MACHINE LEARNING IN COMPUTATIONAL FINANCE

By
Victor Boyarshinov

A Thesis Submitted to the Graduate
Faculty of Rensselaer Polytechnic Institute
in Partial Fulfillment of the
Requirements for the Degree of
DOCTOR OF PHILOSOPHY

Major Subject: Computer Science

Approved by the
Examining Committee:

Malik Magdon-Ismail, Thesis Adviser

Costas Busch, Member

Mark Goldberg, Member

Mukkai Krishnamoorthy, Member

John E. Mitchell, Member

Rensselaer Polytechnic Institute
Troy, NY

April 2005
(For Graduation May 2005)

MACHINE LEARNING IN COMPUTATIONAL FINANCE

By
Victor Boyarshinov

An Abstract of a Thesis Submitted to the Graduate
Faculty of Rensselaer Polytechnic Institute
in Partial Fulfillment of the
Requirements for the Degree of
DOCTOR OF PHILOSOPHY

Major Subject: Computer Science

The original of the complete thesis is on file
in the Rensselaer Polytechnic Institute Library.

Approved by the
Examining Committee:

Malik Magdon-Ismail, Thesis Adviser

Costas Busch, Member

Mark Goldberg, Member

Mukkai Krishnamoorthy, Member

John E. Mitchell, Member

Rensselaer Polytechnic Institute
Troy, NY

April 2005
(For Graduation May 2005)

Contents

1 Introduction and Motivation **1**

2 Collecting Training Dataset Using Optimal Trading Strategies **4**
- 2.1 Introduction and Basic Definitions . . . 4
- 2.2 Related Work . . . 8
- 2.3 Contribution Summary . . . 11
- 2.4 Collecting Training Dataset: Return-Optimal Trading Strategies . . . 12
  - 2.4.1 Unconstrained Return-Optimal Trading Strategies . . . 12
  - 2.4.2 Constrained Return-Optimal Strategies . . . 13
- 2.5 Collecting Training Dataset: Sterling-Optimal Trading Strategies . . . 15
  - 2.5.1 Unconstrained Sterling-Optimal Strategies . . . 16
  - 2.5.2 Maximizing $\frac{c_r}{d_r}$ . . . 19
  - 2.5.3 Constrained Sterling-Optimal Strategies . . . 22
- 2.6 Collecting Training Dataset: Sharpe Optimal Trading Strategies . . . 32
  - 2.6.1 Maximizing the Simplified Sharpe Ratio S . . . 32
  - 2.6.2 Maximizing Shrp_2 . . . 33
  - 2.6.3 Approximation Ratio for Shrp_2 . . . 38
  - 2.6.4 Maximizing Shrp_1 . . . 39
  - 2.6.5 Approximation Ratio for Shrp_1 . . . 40
- 2.7 Conclusion and Discussion . . . 40

3 Learning: Optimal Linear Separation **41**
- 3.1 Introduction and Basic Definitions . . . 41
  - 3.1.1 Convex Hulls . . . 41
  - 3.1.2 Linear Separators . . . 42
- 3.2 Related Work . . . 43
  - 3.2.1 Convex Hulls . . . 43
  - 3.2.2 Separable Sets . . . 44
  - 3.2.3 Unseparable Sets . . . 45
- 3.3 Contribution . . . 47
  - 3.3.1 Optimal Linear Separator for Non-Separable Sets . . . 47
  - 3.3.2 Leave-One-Out Error . . . 49
- 3.4 Discussion and Open Questions . . . 53

4 Avoiding Overfitting: Isotonic and Unimodal Regression **55**
4.1 Introduction and Basic Definitions 55
4.2 Previous Work 57
4.3 Contribution 58
4.3.1 L_1-Isotonic Regression 58
4.3.2 L_1-Isotonic Regression: Algorithms 63
4.3.3 L_∞-Prefix-Isotonic Regression 64
4.3.4 L_∞-Prefix-Isotonic Regression: Algorithms 67
4.3.5 L_∞ Unimodal Regression 68
4.4 Conclusion and Discussion 69

List of Figures

2.1 Constructing training set from optimal strategies. 4
2.2 Possible entry and exit points. 17
2.3 Upper-touching point from $\mathbf{p}$ to the convex hull of the set of points P'_m. 20

3.1 Mirrored-radial coordinates. 49
3.2 Case of two perfect predictors. 53
3.3 Optimal boosting of two classifiers. 54

Acknowledgements

During the development of my graduate studies in the Rensselaer Polytechnic Institute several persons and institutions collaborated directly and indirectly with my research. Without their support it would be impossible for me to finish my work.

I want to start expressing a sincere acknowledgement to my advisor, Dr. Malik Magdon-Ismail because he gave me the precious opportunity to research under his guidance and supervision. I received motivation and encouragement form him during all my studies.

I gratefully acknowledge the example, motivation, inspiration and support I received from Dr. Mark Goldberg.

I owe special thanks to Terry Hayden for her never ended support.

As well as that singular votes of thanks I also like to mention the debt that I owe to the department of Computer Science for providing the funding and the resources for the development of this research.

Abstract

We focus on constructing efficient algorithms for problems arising in applications of machine learning in the field of computational finance, in particular pricing and trading. One generally should consider three different facets of designing an algorithm for an artificial intelligence application:

(i) Constructing a training dataset

(ii) Developing a training algorithm for discovering patterns

(iii) Using side information for avoiding overfitting the training data.

The first part of this thesis addresses *(i)*. Specifically, we consider the problem of finding optimal trading strategies with respect to the total cumulative return, Sterling ratio and two variants of the Sharp Ratio. Knowing what the optimal trades are is necessary for constructing the training dataset for a supervised learning algorithm The contribution in this work is to give *efficient* (low order polynomial time) algorithms to compute the optimal trading strategy for various profit objectives, and under constraints on the number of trades that can be made. The heart of our algorithms is a new general algorithm for maximizing quotients over intervals, which we solve by relating this one-dimensional optimization problem to convex hull operations on the plane.

The second part of this thesis addresses *(ii)*. The problem of learning - discovering hidden patterns in the collected data - often can be considered as the problem of finding a linear separator for two sets of points in multidimensional space. When the sets are not separable, we are interested in finding a subset of points such that after removing these points, the remaining points are separable. Such a set is called a separator set. If we assign a positive weight to every point (its fidelity), then we wish to find a separator set with the minimum possible weight. We provide a combinatorial deterministic algorithm for computing such a minimum separator set in 2 dimensions. The constructed algorithm also addresses the statistical properties of the resulting separator by providing leave-one-out error simultaneously.

The problem of overfitting training data is well recognized in the machine learning community. Standard approach to deal with this threat is the early stopping of the training algorithm iterations. Important question is when the iterations should be stopped. Usually one monitores another error measure and stops iterations when the associated error starts growing. The associated error can be same error function measured on separate set of datapoints (validation set) or even completely different error function. Since the associated error measure is somewhat different from the primary, it does not necessary shows monotinical behavior but often appears as random fluctuations around unknown function. In order to pinpoint the exact location of the critical point, one can perform shape constrained optimization to fit function to the ibserved values of the associated error. Another application of the shape constrained regression arise in the pricing of financial instruments, for example the american put option. Isotonic and unimodal regression are both examples of nonparametric shape constrained regression. In the third part of this work we present efficient algorithms for computing for L_∞, L_1 isotonic and unimodal regressions.

Chapter 1

Introduction and Motivation

We focus on constructing efficient algorithms for problems arising in applications of machine learning in the field of computational finance, in particular pricing and trading. A trader has in mind the task of developing a trading system that optimizes some profit criterion, the simplest being the total return. A more conservative approach is to optimize a risk adjusted return. Widely followed measures of risk adjusted returns are the Sterling Ratio and Sharpe Ratio. In an enviroment where markets exhibit frequent crashes and portfolios encounter sustained periods of losses, the Sterling ratio and the Sharpe ratio have emerged as the leading performance measures used in the industry. Given a set of instruments, a trading strategy is a switching function that transfers the wealth from one instrument to another.

One generally should consider three different facets of designing an algorithm for an artificial intelligence application:

(i) Constructing a training dataset

(ii) Developing a training algorithm for discovering patterns

(iii) Using side information for avoiding overfitting the training data.

The first part of this thesis addresses *(i)*. Specifically, we consider the problem of finding optimal trading strategies with respect to the total cumulative return, Sterling ratio and two variants of the Sharp Ratio. The motivations for constructing such optimal strategies are: (i) Knowing what the optimal trades are is necessary for constructing the training dataset for a supervised learning algorithm (ii) The optimal performance modulo certain trading constraints can be used as a benchmark for real trading systems. (iii) Optimal trading strategies (with or without constraints) can be used to quantitatively rank various markets (and time scales) with respect to their profitability according to a given criterion. A brute force approach to obtaining such optimal trading strategies would search through the space of all possible trading strategies, keeping only the one satisfying the optimality criterion. Since the number of possible trading strategies grows exponentially with time, the brute force approach leads to an exponential time algorithm.

All the reasearch on optimal trading falls into two broad categories. The first group is on the more theoretical side where researchers assume that stock prices satisfy some particular model, for example the prices are driven by a stochastic process of known form; the goal is to derive closed-form solutions for the optimal trading strategy, or a set of equations that the optimal strategy must

follow. Representative approaches from this group include single-period portfolio optimization [69], representation of optimal strategies as a solution of free-boundary problem [74] and characterization of optimal strategies in terms of a nonlinear quasi-variational stochastic inequality [9]. The main drawbacks of such theoretical approaches is that their prescriptions can only be useful to the extent that the assumed models are correct. The second group of research which is more on the practical side is focused on exploring learning methods for the prediction of future stock prices moves. Intelligent agents are designed by training on past data and their performance is compared with some benchmark strategies. Examples from this group include work of Zakamouline [77] where he characterize optimal strategy as function of the investor's horizon, current market state and the composition of the investor's wealth. He also provides numerical method for finding the optimal strategies.

