t on a given day based on past information $r_{1:(t-1)}$ [10, 16, 9, 1, 5]. Instead of proposing new algorithms for the task, we focus on meta optimization to combine the portfolios from a pool of base algorithms from the literature. We now specialize the general case results and algorithms of Section 2 to the task of portfolio selection.

Consider k base algorithms $\{A_1, \ldots, A_k\}$ for portfolio selection where algorithm A_h generates a portfolio $x_{t,h} \in \Delta_n$ based on the past information $r_{1:(t-1)}$. Recall that our analysis does not impose any other constraints on the base algorithms and so they can be based on theoretically well grounded ideas [10, 16, 1] or good heuristics [5]. Given the set of base portfolios $X_t = [x_{t,1} \cdots x_{t,k}]$, the goal of the meta algorithm is to choose $w_t \in \Delta_k$ to construct the portfolio $x_t^{w_t} = X_t w_t$ and subsequently incur loss $f_t(w_t) = \phi_t(x_t^{w_t}) = -\log(r_t^T x_t^{w_t}) = -\log(r_t^T X_t w_t)$. Since a portfolio $x \in \Delta_n$, we have $c = \|x\|_1 = 1$. Among all price relatives over all stocks, let $r_{\min} = \min_{i,t} r_t(i) > 0$ and let $r_{\max} = \max_{i,t} r_t(i)$. Since $\nabla\phi_t(x) = -\frac{r_t}{r_t^T x}$, $\|\nabla\phi_t(x)\|_\infty \leq \frac{r_{\max}}{r_{\min}} = g_\infty$. For convenience, we use $\bar{u} = \frac{r_{\max}}{r_{\min}}$. For our subse-

quent analysis, we note that

$$\nabla f_t(w_t) = -\frac{X_t^T r_t}{r_t^T X_t w_t} \ . \tag{15}$$

Gradient Updates: Since $\nabla\phi_t(x)$ for portfolio selection is a positive vector, one can define the loss vector for OGU in Algorithm 1 as follows:

$$\ell_t = \frac{\nabla f_t(w_t)}{c g_\infty} = -\frac{1}{2\bar{u}} \frac{X_t^T r_t}{r_t^T X_t w_t} + e \ . \tag{16}$$

With this modification, the OGU in Algorithm 1 has the following guarantee:

Corollary 1 *For any sequence of price relatives $r_{1:T}$ and any sequence of base portfolios $X_{1:T}$, the log-wealth accumulated by Algorithm 1 choosing adaptive w_t satisfies the following regret bound:*

$$\begin{aligned} \max_{w\in\Delta_k} \sum_{t=1}^{T} \log(r_t^T X_t w) - \sum_{t=1}^{T} \log(r_t^T X_t w_t) \\ \leq \bar{u}\left(\sqrt{2T\log k} + \log k\right) \ . \end{aligned} \tag{17}$$

The proof follows from a direct application of Theorem 1. We briefly discuss the implication of the fact that the wealth accumulated by the adaptive meta algorithm will be competitive with any fixed combination strategy chosen in hindsight. If one of the base algorithms is *universal* [10, 16, 4, 17, 9], i.e., competitive with best constant rebalanced portfolio (CRP) [10, 4] in hindsight so that

$$\max_{x\in\Delta_n} \sum_{t=1}^{T} \log(r_t^T x) - \sum_{t=1}^{T} \log(r_t^T x_t) = o(T) \ , \tag{18}$$

then our meta algorithm will also be universal. Also, since the best CRP would outperform the best stock, having an universal algorithm in the pool is sufficient to ensure the meta algorithm will be competitive with the single best stock in hindsight. More generally, the meta algorithm will be competitive with the best convex combination of the base algorithms, which is guaranteed to be better than the best base algorithm in the pool (Figure 1).