The Sharpe ratio was introduced in [65], [66] and since then became a popular and widespread risk-sensitive measure of a portfolio performance. Faugere et al. in [68] used Sharpe ratio as one of performance measures to compare effectiveness of different decision-making criteria. Pedersen et al. in [63] performed empirical comparison of many performance measurement methodologies for the global financial services sector and documented strong evidence in support of using Sharp-Ratio based measures. It was also pointed out that correlation of the Sterling ratio with other measures is usually small and dips below 5% in some instances. Surprisingly, there is little previous research on portfolio optimization with respect to the Sharpe ratio or Sterling ratio, partially because of the intristic difficulty of these non-linear optimization criteria. Usually one employs heuristic methods to obtain a locally optimal trading strategy [32] or uses other learning methods where training dataset is not necessary [34].

The contribution in this work is to give *efficient* (low order polynomial time) algorithms to compute the optimal trading strategy for various profit objectives, and under constraints on the number of trades that can be made. The heart of our algorithms is a new general algorithm for maximizing quotients over intervals, which we solve by relating this one-dimensional optimization problem to convex hull operations on the plane.

The second part of this thesis addresses *(ii)*. The problem of learning - discovering hidden patterns in the collected data - often can be considered as the problem of finding a linear separator for two sets of points in multidimensional space. When the sets are not separable, we are interested in finding a subset of points such that after removing these points, the remaining points are separable. Such a set is called a separator set. If we assign a positive weight to every point (its fidelity), then we wish to find a separator set with the minimum possible weight. Optimal linear separator also can be used for optimal boosting of two classifiers (see section 3.4). Popular approaches here are combinatorial heuristics, reduced convex hulls [5] and interior point techniques for combinatorial optimization [49, 11, 33, 40, 35]. We provide a combinatorial deterministic algorithm for computing such a minimum separator set in 2 dimensions with time complexity $O(n^2 log\ n)$. An important issue in the design of efficient machine learning systems is the estimation of the accuracy of learning algorithms, in particular its sensitivity to noisy inputs. One classical estimator is *leave-one-out* error, which is commonly used in practice. Intuitively, the leave-one-out error is defined as the average error obtained by training a classifier on $n-1$ points and evaluating it on the point left out. Our algorithm also addresses the statistical properties of the resulting separator by providing leave-one-out error simultaneously.

The problem of overfitting training data is well recognized in the machine learning community. Standard approach to deal with this threat is the early stopping of the training algorithm iterations.

Important question is when the iterations should be stopped. Usually one monitores another error measure and stops iterations when the associated error starts growing. The associated error can be same error function measured on separate set of datapoints (validation set) or even completely different error function. Since the associated error measure is somewhat different from the primary, it does not necessary shows monotinical behavior but often appears as random fluctuations around unknown function. In order to pinpoint the exact location of the critical point, one can perform shape constrained optimization to fit function to the ibserved values of the associated error. Another application of the shape constrained regression arise in the pricing of financial instruments, for example the american put option. A futher example of application of isotonic regression is epidemiologic studies, where one may be interested in assessing the relationship between dose of a possibly toxic exposure and the probability of an adverse response. In characterizing biologic and public health significance, and the need for possible regulatory interventions, it is important to efficiently estimate dose response, allowing for flat regions in which increases in dose have no effect. In such applications, one can typically assume a priori that an adverse response does not occur less often as dose increases, adjusting for important confounding factors, such as age and race. It is well known that incorporating such monotonicity constraints can improve estimation efficiency and power to detect trends. Isotonic and unimodal regression are both examples of nonparametric shape constrained regression.

It is known that L_2 isotonic regression can be performed efficiently in linear time [1, 61]. For L_1 isotonic regression, algorithms in the efficiency class $O(n \log n)$ are known [13, 57, 62]. While $O(n \log n)$ is optimal for L_1 prefix-isotonic regression [72], it is not known whether the apparently simpler isotonic regression problem can be performed faster than $O(n \log n)$. We take a first step in this direction by obtaining a linear bound in terms of the size of the output (K). In the third part of this work we present efficient algorithms for computing for L_∞, L_1 isotonic and unimodal regressions.

Chapter 2

Collecting Training Dataset Using Optimal Trading Strategies

2.1 Introduction and Basic Definitions

A trader has in mind the task of developing a trading system that optimizes some profit criterion, the simplest being the total return. A more conservative approach is to optimize a risk adjusted return. Widely followed measures of risk adjusted returns are the Sterling Ratio and Sharpe Ratio. In an enviroment where markets exhibit frequent crashes and portfolios encounter sustained periods of losses, the Sterling ratio and the Sharpe ratio have emerged as the leading performance measures used in the industry. Given a set of instruments, a trading strategy is a switching function that transfers the wealth from one instrument to another. In this chapter, we consider the problem of finding optimal trading strategies, i.e., trading strategies that maximize a given optimality criterion.

The knowledge of what the optimal trading strategies are is necessary for constructing a training dataset for a supervised learning algorithm. One popular approach to the predicting the future stock prices moves is training an automated intelligent agent that discover patterns in the stock prices dynamic right before a major market move. During the exploitation stage, the agent observes current state of market. If a pattern recognized that was seen before, agent gives a buy/sell signal. If we have an optimal ex-post trading strategy, we know when it was the best time to buy or sell an asset. Then we can investigate the short interval of time that preceeds the optimal entry/exit point and extract a set of features that describe market condition during the interval. Examples of commonly considered features include market volatility, total volume and amount of open interest. The features vector that was extracted from an interval preceeding the optimal entry point become a training point with value $+1$ and features extracted from an interval preceeding the optimal exit point become training point with value -1 (see Fugure 2.1 above).

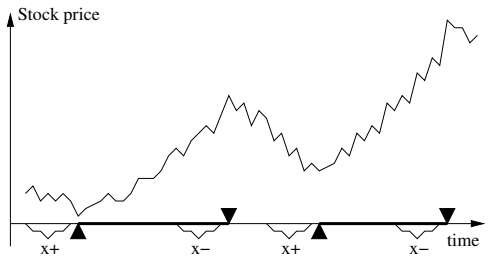


Figure 2.1: Constructing training set from optimal strategies.

In this work we consider optimal strategies with respect to the total cumulative return, as well

as with respect to various risk adjusted measures of return (the Sterling ratio and variants of the Sharpe ratio). A brute force approach to obtaining such optimal trading strategies would search through the space of all possible trading strategies, keeping only the one satisfying the optimality criterion. Since the number of possible trading strategies grows exponentially with time, the brute force approach leads to an exponential time algorithm[1], which for all practical purposes is infeasible – even given the pace at which computing power grows. The contribution in this work is to give *efficient* (polynomial time) algorithms to compute the optimal trading strategy for various profit objectives, and under constraints on the number of trades that can be made.

The motivations for constructing such optimal strategies are: (i) Knowing what the optimal trades are, one can one can try to *learn* to predict good trading opportunities by using market and/or technical indicators as features on which to base the prediction. A host of such activity goes under the name of *financial engineering*. (ii) The optimal performance modulo certain trading constraints can be used as a benchmark for real trading systems. For example, how good is a trading system that makes ten trades with a Sterling ratio of 4 over a given time period? One natural comparison is to benchmark this trading strategy against a Sterling-optimal trading strategy that makes at most ten trades over the same time period. (iii) Optimal trading strategies (with or without constraints) can be used to quantitatively rank various markets (and time scales) with respect to their profitability according to a given criterion. So for example, one could determine the optimal time scale on which to trade a particular market, or given a set of markets, which is the most (risk adjusted) profit-friendly.

In order to make the preceeding discussion more precise and to more accurately state our results, lets introduce a few definitions. Assume that we have two instruments, for concreteness, a stock S and a bond B with price histories $\{S_0, \ldots, S_n\}$ and $\{B_0, \ldots, B_n\}$ over n consecutive time periods, t_i, $i \in \{1, \ldots, n\}$. Thus, for example, over time period t_i, the price of stock moved from S_{i-1} to S_i. We denote the return sequence for the two instruments by $\{s_1, \ldots, s_n\}$ and $\{b_1, \ldots, b_n\}$ respectively: $s_i = \log \frac{S_i}{S_{i-1}}$, and correspondingly, $b_i = \log \frac{B_i}{B_{i-1}}$. We assume that one of the instruments is the benchmark instrument, and that all the equity is held in the benchmark instrument at the begining and end of trading. The bond is usually considered the benchmark instrument, and for illustration, we will follow this convention. The trivial trading strategy is to simply hold onto bond for the entire duration of the trading period. It is useful to define the excess return sequence for the stock, $\hat{s}_i = s_i - b_i$. When the benchmark instrument is the bond, the excess return as we defined it is the conventionally used one. However, one may want to measure performances of a trading strategy with respect to the S&P 500 as benchmark instrument, in which case the excess return would be determined relative to the S&P 500 return sequence. The excess return sequence for the bond is just the sequence of zeros, $\hat{b}_i = 0$. Conventionally, the performance of a strategy is measured relative to some trivial strategy, so the excess return sequence will be the basis of most of our performance measures.