Newton Updates: We start our analysis with the following result:

Lemma 4 *$\phi_t(x) = -\log(r_t^T x)$ is a α-exp-concave function for $\alpha \in (0, 1]$.*

PROOF. Let $\psi_t(x) = \exp(-\alpha\phi_t(x)) = (r_t^T x)^\alpha$. A direct calculation shows the Hessian to be

$$\nabla^2\psi_t(x) = \alpha(\alpha-1)\frac{r_t r_t^T}{r_t^T x}\psi_t(x) \ ,$$

which is negative semi-definite for $\alpha > 0$ if $\alpha \in (0, 1]$. □

As a result, Algorithm 2 can be applied as a meta algorithm for the portfolio selection problem. As before, $c = 1, g_\infty = \bar{u}$. Choosing $\alpha = 1, \beta = \min\{\frac{1}{8\bar{u}}, \alpha\} = \frac{1}{8\bar{u}}$ since $\bar{u} = \frac{r_{\max}}{r_{\min}} \geq 1$. Hence, $\frac{1}{\beta} = 8\bar{u}$. Further, $a = \frac{32 g_\infty}{c^2} = 32\bar{u}$, so that $\frac{a}{e} \leq 12\bar{u}$. Using the above values in Algorithm 2, from Theorem 2 we have the following result:

Corollary 2 *For any sequence of price relatives $r_{1:T}$ and any sequence of base portfolios $X_{1:t}$, the log-wealth accumulated by Algorithm 2 choosing adaptive w_t satisfies the following regret bound:*

$$\max_{w\in\Delta_k} \sum_{t=1}^{T} \log(r_t^T X_t w) - \sum_{t=1}^{T} \log(r_t^T X_t w_t) \leq 8k\bar{u}\log\frac{T}{12\bar{u}} \ . \tag{19}$$

As before, the proof follows from a direct application of Theorem 2. The bound has the same optimality properties as discussed in the context of OGU above. In fact, the worst case regret of ONU grows as $O(\log T)$ as opposed to $O(\sqrt{T})$ for OGU. The bound for OGU can in fact be sharpened in this setting by suitably modifying the OGU algorithm and analysis using the fact that the Hessian $\nabla^2 f_t$ is bounded away from 0 under the assumption $\min_{i,t} r_{i,t} > 0$.

4. EXPERIMENTAL RESULTS

We conducted extensive experiments on two financial data-sets to establish how effective Online Meta Optimization can be carried out by OGU and ONU. In this section, we describe the datasets that were chosen for the experiments, the algorithms, the parameter choices and most importantly the results of our experiments.

Datasets: The experiments were conducted on two major datasets: the New York Stock Exchange dataset (NYSE) [10] and a Standard & Poor's 500 (S&P 500) dataset. The NYSE dataset consists of 36 stocks with data accumulated over a period of 22 years from July 3, 1962 to Dec 31 1984. The dataset captures the bear market that lasted between January 1973 and December 1974. However, all of the 36 stocks increase in value in the 22-year run.

The S&P500 dataset that we used for our experiments consists of 263 stocks which were present in the S&P500 index in December 2010 and were alive since January 1990. This period of 21 years from 1990 to 2010 covers bear and bull markets of recent times.

Methodology: We ran a pool of base portfolio selection algorithms and the Meta Algorithms on the datasets (NYSE and S&P500). For our experiments, this pool included universal and non-universal algorithms. We start by briefly describing the base algorithms and the Meta Algorithms.

4.1 Base Algorithms

Of the base algorithms that we used for our experiments UP, EG and ONS are *universal* while Anticor and its variant are heuristics.