Definition 2.1.1 (Trading Strategy) *A trading strategy $\mathcal{T}$ is a boolean n-dimensional vector indicating where the money is at the end of time period t_i:*

$$\mathcal{T}[i] = \begin{cases} 1 & \textit{if money is in stock at the end of } t_i, \\ 0 & \textit{if money is in bond at the end of } t_i. \end{cases}$$

[1]The asymptotic running time of an algorithm is measured in terms of the input size n. If the input is a time sequence of n price data points, then polynomial time algorithms have run time that is bounded by some polynomial in n. Exponential time algorithms have running time greater than some exponentially growing function in n [17].

We assume that $\mathcal{T}[0] = \mathcal{T}[n] = 0$, i.e., all the money begins and ends in bond. A trade is entered at time t_i if $\mathcal{T}[i] = 0$ and $\mathcal{T}[i+1] = 1$. A trade is exited at time t_i if $\mathcal{T}[i] = 1$ and $\mathcal{T}[i+1] = 0$. The number of trades made by a trading strategy is equal to the number of trades that are entered.

Note that we make the following assumptions regarding the trading:

A1 *[All or Nothing]*: The position at all times is either entirely bond or entirely stock.

A2 *[No Market Impact]*: Trades can be placed without affecting the quoted price.

A3 *[Fractional Market]*: Arbitrary amounts of stock or bond can be bougnt or sold at any time.

A4 *[Long Strategies]*: We assume that we can only hold long positions in stock or bond.

These assumptions are rather mild and quite accurate in most liquid markets, for example foreign exchange. **A1** implies (for example) that one can not leg into a trade. For some optimality criteria, legging into a trade may be beneficial, however, in most circumstances, an all-or-nothing optimal strategy can be chosen. **A3** is a consequence of **A1**, since if all the money should be transfered to a stock position, this may necessitate the purchase of a fractional number of shares. Note that if $\mathcal{T}[i-1] \neq \mathcal{T}[i]$, then at the begining of time period t_i, the position was transfered from one instrument to another. Such a transfer will incur an instantaneous per unit transaction cost equal to the bid-ask spread of the instrument being transfered into. We assume that the bid-ask spread is some fraction (f_B for bond and f_S for stock) of the bid price.

We denote the equity curve for a trading strategy $\mathcal{T}$ by the vector $\mathcal{E}_\mathcal{T}$, i.e., $\mathcal{E}_\mathcal{T}[i]$ is the equity at the end of time period t_i, with $\mathcal{E}_\mathcal{T}[0] = 1$. Corresponding to the equity curve is the excess return sequence $r_\mathcal{T}$ for the trading strategy $\mathcal{T}$, i.e., for $i \geq 1$

$$r_\mathcal{T}[i] = \log \frac{\mathcal{E}_\mathcal{T}[i]}{\mathcal{E}_\mathcal{T}[i-1]} - b_i. \tag{2.1}$$

If we ignore the bid-ask spread, then the excess return in time period t_i is given by

$$r_\mathcal{T}[i] = \hat{s}_i \mathcal{T}[i] = (s_i - b_i)\mathcal{T}[i]. \tag{2.2}$$

The bid-ask spread affects the return, reducing it by an amount depending on $\mathcal{T}[i-1]$. Denoting this transactions cost attributable to $\mathcal{T}[i]$ by $\Delta[i]$, we have that

$$\Delta[i] = -\mathcal{T}[i-1](1 - \mathcal{T}[i])\hat{f}_B - (1 - \mathcal{T}[i-1])\mathcal{T}[i]\hat{f}_S, \tag{2.3}$$

where $\hat{f}_S = \log(1+f_S)$ and $\hat{f}_B = \log(1+f_B)$. Thus, the bid-ask spread can be viewed as introducing an instantaneous return of $-\hat{f}_B$ or $-\hat{f}_S$ whenever the position is switched. To exactly which time period this transactions cost is applied may depend on the nature of the market, i.e., it may be applied to $r_\mathcal{T}[i]$, $r_\mathcal{T}[i-1]$ or $r_\mathcal{T}[i+1]$. The nature of the results will not change significantly for either of these options, so in our algorithms, we will generally make the choice that offers the greatest technical simplicity. For a trading strategy $\mathcal{T}$, we define the total return $\mu(\mathcal{T})$, the sum of the squared returns $s^2(\mathcal{T})$, the sum of squared deviations of the returns $\sigma^2(\mathcal{T})$ and the maximum

drawdown $MDD(\mathcal{T})$ as follows,

$$\mu(\mathcal{T}) = \sum_{i=1}^{n} r_{\mathcal{T}}[i], \tag{2.4}$$

$$s^2(\mathcal{T}) = \sum_{i=1}^{n} r_{\mathcal{T}}[i]^2, \tag{2.5}$$

$$\sigma^2(\mathcal{T}) = \sum_{i=1}^{n} \left(r_{\mathcal{T}}[i] - \frac{1}{n}\mu(\mathcal{T}) \right)^2 = s^2(\mathcal{T}) - \frac{1}{n}\mu^2(\mathcal{T}), \tag{2.6}$$

$$MDD(\mathcal{T}) = \max_{1 \le k \le l \le n} -\sum_{i=k}^{l} r_{\mathcal{T}}[i]. \tag{2.7}$$

When it is clear from the context what trading strategy we are talking about, we will generally suppress the explicit dependence on $\mathcal{T}$. The performance measures that we consider in this chapter are derived from these statistics. In particular, we are interested in the total return μ, the Sterling ratio Strl, and variants of the Sharpe ratio, Shrp_1 and Shrp_2:

$$\mathsf{Strl}(\mathcal{T}) = \frac{\mu(\mathcal{T})}{MDD(\mathcal{T})}, \qquad \mathsf{Shrp}_1(\mathcal{T}) = \frac{\mu(\mathcal{T})}{\sigma(\mathcal{T})}, \qquad \mathsf{Shrp}_2(\mathcal{T}) = \frac{\mu(\mathcal{T})}{\sigma^2(\mathcal{T})}. \tag{2.8}$$

Shrp_1 is the conventionally used Sharpe ratio. Shrp_2 is a more risk averse performance measure, as it is more sensitive to the variance in the returns. Often, Strl as we have defined it is refered to as the Calmar ratio in the literature [31], and the Sterling ratio adds a constant (for example 10%) to the MDD in the denominator [53]. Such a constant can easily be accomodated by our algorithms, and so we will maintain this simpler definition for the Sterling ratio.

We will use standard $O()$ notation in stating our results: let n be the length of the returns sequences; we say that the run time of an algorithm is $O(f(n))$ if, for some constant C, the runtime is $\le Cf(n)$ for any possible return sequences. If $f(n)$ is linear (quadratic), we say that the runtime is linear (quadratic).

Next, we discuss the existing related work, followed by a detailed discussion of the algorithms, along with all necessary proofs.

2.2 Related Work

The body of literature on optimal trading is so enormous that we only mention here some representative papers. All the reasearch on optimal trading falls into two broad categories. The first group is on the more theoretical side where researchers assume that stock prices satisfy some particular model, for example the prices are driven by a stochastic process of known form; the goal is to derive closed-form solutions for the optimal trading strategy, or a set of equations that the optimal strategy must follow. We highlight some of this research below.

Lobo et al. in [69] consider the problem of single-period portfolio optimization. They consider the maximization of the expected return subject to different types of constraints on the portfolio (margin, diversification, budget constraints and limits on variance or shortfall risk). Under the assumption that the return on asset i at the end of period is the random variable a_i and both first and second moments of the joint distribution of the random variables are known, this optimization problem can be represented as a convex optimization problem and thus can be efficiently solved by a numerical methods (eq. by Karmarkar's interior-point method [38]).

Thompson in [74] considered the problem of maximizing the (expected) total cumulative return of a trading strategy under the assumption that the asset price satisfies a stochastic differential equation of the form $dS_t = dB_t + h(X_t)dt$, where B_t is a Brownian motion, h is a known function and X_i is a Markov Chain independent of the Brownian motion. In this work, he assumes fixed transaction costs and imposes assumptions $A1$, $A2$, $A4$ on the trading. He also imposes a stricter version of our assumption $A3$: at any time, trader can have only 0 or 1 unit of stock. He proves that the optimal trading strategy is the solution of a free-boundary problem, gives explicit solutions for several functions h and provides bounds on the transaction cost above which it is optimal never to buy the asset at all.

Pliska et al. in [8] considered the problem of an optimal investment for a continuous-time market consisting of the usual bank account, a rolling horizon bond and a discount bond whose maturity coincides with the planning horizon. They assume interest rates to be stochastic (driven by a stochastic differential equation) and derive an equation satisfied by the trading strategy that maximizes the HARA utility wealth function.

Bielecki in [9] considered the problem of maximizing the risk sensitive expected exponential growth rate of the investor's portfolio in a economy model consisting of a bank account and a risky security (stock) with a stochastic interest rate. The optimal trading strategy is characterized in terms of a nonlinear quasi-variational inequality and he developed a numerical approach to solving this equation.

Berkelaar and Kouwenberg in [7] considered asset allocation in a mean versus downside-risk framework. Downside-risk measures penalize only negative returns relative to a given benchmark. Investors have a trade-off between mean and downside-risk. Closed-form solutions for the optimal asset allocation are derived for the case where asset prices follow geometric Brownian motions with constant interest rate.

The main drawbacks of such theoretical approaches is that their prescriptions can only be useful to the extent that the assumed models are correct. The second group of research which is more on the practical side is focused on exploring learning methods for the prediction of future stock prices moves. Intelligent agents are designed by training on past data and their performance is compared with some benchmark strategies.

Liu in [42] consider the optimal investment policy of a constant absolute risk aversion (CARA)

investor who faces fixed and proportional transaction costs when trading multiple risky assets. He show that when asset returns are uncorrelated, the optimal investment policy is to keep the dollar amount invested in each risky asset between two constant levels and upon reaching either of these thresholds, to trade to the corresponding optimal targets.

Zakamouline in [77] studies the optimal portfolio selection problem for a constant relative risk averse investor who faces fixed and proportional transaction costs and maximizes expected utility of the investor's end-of-period wealth. The author applies the method of the Markov chain approximation to numerically solve for the optimal trading strategy. The numerical solution indicates that the portfolio space is divided into three disjoint regions (Buy, Sell and No-Transaction), and four boundaries describe the optimal strategy. If a portfolio lies in the Buy region, the optimal strategy is to buy the risky asset until the portfolio reaches the lower (Buy) target boundary. Similarly, if a portfolio lies in the Sell region, the optimal strategy is to sell the risky asset until the portfolio reaches the upper (Sell) target boundary. All these boundaries are functions of the investor's horizon and the composition of the investor's wealth.

Choi and Liu in [41] considered trading tasks faced by an autonomous trading agent. An autonomous trading agent works as follows. First, it observes the state of the environment. According to the environment state, the agent responds with an action, which in turn influences the current environment state. In the next time step, the agent receives a feedback (reward or penalty) from the environment and then perceives the next environment state. The optimal trading strategy for the agent was constructed in terms of the agent's expected utility (expected accumulated reward).

Cuoco et al. in [36] considered Value at Risk as a tool to measure and control the risk of the trading portfolio. The problem of a dynamically consistent optimal porfolio choice subject to the Value at Risk limits was formulated and they proved that the risk exposure of a trader subject to a Value at Risk limit is always lower than that of an unconstrained trader and that the probability of extreme losses is also decreased. They also prove that another risk measure - Tail Conditional Expectation - is equivalent to the Value at Risk. In particular, they showed that in a dynamic setting it is always possible to transform any given Tail Conditional Expectation limit into an equivalent Value at Risk limit and conversely.

Dammon and Spatt in [70] explored the optimal trading and pricing of taxable securities with asymmetric capital gains taxes and transaction costs. Under current U.S. tax law, gains on capital assets are not taxed until the investor sells the asset and the tax rate that applies to capital gains and losses may be a function of the investor's holding period. These features give investors the incentive to time their asset sales so as to minimize the tax burden of owning taxable securities. In this work the optimal trading strategy is derived in the presence of capital gains taxes and transaction costs and the implications of the optimal trading strategy for asset pricing is explored.

Mihatsch and Neuneier in [46] considered problem of optimization of a risk-sensitive expected return of a Markov Decision Problem. Based on an extended set of optimality equations, risk-sensitive versions of various well-known reinforcement learning algorithms were formulated and they showed that these algorithms converge with probability one under reasonable conditions.

The Sharpe ratio was introduced in [65], [66] and since then became a popular and widespread risk-sensitive measure of a portfolio performance. Faugere et al. in [68] used Sharpe ratio as one of performance measures to compare effectiveness of different decision-making criteria. Pedersen et al. in [63] performed empirical comparison of many performance measurement methodologies for the global financial services sector and documented strong evidence in support of using Sharp-Ratio based measures. It was also pointed out that correlation of the Sterling ratio with other measures

is usually small and dips below 5% in some instances.

A number of authors associated with BARRA (a major supplier of analytic tools and databases) have used the terms *information ratio* [29] or *Appraisal ratio* instead [10]. Goodwin in [29] considers the relationship between the Sharpe ratio and other performance measures, compares four methods of annualizing an information ratio, and presents the empirical evidence on the distribution of information ratios by style, which provides a context in which to examine manager performance.

Surprisingly, there is little previous research on portfolio optimization with respect to the Sharpe ratio or Sterling ratio, partially because of the intristic difficulty of these non-linear optimization criteria.

Moody and Saffell in [64] presented methods for optimizing portfolios, asset allocations and trading systems based on a direct reinforcement approach. In this approach, investment decision making is viewed as a stochastic control problem and the need to build forecasting models is eliminated. An adaptive algorithm called recurrent reinforcement learning for discovering investment policies was proposed and they demonstrated how it can be used to optimize risk-adjusted investment returns like the Sterling Ratio or Sharpe Ratio, while accounting for the effects of transaction costs.

Liu et al. in [32] proposed a learning-based trading strategy for portfolio management, which aims at maximizing the Sharpe Ratio by actively reallocating wealth among assets. The trading decision is formulated as a non-linear function of the latest realized asset returns, and the function cam be approximated by a neural network. In order to train the neural network, one requires a Sharpe-Optimal trading strategy to provide the supervised learning method with target values. In this work they used heuristic methods to obtain a locally Sharp-optimal trading strategy. The transaction cost was not taken into consideration. Our methods can be considerably useful in the determination of target trading strategies for such approaches.

Tasche and Tibiletti in [73] explored ways to speed up the online computation of the Sharpe ratio of a current portfolio and how the Sharpe ratio will change if a candidate new asset will be incorporated into the portfolio. Approximation formulae were derived that are based on certain derivatives of the Value-at-Risk.

Hellstrom in [34] formulates an alternative formulation of the stock prediction problem based on a statistical investigation of the stock's ranks. In this work a single perceptron neural network is trained to find the parameter vector that maximizes the Sharpe ratio. Due to the lack of the Sharpe-optimal trading decisions that can be supplied as target values, a special technique for optimization without derivatives is utilized [59].

Our work does not make any assumptions about the price dynamics to construct ex-post optimal trading strategies. Obtained results furnish (i) optimal strategies on which to train intelligent agents and (ii) benchmarks with which to compare their performance.

2.3 Contribution Summary

The contribution in this work is to give *efficient* (polynomial time) algorithms to compute the optimal trading strategy for various profit objectives, and under constraints on the number of trades that can ber made. A main component in presented algorithms is the ability to efficiently maximize quotients over intervals. Consider following optimization problem:

Definition 2.3.1 *Given two sequences of real numbers* $\{c_i\}_{i\in I}$, $\{d_i\}_{i\in I}$, $c_i \geq 0$, $d_i \geq 0$, *find a single continuous subinterval of indeces* $J \subseteq I$ *such that*

$$\frac{\sum_{i\in J} c_i}{1+\sum_{i\in J} d_i} \geq \frac{\sum_{i\in J'} c_i}{1+\sum_{i\in J'} d_i}, \qquad \forall J' \subseteq I$$

This problem is relevant because problems of finding the Sterling optimal trading strategy, $Sharp_1$ and $Sharp_2$ optimal trading strategies can be solved by solving a sequence of corresponding optimization problems of type 2.3.1.

We relate this one-dimensional optimization problem to convex hull operations on the plane, that allowed us to constructed an efficient $O(NlogN)$ algorithm that solves problem 2.3.1. Using this algorithm as a starting point, we can then prove the following theorems.

Theorem 2.3.2 (Return Optimal Trading Strategies) *A total return optimal trading strategy can be computed in linear time. Specifically,*

i. Unconstrained Trading. A trading strategy $\mathcal{T}_\mu$ *can be computed in* $O(n)$ *such that for any other strategy* $\mathcal{T}$, $\mu(\mathcal{T}_\mu) \geq \mu(\mathcal{T})$.

ii. **Constrained Trading.** *A trading strategy* $\mathcal{T}_\mu^K$ *making at most* K *trades can be computed in* $O(K \cdot n)$ *such that for any other strategy* $\mathcal{T}^K$ *making at most* K *trades,* $\mu(\mathcal{T}_\mu^K) \geq \mu(\mathcal{T}^K)$.

Theorem 2.3.3 (Sterling Optimal Trading Strategies) *A Sterling optimal trading strategy can be computed in near linear time. Specifically,*

i. **Unconstrained Trading.** *A trading strategy* $\mathcal{T}_{\mathsf{Strl}}$ *can be computed in* $O(n \log n)$ *such that for any other strategy* $\mathcal{T}$, $\mathsf{Strl}(\mathcal{T}_{\mathsf{Strl}}) \geq \mathsf{Strl}(\mathcal{T})$.

ii. **Constrained Trading.** *A trading strategy* $\mathcal{T}_{\mathsf{Strl}}^K$ *making at most* K *trades can be computed in* $O(n \log n)$ *such that for any other strategy* $\mathcal{T}^K$ *making at most* K *trades,* $\mathsf{Strl}(\mathcal{T}_{\mathsf{Strl}}^K) \geq \mathsf{Strl}(\mathcal{T}^K)$.

Theorem 2.3.4 (Sharpe Optimal Trading Strategies) *A Sharpe optimal trading strategy can be computed in near quadratic time. Specifically, trading strategies* $\mathcal{T}_{\mathsf{Shrp}_1}$ *and* $\mathcal{T}_{\mathsf{Shrp}_2}$ *can be found in* $O(n^2 \log n)$ *such that for any other strategy* $\mathcal{T}$, $\mathsf{Strl}_1(\mathcal{T}_{\mathsf{Shrp}_1}) \geq \mathsf{Strl}_1(\mathcal{T})$ *and* $\mathsf{Strl}_2(\mathcal{T}_{\mathsf{Shrp}_2}) \geq \mathsf{Strl}_2(\mathcal{T})$

2.4 Collecting Training Dataset: Return-Optimal Trading Strategies

We use the notation $[t_i, t_j]$ to denote the set of time periods $\{t_i, t_{i+1}, \ldots, t_j\}$. In order to compute the return optimal strategies, we will use a dynamic programming approach to solve a more general problem. Specifically, we will construct the return optimal strategies for every prefix of the returns sequence. First we consider the case when there is no restriction on the number of trades, and then the case when the number of trades is constrained to be at most K. Although we maintain assumptions **A1-A4** for simplicity, **A1**, **A3** and **A4** can be relaxed without much additional effort.