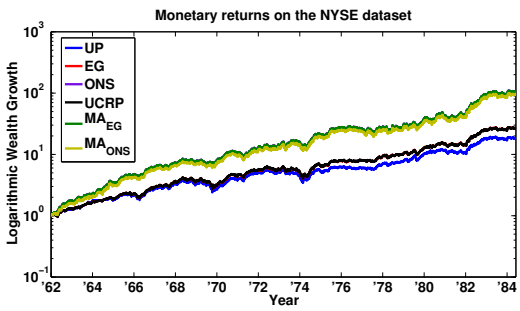
Universal Portfolios (UP): The key idea behind Cover's [10] UP is to maintain a distribution over all Constant Rebalanced Portfolios (CRPs) and perform a Bayesian update after observing every r_t. Each CRP q is a distribution over n stocks and hence lies in the n-simplex, one uses a distribution $\mu(q)$ over the n-simplex. The universal portfolio x_t is defined as:

$$x_t(i) = \frac{\int_q q(i) S_{t-1}(q, r_{1:t-1})\mu(q)dq}{\int_q S_{t-1}(q, r_{1:t-1})\mu(q)dq} \ . \tag{20}$$

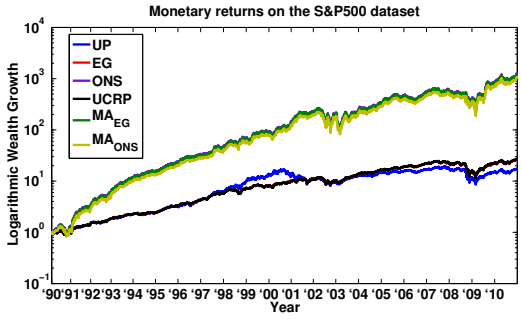
UP has a regret of $O(\log T)$ with respect to the best CRP in hindsight. However, the updates for UP are computationally prohibitive.

Exponentiated Gradient (EG): Exponentiated Gradient (EG) [16] scales linearly with the number of stocks but is weaker in regret than UP. The EG investment strategy was introduced and analyzed by [16]. At the start of day t, the algorithm computes its new portfolio vector x_t such that it stays close to x_{t-1} and does well on the price relatives r_{t-1} for the previous day. The updated portfolio turns out to be

$$x_t(i) = \frac{x_{t-1}(i)\exp(\eta r_{t-1}(i)/x_{t-1}^T r_{t-1})}{\sum_{i'=1}^{n} x_{t-1}(i')\exp(\eta r_{t-1}(i')/x_{t-1}^T r_{t-1})} \ . \tag{21}$$

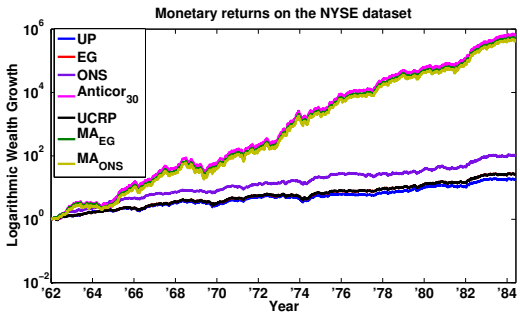


(a) Monetary returns on NYSE.

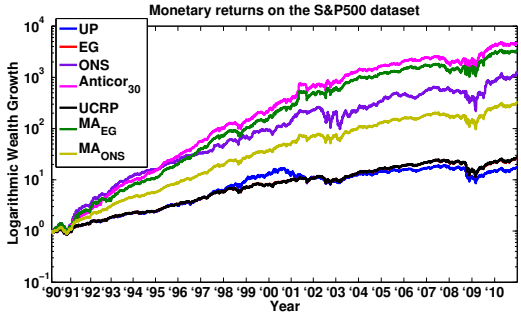


(b) Monetary returns on S&P500.

Figure 2: Monetary returns of the Meta Algorithms, MA $_{EG}$ and MA $_{ONS}$ for $1 investment, is competitive with the best performing base algorithm ONS in this case(best viewed in color).



(a) Monetary returns on NYSE.



(b) Monetary returns on S&P500.

Figure 3: Monetary returns of the meta algorithms(MA$_{EG}$, MA$_{ONS}$) when Anticor$_{30}$ is added to the pool of base algorithms. Anticor$_{30}$ performs best and particularly MA$_{EG}$ is able to track Anticor$_{30}$ (best viewed in color).

where $\eta > 0$ is a parameter called the learning rate.