2.4.1 Unconstrained Return-Optimal Trading Strategies

First we give the main definitions that we will need in the dynamic programming algorithm to compute the optimal strategy. Consider a return-optimal strategy for the first m time periods, $[t_1, t_m]$. Define $\mathcal{S}[m, 0]$ ($\mathcal{S}[m, 1]$) to be a return-optimal strategy over the first m periods ending in bond (stock) at time t_m. For $\ell \in \{0, 1\}$, let $\mu[m, \ell]$ denote the return of $\mathcal{S}[m, \ell]$ over $[t_1, t_m]$, i.e., $\mu[m, \ell] = \mu(\mathcal{S}[m, \ell])$. Let $\textsc{Prev}[m, \ell]$ denote the penultimate position of the optimal strategy $\mathcal{S}[m, \ell]$ just before the final time period t_m.

The optimal strategy $\mathcal{S}[m, \ell]$ must pass through either bond or stock at time period $m - 1$. Thus, $\mathcal{S}[m, \ell]$ must be the extension of one of the optimal strategies $\{\mathcal{S}[m-1, 0], \mathcal{S}[m-1, 1]\}$ by adding the position ℓ at time period t_m. More specifically, $\mathcal{S}[m, \ell]$ will be the extension that yields the greatest total return. Using (2.2) and (2.3), we have that

$$\begin{aligned}\mu(\{\mathcal{S}[m-1, 0], \ell\}) &= \mu[m-1, 0] + \hat{s}_m \ell - \hat{f}_S \ell, \\ \mu(\{\mathcal{S}[m-1, 1], \ell\}) &= \mu[m-1, 1] + \hat{s}_m \ell - \hat{f}_B (1 - \ell).\end{aligned}$$

Since $\mu[m, \ell]$ is the maximum of these two values, we have the following recursion,

$$\mu[m, \ell] = \max \left\{ \mu[m-1, 0] + \hat{s}_m \ell - \hat{f}_S \ell,\ \mu[m-1, 1] + \hat{s}_m \ell - \hat{f}_B (1 - \ell) \right\}.$$

The position of the optimal strategy $\mathcal{S}[m, \ell]$ just before time period m is given by the ending position of the strategy that was extended. Thus,

$$\textsc{Prev}[m, \ell] = \begin{cases} 0 & \text{if } \mu[m-1, 0] + \hat{s}_m \ell - \hat{f}_S \ell \geq \mu[m-1, 1] + \hat{s}_m \ell - \hat{f}_B (1 - \ell), \\ 1 & \text{otherwise.} \end{cases}$$

If we already know $\mu[m-1, 0]$ and $\mu[m-1, 1]$, then we can compute $\mu[m, \ell]$ and $\textsc{Prev}[m, \ell]$ for $\ell \in \{0, 1\}$ in constant time. Further, we have that $\mu[1, 0] = 0$ and $\mu[1, 1] = \hat{s}_1 - \hat{f}_S$, and so, by a straight forward induction, we can prove the following lemma.

Lemma 2.4.1 *$\textsc{Prev}[m, \ell]$ for all $\ell \in \{0, 1\}$ and $m \leq n$ can be computed in $O(n)$.*

The optimal strategy $\mathcal{T}_\mu$ is exactly $\mathcal{S}[n, 0]$. $\textsc{Prev}[n, 0]$ gives the position at t_{n-1}, and the optimal way to reach $\textsc{Prev}[n, 0]$ at t_{n-1} is given by optimal strategy $\mathcal{S}[n-1, \textsc{Prev}[n, 0]]$. Continuing backward in this fashion, it is easy to verify that we can reconstruct the full strategy $\mathcal{T}_\mu$ using the following backward recursion:

$$\begin{aligned}\mathcal{T}_\mu[n] &= 0, \\ \mathcal{T}_\mu[m] &= \textsc{Prev}[m+1, \mathcal{T}_\mu[m+1]], \text{ for } 1 \leq m < n.\end{aligned}$$

Thus, a single backward scan is all that is required to compute $\mathcal{T}_\mu[i]$ for all $i \in \{1, \ldots, n\}$, which is linear time, and so we have proved the first part of Theorem 2.3.2. Further, it is clear that the algorithm requires memory that is linear in n to store $\textsc{Prev}[m, \ell]$. While we have assumed that the algorithm works with excess returns, the optimal strategy does not depend on this assumption, thus the algorithm works correctly even with the actual return sequences. The generalization of this algorithm to $N > 2$ instruments is straightforward by suitably generalizing a trading strategy. $\mathcal{S}[m, \ell]$ retains its definition, except now $\ell \in \{0, \ldots, N-1\}$. To compute $\mu[m, \ell]$ will need to take a maximum over N terms depending on $\mu[m-1, \ell']$, and so the algirithm will have runtime $O(Nn)$.

One concern with the unconstrained optimal strategy is that it may make too many trades. It is thus useful to compute the optimal strategy that makes at most a given number of trades. We discuss this next.

2.4.2 Constrained Return-Optimal Strategies

We suppose that the number of trades is constrained to be at most K. It is more convenient to consider the number of jumps k, which we define as the sum of the number of trades entered and the number exited. For a valid trading strategy, the number of trades entered equals the number of trades exited, so $k = 2K$. Analogous to $\mathcal{S}[m, \ell]$ in the previous section, we define $\mathcal{S}[m, k, \ell]$ to be the optimal trading strategy to time period t_m that makes at most k jumps ending in instrument ℓ. Let $\mu[m, k, \ell]$ be the return of strategy $\mathcal{S}[m, k, \ell]$, and let $\textsc{Prev}[m, k, l]$ store the pair (k', ℓ'), where ℓ' is the penultimate position of $\mathcal{S}[m, k, \ell]$ at t_{m-1} that leads to the end position ℓ, and k' is the number of jumps made by the optimal strategy to time period t_{m-1} that was extended to $\mathcal{S}[m, k, \ell]$.

The algorithm once again follows from the observation that the the optimal strategy $\mathcal{S}[m, k, \ell]$ must pass through either bond or stock at t_{m-1}. A complication is that if the penultimate position is bond and $\ell = 0$, then at most k jumps can be used to get to thhe penultimate position, however, if $\ell = 1$, then only at most $k-1$ jumps may be used. Similarily if the penultimate position is stock. We thus get the following recursion,

$$\begin{aligned}
\mu[m, k, 0] &= \max\left\{\mu[m-1, k, 0],\ \mu[m-1, k-1, 1] - \hat{f}_B\right\}, \\
\mu[m, k, 1] &= \max\left\{\mu[m-1, k-1, 0] + \hat{s}_m - \hat{f}_S,\ \mu[m-1, k, 1] + \hat{s}_m\right\}.
\end{aligned}$$

This recursion is initialized with $\mu[m, 0, 0] = 0$ and $\mu[m, 0, 1] = \mathsf{NULL}$ for $1 \le m \le n$. Once $\mu[m, k, \ell]$ is computed for all m, ℓ, then the above recursion allows us to compute $\mu[m, k+1, \ell]$ is computed for all m, ℓ. Thus, the computation of $\mu[m, k, \ell]$ for $1 \le m \le n$, $0 \le k \le 2K$ and $\ell \in \{0, 1\}$ can be accomplished in $O(nK)$. Once again, the strategy that was extended gives $\textsc{Prev}[m, k, \ell]$,

$$\begin{aligned}
\textsc{Prev}[m, k, 0] &= \begin{cases} (k, 0) & \text{if } \mu[m-1, k, 0] > \mu[m-1, k-1, 1] - \hat{f}_B, \\ (k-1, 1) & \text{otherwise.} \end{cases} \\
\textsc{Prev}[m, k, 1] &= \begin{cases} (k-1, 0) & \text{if } \mu[m-1, k-1, 0] + \hat{s}_m - \hat{f}_S > \mu[m-1, k, 1] + \hat{s}_m, \\ (k, 1) & \text{otherwise.} \end{cases}
\end{aligned}$$

Since computing $\mu[m, k, \ell]$ immediately gives $\textsc{Prev}[m, k, \ell]$, we have the following lemma,

Lemma 2.4.2 *$\textsc{Prev}[m, k, \ell]$ for all $\ell \in \{0, 1\}$, $m \le n$ and $k \le 2K$ can be computed in $O(nK)$.*

T_μ^K is given by $S[n, 2K, 0]$, and the full strategy can be reconstructed in a single backward scan using the following backward recursion (we introduce an auxilliary vector κ),

$$\begin{aligned} \mathcal{T}_\mu^K[n] &= 0, \\ (\kappa[n-1], \mathcal{T}_\mu^K[n-1]) &= \textsc{Prev}[n, 2K, \mathcal{T}_\mu^K[n]), \\ (\kappa[m], \mathcal{T}_\mu^K[m]) &= \textsc{Prev}[m+1, \kappa[m+1], \mathcal{T}_\mu^K[m+1]), \text{ for } 1 \le m < n-1. \end{aligned}$$

Since the algorithm needs to store $\textsc{Prev}[m, k, \ell]$ for all m, k, the memory requirement is $O(nK)$. Once again, it is not hard to generalize this algorithm to work with N instruments, and the resulting run time will be $O(nNK)$.