Online Newton Step (ONS): ONS uses a Newton step based method to compute the portfolio for the next iteration [1]. The Online Newton Step method can be shown to achieve sublinear regret and hence is an universal strategy.

The ONS algorithm uses following portfolio update method for round $t > 1$:

$$x_t = \prod_{\Delta_n}^{\mathbf{A_t}-1} \left(x_{t-1} - \frac{1}{\beta} \mathbf{A_{t-1}^{-1}} \nabla_{t-1} \right) \tag{22}$$

where $\nabla_t = \nabla[\log(x_t \cdot r_t)]$, $\mathbf{A}_t = \sum_{\tau=1}^{t} \nabla_\tau \nabla_\tau + \mathbb{I}$, β is a non-negative constant, and $\prod_{\Delta_\mathbf{n}}^{\mathbf{A_t}-1}$ is the projection onto the n-simplex Δ_n.

Anticor: Anticor is a heuristic based method which does not confirm to the *universal* property for portfolio selection algorithms [5]. Here learning the best stocks (to invest money in) is done by exploiting the volatility of the market and the statistical relationship between the stocks. It implements the 'reversal to the mean' market phenomenon rather aggressively. One of the most important parameters for Anticor is the window length win. The version of Anticor implemented, works with two most recent windows of length win. The strategy is to move money from a stock i to stock j if the growth rate of stock i is greater than the growth rate of j in the most recent window. An additional condition that requires to be satisfied is the existence of a positive correlation between stock i in the second last window and stock j in the last window. For more details on the Anticor algorithm please refer to [5]. The experiments with different variations of the Anticor algorithm in [5], brought to the fore the exceptional empirical performance improvement that this heuristic-based approach can achieve over theoretically motivated approaches.

The performance of Anticor is sensitive to the window size win [5]. One way to address this issue is to adaptively learn the weights and invest in a weighted version of all Anticor$_{win}$s where $win \leqslant W$. We consider a variant BAH(Anticor$_W$), which maintains a uniform buy-and-hold investment on the Anticor$_{win}$, $win \in [2, W]$.

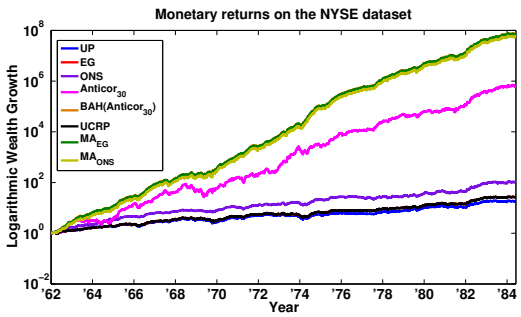
4.2 Meta Algorithms

Meta Algorithms (MAs) are constructed by combining a pool of base algorithms. Meta algorithm MA$_{EG}$ uses the gradient updates and this follows from OGU in Algorithm 1 and Meta Algorithm MA$_{ONS}$ uses Newton updates and follows from ONU in Algorithm 2. Meta Algorithms MA$_{EG}$ and MA$_{ONS}$ are universal if at least one of the base algorithms in their pool is universal.

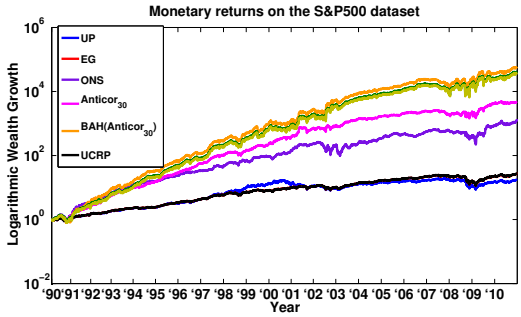
Additionaly, we used two other versions of Meta Algorithms for our experiments: MA$_{Anticor}$ and MA$_{BAH}$. MA$_{Anticor}$ is a Meta Algorithm version of Anticor. Like Anticor it works with different window lengths over the pool of base algorithms. MA$_{BAH}$ does a uniform buy-and-hold over the base algorithms and does not move money between the algorithms. Unlike MA$_{EG}$ and MA$_{ONS}$, MA$_{Anticor}$ and MA$_{BAH}$ have no performance guarantees.

4.3 Results

The experimental setup can be broadly categorized into three subcategories: (a) Universal Pool, (b) Mixed Pool 1 and (c) Mixed Pool 2 based on the pool of base algorithms that were used by the MAs.

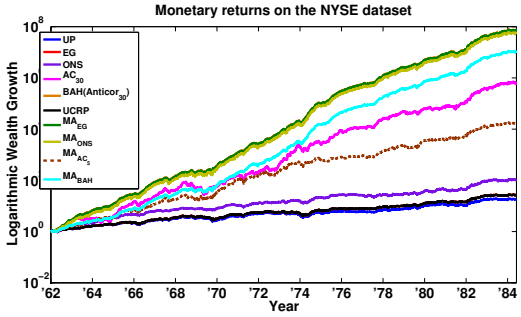


(a) Monetary returns on NYSE.

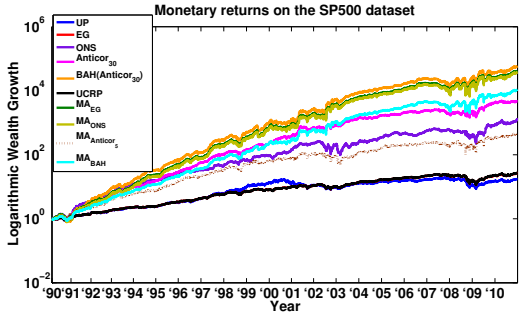


(b) Monetary returns on S&P500.

Figure 4: Monetary returns of the meta algorithms(MA_{EG}, MA_{ONS}). BAH(Anticor_{30})is added to the pool of base algorithms. The Meta Algorithms continue to track the best algorithm in the pool.(best viewed in color).



(a) Monetary returns on NYSE.



(b) Monetary returns on S&P500.

Figure 5: Monetary returns of the meta algorithms(MA_{EG}, MA_{ONS}, $\text{MA}_{Anticor}$, MA_{BAH}). MA_{EG} performs best while $\text{MA}_{Anticor}$ doesn't fare well.(best viewed in color).

Universal Pool: In the Universal Pool setup, the MAs worked with universal base algorithms UP, EG, and ONS. Figure 2 shows the wealth accumulated by the universal base algorithms on the NYSE and S&P500 datasets. We see that ONS performs best, followed by EG and UP. Figure 2 also shows that the two Meta Algorithms MA_{EG} and MA_{ONS} are able to catch up with the performance of ONS, the best performing algorithm in the pool.

Mixed Pool 1: The Mixed Pool 1 is formed by adding Anticor to the Universal Pool of base algorithms. In Figure 3, we see that there is a stark difference in the wealth garnered by Anticor as compared to the Universal Pool of base algorithms. While for the NYSE dataset, Anticor's wealth is of the order of 10^6 (almost 10^4 times the wealth gathered by ONS), for S&P500, the wealth reaches the order of 10^3. MA_{EG} is able to catch up with the performance of Anticor for both the datasets. MA_{ONS} is very slightly behind Anticor and MA_{EG} on the NYSE dataset. On S&P500, the base algorithm ONS outperforms MA_{ONS} (which is still better than UP and EG by a sybstantial margin).