2.5 Collecting Training Dataset: Sterling-Optimal Trading Strategies

It will first be useful to discuss some of the MDD properties of the return-optimal strategy $\mathcal{T}_\mu$, as these properties will have implications on our algorithm to determine Sterling-optimal strategies. For a strategy $\mathcal{T}$, it is useful to define the cumulative return series, $C_\mathcal{T}[i]$ as the sum of the returns, $C_\mathcal{T}[i] = \sum_{j=1}^{i} r_\mathcal{T}[j]$. Note that $\mu(\mathcal{T}_\mu) = C_{\mathcal{T}_\mu}[n] \geq C_\mathcal{T}[n] = \mu(\mathcal{T})$ for any strategy $\mathcal{T}$. The equity curve is given by $\mathcal{E}_\mathcal{T}[i] = exp\left(C_\mathcal{T}[i] + \sum_{j=1}^{i} b_j\right)$.

First, we will upper bound $MDD(\mathcal{T}_\mu)$, because this in turn serves as an upper bound for the MDD of the Sterling-optimal strategy,

Lemma 2.5.1 $MDD(\mathcal{T}_{\mathsf{Strl}}) \leq MDD(\mathcal{T}_\mu)$.

Proof: By definition, $\frac{\mu(\mathcal{T}_{\mathsf{Strl}})}{MDD(\mathcal{T}_{\mathsf{Strl}})} \geq \frac{\mu(\mathcal{T})}{MDD(\mathcal{T})}$ for any $\mathcal{T}$. Thus, $MDD(\mathcal{T}_{\mathsf{Strl}}) \leq \frac{\mu(\mathcal{T}_{\mathsf{Strl}})}{\mu(\mathcal{T})} MDD(\mathcal{T})$ for any $\mathcal{T}$. Choosing $\mathcal{T} = \mathcal{T}_\mu$ and noting that $\mu(\mathcal{T}_{\mathsf{Strl}}) \leq \mu(\mathcal{T}_\mu)$, we obtain the desired result. ■

Since the cost (in terms of the cumulative return) of entering and exiting a trade is $-(\hat{f}_S + \hat{f}_B)$, no segment of the optimal trading strategy $\mathcal{T}_\mu$ should lose more than this in return.

Lemma 2.5.2 *For any $i < j$, $C_{\mathcal{T}_\mu}[j] - C_{\mathcal{T}_\mu}[i] \geq -(\hat{f}_S + \hat{f}_B)$.*

Proof: Suppose, for contradiction, that for some $i < j$, $C_{\mathcal{T}_\mu}[j] - C_{\mathcal{T}_\mu}[i] < -(\hat{f}_S + \hat{f}_B)$. By setting $\mathcal{T}_\mu[i+1], \ldots, \mathcal{T}_\mu[j]$ to be all equal to 0, it is easy to verify that the cumulative return of the strategy must increase, which contradicts the optimality of $\mathcal{T}_\mu$. ■

For technical convenience, we will assume that the transactions cost when entering a trade is assigned to the time period prior to the entry, and the transactions cost when exiting a trade is assigned to the time period after the trade. Note that just prior to entering, the position is 0 and so it will now have a return of $-\hat{f}_S$, and just after exiting, the position is 0, and will now have a return of $-\hat{f}_B$.

Let $f_{sp} = \hat{f}_S + \hat{f}_B$. From Lemma 2.5.2, no segment of the optimal strategy can lose more than f_{sp}, and so this immediately gives an upper bound on $MDD(\mathcal{T}_\mu)$. For the trivial strategy that makes no trades, the MDD is 0. If a strategy makes exactly one trade, then there is a drawdown of at least $\hat{f}_S$ at the begining, and of at least $\hat{f}_B$ at the end. If at least two trades are made, then there is a drawdown of at least f_{sp} between the exit of one trade and the entry of another, and since the drawdown cannot exceed f_{sp}, the MDD must therefore equal f_{sp}. We thus have the following lemma.

Lemma 2.5.3 (MDD of $\mathcal{T}_\mu$) *If $\mathcal{T}_\mu$ makes no trades, $MDD(\mathcal{T}_\mu) = 0$. If $\mathcal{T}_\mu$ makes one trade, $\max\{\hat{f}_S, \hat{f}_B\} \leq MDD(\mathcal{T}_\mu) \leq f_{sp}$. If $\mathcal{T}_\mu$ makes at least two trades, $MDD(\mathcal{T}_\mu) = f_{sp}$.*

Note that if we relax assumption **A1**, then by legging into a trade, it may be possible to decrease the drawdown, in which case Lemma 2.5.3 would no longer be valid. We are now ready to discuss the $O(n \log n)$ algorithms to obtain Sterling-optimal trading strategies. First we will consider unconstrained Sterling optimal strategies, and then we will require number of trades $\leq K$.

2.5.1 Unconstrained Sterling-Optimal Strategies

For a degenerate trading system with all returns equal to zero, we define its Sterling ratio as 1. The only trading system with a MDD of 0 is a degenerate trading system, so with this definition, the Sterling ratio is defined for all possible trading systems. The computation of the Sterling-optimal trading system breaks down into three cases, according to the number of trades its makes:

i. Sterling-optimal that makes zero trades. In this case Sterling Ratio is 1.

ii. Sterling-optimal that makes one trade. Then optimal trading strategy contains a single interval of 1's.

iii. Sterling-optimal that makes at least two trades. Any trading system that makes at least two trades has an $MDD \geq f_{sp}$. Since $MDD(\mathcal{T}_\mu) \leq f_{sp}$ (Lemma 2.5.3), $\mathcal{T}_\mu$ has the smallest MDD among all such systems. Since it also has the highest total return, we conclude that if the Sterling-optimal system makes at least two trades, then $\mathcal{T}_{\mathsf{Strl}} = \mathcal{T}_\mu$.

The first case is trivially computed. The third case, i.e., the Sterling optimal strategy that makes at least two trades can be computed in linear time using the dynamic programming algorithm to compute $\mathcal{T}_\mu$. If we also compute the Sterling-optimal system that makes exactly one trade, then, we solve our problem by taking the case with the maximum Sterling ratio. We now focus on finding the trading strategy that makes only one trade and has greatest Sterling Ratio among all such strategies.

Let $\mathcal{T}$ be a strategy that makes exactly one trade. The trade interval is the interval of time periods, $[t_i, t_j]$ on which $\mathcal{T} = 1$, i.e., the trade interval is an interval of 1's in the trading strategy. An elementary algorithm that considers all the $O(n^2)$ possible trade intervals, picking the best is a quadratic time algorithm. The remainder of this section is devoted to providing an algorithm which computes such a strategy in $O(n \log n)$ time, which will complete the proof of the first part of Theorem 2.3.3. In fact the algorithm that we present is a much more general algorithm that computes the single interval that optimizes a general class of optimality criteria. This algorithm will be useful when we discuss the Sharpe-optimal strategy.

Consider a consecutive sequence of time periods $t_i, t_{i+1}, \ldots, t_{i+k}$ where $k \geq 1$, with all the excess returns non-negative and the last one positive, i.e., $\hat{s}_i \geq 0, \hat{s}_{i+1} \geq 0, \ldots, \hat{s}_{i+k} > 0$.

Lemma 2.5.4 *Either the optimal single trade interval does not intersect these time periods, or an optimal single interval can be chosen to contain this interval.*

Proof: Suppose that $\mathcal{T}[i+j] = 1$ and $\mathcal{T}[i+j+1] = 0$ for some $0 \leq j < k$. Extend the trading interval by setting $\mathcal{T}[i+j+1] = 1, \ldots, \mathcal{T}[i+k] = 1$, which adds positive return, without increasing the MDD, contradicting the optimality of the original interval. On the other hand, suppose that $\mathcal{T}[i+j] = 1$ and $\mathcal{T}[i+j-1] = 0$ for some $0 < j \leq k$. Once again, by extending the trading interval, setting $\mathcal{T}[i] = 1, \ldots, \mathcal{T}[i+j] = 1$, we add non-negative returns, without increasing the MDD hence this new interval is at least as good as the previous interval. ■

A similar result holds for a sequence of consecutive negative time periods, $t_i, t_{i+1}, \ldots, t_{i+k}$ where $k \geq 1$, with $\hat{s}_i \leq 0, \hat{s}_{i+1} \leq 0, \ldots, \hat{s}_{i+k} < 0$. If an optimal trading interval only intersects part of these time periods, this intersection can be removed without decreasing the Sterling ratio. Thus, by Lemma 2.5.4, any sequence of time periods with all returns non-negative (non-positive) can be

condensed into a single time period, $t'_i = t_i + \cdots + t_{i+k}$, with $\hat{s}'_i = \hat{s}_i + \cdots + \hat{s}_{i+k}$. Further, this operation can be performed in linear time on the entire excess return sequence, so from now on we assume without loss of generality that the excess return sequence consists of alternating time periods of strictly positive and negative excess returns. If $\hat{s}_i < 0$, then t_i cannot be the first 1 of a trade, since by entering one time period later, we exclude only this negative return and do not increase the MDD. Similarily, it cannot be the last 1 of a trade.