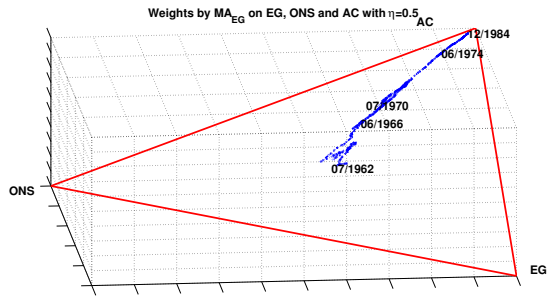
Mixed Pool 2: To further emphasize the strength of the MAs we formed Mixed Pool 2 of base algorithms by adding $\text{BAH}(\text{Anticor}_W)$ to Mixed Pool 1. $\text{BAH}(\text{Anticor}_W)$ outperforms Anticor_{win} (for $win \leqslant W$), EG, UP, and ONS. Figure 4 shows that the wealth achieved by MA_{EG} and MA_{ONS} with $\text{BAH}(\text{Anticor}_W)$ in the pool, is almost as much as $\text{BAH}(\text{Anticor}_W)$ itself. Thus we see that Meta Algorithms, MA_{EG} and MA_{ONS} are competitive with the best base algorithm in all the three experimental setups.

Figure 5, shows the performance of $\text{MA}_{Anticor}$ and MA_{BAH} with the Mixed Pool 2 of base algorithms. The performance of the $\text{MA}_{Anticor}$ is inferior than both MA_{EG} and MA_{ONS}. We could attribute this inferior performance to the inherent nature of Anticor which will tend to move money away from the base algorithms which are performing well. With a buy-and-hold version of MA called MA_{BAH}, the wealth gained is more than Anticor_w, but less than that of $\text{BAH}(\text{Anticor}_W)$ and MA_{EG} and MA_{ONS}.

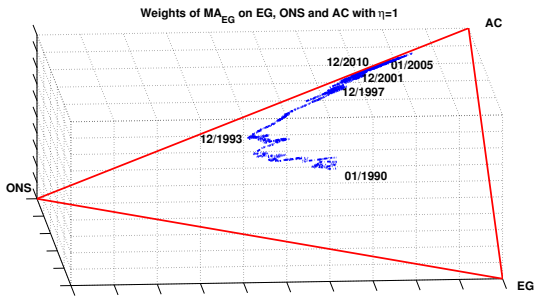
Parameter choices: Parameter choices had to be made for the universal as well as non-universal base algorithms. For EG, we experimented with different learning rate (η) values and found $\eta = 0.05$ to be a good choice, validating the observations in [16]. For ONS, β value was chosen as 1. This gave better results than when β was chosen as a function of the market-variability (refer to [1] for details).

The window size w for Anticor_{win} was taken to be 30. The performance of Anticor_{win} is a function of win as demonstrated in [5]. $\text{BAH}(\text{Anticor}_W)$ combines multiple Anticor_{win} algorithms, $win \in [2, W]$ to harness the strength of these different versions. We choose $W = 30$ for our experiments which might not be the optimal value for the NYSE and S&P500 datasets. Hoever, it is observed that $\text{BAH}(\text{Anticor}_{30})$ surpasses all the base universal algorithms and Anticor_{30} in terms of empirical performance and helps us establish that MA_{EG} and MA_{ONS} can always manage to track the best strategy even if it is a heuristic.

The rate at which MA_{EG} caught up with Anticor in terms of wealth increased as we increased the value of η. However, for very high values of η ($\eta \geqslant 50$), the increase in the wealth gathered by MA_{EG} changed by a small amount. For MA_{ONS}, the β value was chosen according to Lemma 3. $\text{MA}_{Anticor}$ was run with different window lengths. We plotted the results with a window length of 5, as this version was observed to perform reasonably well. It



(a) NYSE: weights of MA_{EG} with η=0.5.



(b) S&P500: weights of MA_{EG} with η=1.

Figure 6: Traces the weights maintained by MA_{EG} on the base algorithms EG, ONS and AC for NYSE and S&P500 with η values 0.5 and 1 respectively(best viewed in color).

was observed that the performance of $MA_{Anticor}$ decreased as the window length was increased beyond 10 days.