Lemma 2.5.5 *The first 1 and the last 1 of the optimal trade interval must occus at time periods t_f and t_l for which $\hat{s}_f > 0$ and $\hat{s}_l > 0$.*

The pictorial illustration of this lemma is given below on Figure 2.2 where we show the cumulative return curve. The time *instants* a_i are the possible entry points, and the time instants b_i are the possible exit points. Let the alternating sequence of entry and exit points be $\{a_1, b_1, a_2, b_2, \ldots, a_k, b_k\}$ (a_i are the entry points, and b_i are the exit points). Note that after the preprocessing into alternating intervals, $k \leq \lceil n/2 \rceil$. Notice that without loss of generality, we can throw away the first interval if it has a negative return, as it can never be an entry point, and the last interval if it has a negative return, for a similar reason. The optimal trade interval will be of the form (a_t, b_{t+r}), $r \geq 0$.

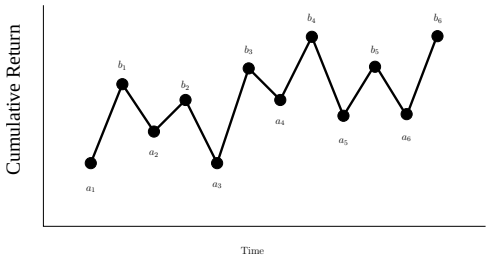


Figure 2.2: Possible entry and exit points.

Our algorithm for finding the Sterling-optimal interval will be to consider every possible starting point a_i, and find the Sterling-optimal interval with this point as starting point (i.e. we have to find the end point of this interval). As the algorithm proceeds, we keep track of the best entry point (and its corresponding exit point). The entry points a_t are processed from right to left. After processing a new entry point a_t, we will modify the alternating sequence to facilitate faster processing of the remaining points. More specifically, we will delete the processed entry point and add a weight to the edge between b_{t-1} and b_t to preserve all the necessary information – we cannot simply delete the entry point a_t, since we have to keep track of maximum MDD that occurs in our activity interval. Since between b_{t-1} and a_t we have a drawdown of $b_{t-1} - a_t$, we need to keep this information in an edge weight connecting b_{t-1} to b_t. Please note that at any stage of algorithm edge weight connecting b_{t-1} to b_t will be equal to the MDD of the interval $[b_{t-1}, b_t]$ **and this MDD is realized on prefix of** $[b_{t-1}, b_t]$, i.e. $MDD([b_{t-1}, b_t]) = C(b_{t-1}) - C(x)$, for some $x \in [b_{t-1}, b_t]$ **- Invariant (*)**. We will show this weight attached to b_t in parentheses, $(w_t)b_t$, where the value of w_t appearing in parentheses indicates the weight.

We start our backward scan at the last entry point, a_k, for which there is only one possible interval (a_k, b_k). We update the weight $w_k \leftarrow b_{k-1} - a_k$, store the current best interval $(a_k, b_k, \mathsf{Strl}_k)$, and delete the possible start point a_k from the sequence to give the processed sequence $\{a_1, b_1, ..., a_{k-1}, b_{k-1}, (w_k)b_k\}$. Note that $(a_k, b_k, \mathsf{Strl}_k)$ is a one-step trade, but we keep it here for simplicity. We now proceed to a_{k-1} and so on.

In the general case, suppose we have processed (backwards) all the entry points up to (including) the entry point a_{t+1}, and are currently processing entry point a_t. The weighted exit sequence is $\{a_1, b_1, ..., a_t, b_t, (w_{t+1})b_{t+1}, \ldots, (w_{t+m})b_{t+m}\}$. $b_t, \ldots, b_{t+m}$ are the possible exit points – note that

$t+m$ may not equal k due to possible deletion of points which we discuss below. Assume that $\{b_{t+1} < \ldots < b_{t+m}\}$: this is true after we have processed the first start point (since the sequence consists only of one point), and we will maintain this condition by induction. If $b_t < b_{t+1}$, then the entire sequence of exit points is monotonically increasing. On the other hand, if $b_t \geq b_{t+1}$, b_{t+1} need not be considered as an exit point for any optimal interval with entry point a_t or earlier, because by stopping earlier at b_t, we do not decrease the cumulative return, nor increase the MDD. Hence, we can delete the possible exit point b_{t+1}. However, we must now update the weight in $(w_{t+2})b_{t+2}$ to store the new drawdown between b_t and b_{t+2} as follows $w_{t+2} \leftarrow \max\{w_{t+1}, w_{t+2} + b_t - b_{t+1}\}$.

Lemma 2.5.6 *If $b_t \geq b_{t+1}$, the weighted exit sequence is updated as follows:*

$$a_t, b_t, (w_{t+1})b_{t+1}, \ldots, (w_{t+m})b_{t+m} \rightarrow a_t, b_t, (\max\{w_{t+1}, w_{t+2} + b_t - b_{t+1}\})b_{t+2}, \ldots, (w_{t+m})b_{t+m}$$

The correctness of the weight updating rule above follows from the Invariant (*). Please note that the transformation above preserves the Invariant (*).

This process is continued until the next exit point after b_t is either above b_t or there are no remaining exit points after b_t. In either event, the new sequence of exit points available for a_t is strictly monotonically increasing (by the induction hypothesis). Observe that any deletion of a possible exit point is a constant time operation. Further, since each deletion drops a point from the set $\{b_1, \ldots, b_k\}$, there can be at most $k-1$ such deletions during the course of the *entire* algorithm. We thus have the following lemma.

Lemma 2.5.7 *When a_t is processed by the algorithm, the exit points $b_t < b_{t+1} < \cdots$ are monotonically increasing. The cost of maintaining this condition for the* entire *algorithm is $O(k)$ operations.*

When processing a_t, the weighted exit sequence is $\{a_1, b_1, ..., a_t, b_t, (w_{t+1})b_{t+1}, \ldots, (w_{t+m})b_{t+m}\}$. Suppose that $w_{t+2} < w_{t+3} < \ldots < w_{t+m}$. Initially this sequence is the empty sequence and so this condition holds, and once again, by induction we will ensure that this condition will always hold. If $w_{t+1} \geq w_{t+2}$, then no optimal interval can have entry point a_t or earlier, and exit at b_{t+1}, because otherwise by exiting at b_{t+2}, since $b_{t+2} > b_{t+1}$ (Lemma 2.5.4), the MDD is not increased, however the total return is increased. Thus if $w_{t+1} \geq w_{t+2}$, we can remove the possible exit point b_{t+1} from the weighted exit sequence and update $w_{t+2} \leftarrow w_{t+1}$. Please note that this transformation also preserves the Invariant (*).

Lemma 2.5.8 *If $w_{t+1} \geq w_{t+2}$, the weighted exit sequence is updated as follows:*

$$a_t, b_t, (w_{t+1})b_{t+1}, \ldots, (w_{t+m})b_{t+m} \rightarrow a_t, b_t, (w_{t+1})b_{t+2}, \ldots, (w_{t+m})b_{t+m}$$

We continue removing exit points in this way until either there is only one weight left in the weighted exit sequence, or all the weights are strictly monotonically increasing (by induction).

Suppose that $w_{t+1} \leq f$. In this case, we observe that b_t cannot be the exit of an optimal interval with entry a_{t-r}, where $r \geq 0$. To see this note that if $b_t - a_{t-r} - \hat{f}_S < 0$, then the return of this interval is negative and this interval cannot be an optimal interval. If $b_t - a_{t-r} - \hat{f}_S \geq 0$ then since the interval already has MDD of at least f, so by continuing to b_{t+1}, we do not increase the MDD but strictly increase the return, hence it cannot be optimal to exit at b_t.

Lemma 2.5.9 *Let*

$$a_t, b_t, (w_{t+1})b_{t+1}, \ldots, (w_{t+m})b_{t+m}$$

be the weighted exit sequence and let $T \in [t, \ldots, t+m]$ be such an index that $w_T < f$ and $w_{T_1} \geq f$. Then no Sterling-optimal interval that starts at a_t or earlier can exit at $\{b_t, \ldots, b_{T-1}\}$.

Lemma 2.5.10 *When a_t is processed by the algorithm, the weights w_{t+1} are monotonically increasing. The cost of mainitaining this condition for the* entire *algorithm is $O(k)$ operations.*

We thus assume from now on that when processing entry point a_t, the weighted exit sequence $\{a_1, b_1, ..., a_t, b_t, (w_{t+1})b_{t+1}, \ldots, (w_{t+m})b_{t+m}\}$ with $m \geq 0$ satisfies the conditions of Lemmas 2.5.7 and 2.5.10. The first available exit gives the trade interval (a_t, b_T). If $b_T - a_t - f_{sp} \leq 0$, i.e., if the return is not positive, then this cannot possibly be an optimal interval. Otherwise, the Sterling Ratio is

$$\mathsf{Strl}_t = \frac{b_T - a_t - f_{sp}}{f},$$

where $f_{sp} = \hat{f}_S + \hat{f}_B$ and $f = \max\{\hat{f}_S, \hat{f}_B\}$. Now consider the exit points b_{T+r}, $r > 0$, and suppose that $b_T - w_{T+1} < a_t$. No trade with entry at a_t, exiting at b_{T+r} with $r > 0$ can possibly be optimal, since we could always enter later at $b_T - w_{T+1}$, exit at b_{T+r}, and strictly increase the return without increasing the drawdown. We are thus done with processing the entry point a_t, and we can proceed to a_{t-1} after updating weight w_t and comparing $\frac{b_T - a_t - f_{sp}}{f}$ with the current champion. Similarly, if $b_{\bar{t}} - w_{\bar{t}+1} < a_t$ for some $\bar{t} \in [t, \ldots, T-1]$, we are done with the starting point a_t, and we can proceed to a_{t-1} after updating weight w_t. We assume that at any stage of the algorithm we keep the value of $min_{\bar{t} \in [t,...,T-1]} b_{\bar{t}} - w_{\bar{t}+1}$ and thus this check can be done in constant time for any given point a_t. Thus, without loss of generality, we can assume that $b_T - w_{T+1} \geq a_t$ and $b_{\bar{t}} - w_{\bar{t}+1} \geq a_t$ for all $\bar{t} \in [t, \ldots, T-1]$. Since $w_{T+1} \geq f$, we conclude that $b_T - a_t \geq f$. A trade, entering at a_t and exiting at b_{T+r}, $r > 0$ has total return $b_{T+r} - a_t - f_{sp}$. The next lemma gives the MDD.