APY and Volatility: Table 1 presents the monetary returns in dollars, APY and volatility of the universal and non-universal algorithms on the two datasets.

The wealth for the algorithms has been expressed as the final return on an initial investment of \$1. The values given for MA_{EG} and MA_{ONS} are with Mixed Pool 2 of base algorithms. The top three final returns appear in bold-face.

APY: The Annual Percentage Yield (APY) of the algorithms were calculated based on the following formula:

$$\mathbf{APY} = \left[(\mathbf{Final}/\mathbf{Initial})^{\frac{1}{\mathbf{T_{years}}}} - 1\right] \times 100$$

where $\mathbf{Final}$ and $\mathbf{Initial}$ are the final return and initial investment respectively for $\mathbf{T_{years}}$. BAH($Anticor_{30}$, MA_{EG} and MA_{ONS} have the top three (in the order mentioned) APY for both the datasets.

Volatility: Volatility of the Algorithms were calculated by taking the standard deviation of the sequence of daily wealth relatives $({x_t}^T r_t)$ over $\mathbf{T_{years}}$. $Anticor_{30}$ and BAH($Anticor_{30}$) have more than twice the volatility of the base universal algorithms. MA_{EG} and MA_{ONS} have almost the same volatility as BAH($Anticor_{30}$). This could be explained by the fact that the MAs try to track BAH($Anticor_{30}$).

Figure 6 renders the path traced by MA_{EG} as a weighted combination of EG, ONS, and Anticor. Our experiments show that Anticor outperforms EG and ONS in terms of accumulated wealth by an overwhelming margin. Hence, we see that MA_{EG} converges towards the Anticor corner and the rate of convergence depends on the learning rate η. We also conducted experiments where we interchanged the weights of AC and ONS (switched corners) once every t' days. The value of t' was taken to be 500, 1000 and 2000 days for our experiments. Even after switching, MA_{EG} quickly learned that Anticor is the best performing strategy. This led MA_{EG} to automatically adjust its weight distribution over the base algoothms. MA_{EG} is sensitive to the volatility. Our experiments show that MA_{EG} might prefer ONS earlier on due to its low volatility (See Figure 6 (b)). But as the wealth accumulated by Anticor increases, it shifts its weight from ONS to Anticor.

Table 1: Monetary returns in dollars (per \$1 investment), APY and volatility of universal and non-universal algorithms

Algorithm		NYSE	SP500
UP	wealth	18.56	17.42
	APY	14.20	15.36
	volatility	0.0089	0.0139
EG	wealth	27.10	26.83
	APY	16.18	17.88
	volatility	0.0085	0.0116
ONS	wealth	109.17	1217.39
	APY	23.78	42.65
	volatility	0.0113	0.0201
$Anticor_{30}$	wealth	617754.61	4769.39
	APY	83.32	52.73
	volatility	0.0284	0.0191
BAH($Anticor_{30}$)	wealth	**77626129.46**	**58591.86**
	APY	**128.37**	**73.14**
	volatility	0.0195	0.03296
MA_{EG}	wealth	**68381688.71**	**41678.69**
	APY	**127.06**	**70.21**
	volatility	0.0194	0.0287
MA_{ONS}	wealth	**61119978.64**	**37667.41**
	APY	**125.90**	**69.36**
	volatility	0.0194	0.0286

5. CONCLUSIONS

In this paper, we have presented the idea of designing new Meta Algorithms which work with a pool of base algorithms for optimization. We have shown that solutions from multiple iterative algorithms can be combined by Meta Algorithms to outperform the single best base algorithm. We demonstrate the efficacy of the Meta Algorithms in the domain of Online Portfolio Selection. Detailed experiments over the NYSE and S&P500 datasets show that the Meta Algorithms MA_{EG} and MA_{ONS} beat the universal algorithms in terms of empirical performance but are still competitive with the best CRP in hindsight.

Although, the Meta Algorithms have exceptional performance, they do not take into account the commission one has to pay while trading. This is also a shortcoming with the existing universal and non-universal portfolio selection algorithms [10] [16] [1] [5]. Most of these algorithms trade every stock every day which is not practical as one can incur huge amount of commission costs. As part of our future work, we would like to investigate if a sparse version of the meta algorithm can take care of commissions and yet achieve good empirical performance. Also, the current models for on-line portfolio selection do not model risk. Modeling risk and taking ac-

count of volatility of stocks is an interesting direction for our future work.

Acknowledgements: The research was supported by NSF CAREER award IIS-0953274, and NSF grants IIS-0916750, IIS-08121-83, IIS-1029711, and NetSE-1017647. The authors also wish to thank Huahua Wang and Padmanabhan Balasubramanian for their help.

6. REFERENCES

[1] A. Agarwal, E. Hazan, S. Kale, and R. Schapire. Algorithms for portfolio management based on the newton method. *Proceedings of the 23rd International Conference on Machine Learning*, pages 9–16, 2006.

[2] S. Arora, E. Hazan, and S. Kale. The multiplicative update algorithm: A meta algorithm and applications. Technical report, Dept of Computer Science, Princeton University, 2005.

[3] A. Banerjee. On Bayesian bounds. In *Proceedings of the 23rd International Conference on Machine Learning*, 2006.

[4] A. Blum and A. Kalai. Universal portfolios with and without transaction costs. In *Proceedings of the 10th Annual Conference on Learning Theory*, 1997.

[5] A. Borodin, R. El-Yaniv, and V. Gogan. Can we learn to beat the best stock. *Journal of Artificial Intelligence Research*, 21:579–594, 2004.

[6] L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

[7] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

[8] N. Cesa-Bianchi, Y. Freund, D. P. Helmbold, D. Haussler, R. Schapire, and M. K. Warmuth. How to use expert advice. *Journal of the ACM*, 44(3):427–485, 1997.

[9] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[10] T. Cover. Universal portfolios. *Mathematical Finance*, 1:1–29, 1991.

[11] Y. Freund and R. Schapire. Adaptive game playing using multiplicative weights. *Games and Economic Behavior*, 29:79–103, 1999.

[12] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[13] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *Annals of Statistics*, 2000.

[14] B. Fristedt and L. Gray. *A Modern Approach to Probability Theory*. Birkhauser Verlag, 1997.

[15] E. Hazan, A. Agarwal, and S. Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.

[16] D. Helmbold, E. Scahpire, Y. Singer, and M. Warmuth. Online portfolio setection using multiplicative weights. *Mathematical Finance*, 8(4):325–347, 1998.

[17] A. Kalai and S. Vempala. Efficient algorithms for universal portfolios. *Journal of Machine Learning Research*, 3(3):423–440, 2002.

[18] A. Kalai and S. Vempala. Efficient algorithms for on-line optimization. *Journal of Computer and System Sciences*, 713:291–307, 2005.

[19] J. L. Kelly. A new interpretation of information rate. *Bell Systems Technical Journal*, 35:917–926, 1956.

[20] J. Kivinen and M. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–64, 1997.

[21] O. Knill. Probability. Course notes from Caltech, 1994.

[22] N. Littlestone and M. Warmuth. The weighted majority algorithm. *Information and Computation*, 108:212–261, 1994.

[23] H. Markowitz. Portfolio selection. *Journal of Finance*, 7:77–91, 1952.

[24] T. Soule. Voting teams: A cooperative approach to non-typical problems. *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 916–922, 1999.

[25] P. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining, (First Edition)*. 2005.

[26] D. Williams. *Probability with Martingales*. Cambridge University Press, 1991.

[27] W. Yan, M. Sewell, and C. D. Clack. Learning to optimize profits beats predicting returns – comparing techniques for financial portfolio optimisation. *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1681–1688, 2008.

[28] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. *Proceedings of the 20th International Conference on Machine Learning*, 2003.