Lemma 2.5.11 *Assume that $b_T - w_{T+1} \geq a_t$ and $b_{\bar{t}} - w_{\bar{t}+1} \geq a_t$ for all $\bar{t} \in [t, \ldots, T-1]$. The trade (a_t, b_{T+r}), $r > 0$, has $MDD = w_{T+r}$.*

Proof: The local maxima of the cumulative return sequence for this trade are $\{0, b_t - a_t - \hat{f}_S, , \ldots, b_T - a_t - \hat{f}_S, \ldots, b_{T+r} - a_t - \hat{f}_S\}$. Since $b_{\bar{t}} - w_{\bar{t}+1} - a_t \geq 0 \; \forall \bar{t} \in [t, \ldots, T-1]$, MDD of the interval $[a_t, b_T]$ is equal to $\hat{f}_S$.
Since $b_T - a_t - \hat{f}_S \geq 0$ and since the sequence of exit points is strictly increasing, $MDD([a_t, b_{T+r}]) = max(MDD([a_t, b_T]), MDD([b_T, b_{T+r}])) = max(\hat{f}_S, MDD([b_T, b_{T+r})), \hat{f}_B)$ where $\hat{f}_B$ is the draw down after the last point b_{T+r}.
Since the drawdown at any time in a trade is given by the the diffenence between the previous maximum and the current cumulative return, $MDD([b_T, b_{T+r}))$ is at most $\max_{i \in [1,r]} w_{t+i}$. Since the weights are monotonically increasing, we see that this drawdown $\leq w_{t+r}$, which is achieved in the interval (b_{t+r-1}, b_{t+r}). Since $w_{t+r} \geq f = max(\hat{f}_S, \hat{f}_B) \; \forall r > 0$, we conclude $MDD([a_t, b_{T+r}]) = w_{t+r}$. ■

Summary: For entry point a_t, the sequence of exit points b_{T+r}, $r \in [0, m]$ have cumulative returns $c_r = b_{T+r} - a_t - f_{sp}$ and MDD's $d_r = w_{T+r}$ for $r > 0$ and $d_0 = f$. The task is to maximize c_r/d_r with respect to r. The sequences $\{c_r\}$ and $\{d_r\}$ are both strictly increasing. We now describe a general algorithm for performing such a maximization.

2.5.2 Maximizing $\frac{c_r}{d_r}$

On the two dimensional plane, consider the set of points $P'_m = \{(d_r, c_r)\}_{r=0}^m$. Let $\mathbf{p}' = (0, 0)$. Then the slope of the line joining $\mathbf{p}$ to (d_r, c_r) is exactly the Sterling ratio of the trade (a_t, b_{t+r}).

Thus, finding the optimal trade is equivalent to finding the upper-touching point from $\mathbf{p}$ to the convex hull of the set of points P'_m (see Figure 2.3 below). We call $\mathbf{p}$ the source point. We get the same result if we define $P_m = \{(d_r, b_{t+r})\}_{r=0}^m$, $\mathbf{p} = (0, a_t + f_{sp})$. Given the convex hull, this touching line can found in time $O(\log A)$ where A is number of the points on the convex hull, [14]. It is easy to see that $A \leq m + 1 \leq \lceil n/2 \rceil$. This algorithm that computes the touching point in $O(\log A)$ requires the ability to efficiently search through the convex hull. We accomplish this by maintaining the convex hull as a doubly linked list where each point in the convex hull maintains $O(\log A)$ pointers to other points in the convex hull. More specifically, point i points forward to points at position 2^j in the convex hull of the points $\{(d_r, c_r)\}_{r=i}^m$, where $j \geq 1$. Each point also maintains backward pointers to any point that points forward to it. At point j, the backward pointers specific to the convex hull starting at point $i < j$ are maintained separatly for each i so that constant time access to this set of pointers is possible. An array of an array of pointers suffices. It is clear that the worst case memory requirement is $O(m \log m)$ pointers. We now discuss the main operations we would like to be able to do on our set of points $\{(d_r, c_r)\}$ and the point $\mathbf{p}$ and still be able to compute the upper tangent line efficiently. First we recall that the d_r are monotonically increasing. Assume that each point (d_r, c_r) stores all the necessary forward and backward pointers. We also assume that the point (d_r, c_r) stores the pointer to the point $\mathsf{nxt}(r)$, which is the next point in the convex hull if all the points $d_0, \ldots, d_{r-1}$ were removed – note that in this case, d_r becomes leftmost, and so must be in the convex hull. We will see that these assumptions are maintained inductively. Note that the initial convex hull with all the points is given by (d_0, c_0) followed by the points pointed to by $\mathsf{nxt}(0), \mathsf{nxt}(\mathsf{nxt}(0)), \ldots$. We would like to perform the following operations on our points and still be able to compute the upper tangent point efficiently:

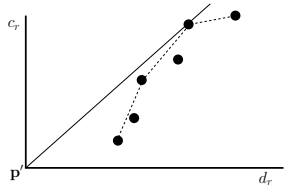


Figure 2.3: Upper-touching point from $\mathbf{p}$ to the convex hull of the set of points P'_m.

(1) Translate $\mathbf{p}$ by some given vector $\mathbf{v}$.

(2) Translate all the points in $\{(d_r, c_r)\}$ by some given vector $\mathbf{v}$.

(3) Remove the leftmost point (d_0, c_0).

(4) Add a new leftmost point (d_{-1}, c_{-1}).

Lemma 2.5.12 *Assuming that* $\mathbf{p}$, $\{(d_r, c_r, \mathsf{nxt}_r)\}_{r=1}^m$ *are given, all the operations in (1)-(4) above can be accomplished in time* $O(\log m)$. *Further, in the event that a point is added or deleted, all necessary pointers are maintained.*

Proof: Let $A = O(m)$ be the size of the current convex hull. For (1), we do not change the points at all, we simply compute the new tangent point for $\mathbf{p}' = \mathbf{p}+\mathbf{v}$, which can be accomplished in $O(\log A)$. (2) is equivalent to shifting $\mathbf{p}$ by $-\mathbf{v}$. To prove (3), notice that if we remove (d_0, c_0), then the new leftmost point becomes (d_1, c_1) and we immediately have the new convex hull $\mathsf{nxt}(1), \mathsf{nxt}(\mathsf{nxt}(1)), \ldots$. Thus we can find the new upper tangent point in $O(\log A') = O(\log m)$, where A' is the size of the new convex hull. Further, deleting (d_0, c_0) requires first deleting the backward pointers of the points that it points to $O(\log A)$, and then deleting the point itself, and its forward pointers (it has no backward pointers), $O(\log A)$. To prove (4), note that when we add (d_{-1}, c_{-1}), $\mathsf{nxt}(-1)$ is exactly the upper tangent point from $\mathbf{p}' = (d_{-1}, c_{-1})$ to the current convex hull. This can be computed in $O(\log A)$. We now need to add all the necessary pointers into the data structure. For each forward pointer we add, we will add the coresponding backward pointer as well. We need a pointer at position 2^j in the convex hull of (d_{-1}, c_{-1}). But this is exactly the point at position $2^j - 1$ in the convex hull of point $\mathsf{nxt}(-1)$. Since $\mathsf{nxt}(-1)$ maintains a pointer to point 2^j in its convex hull, and this point will have a backward pointer by one step of this same convex hull, we can construct the forward an backward pointer for point 2^j in the convex hull of (d_{-1}, c_{-1}) in constant time, requiring total time $O(\log A') = O(\log m)$ to construct all the new forward and backward pointers, where A' is the size of the new convex hull. We now construct the new upper tangent point from $\mathbf{p}$ to the new convex hull of (d_{-1}, c_{-1}) in $O(\log A')$ time. The entire process is therefore $O(\log m)$. ■

The algorithm that we have just described is a general purpose algorithm for efficiently maintaining the upper tangent point to *any* set of points, as long as only a limited set of operations is allowed on the set of points and the source point. We will now see that these limited operations are all that is needed for maximizing the Sterling ratio.

Suppose that point a_t has been processed in the algorithm – i.e., the upper tangent point (optimal trade with entry at a_t) from $\mathbf{p}_t = (0, a_t + f_{sp})$ to $P_m = \{(d_r, b_{T+r})\}_{r=0}^{m}$ has been computed. Now consider the addition a_{t-1} to construct the new weighted exit sequence. Delete the leftmost point (d_0, b_T) from the convex hull. Lets consider all posible operations that may take place. There are several possibilities.

i. $b_{t-1} \geq b_t$. We remove (leftmost) points b_{t+i}, $i \geq 0$, until $b_{t-1} < b_{t+i+1}$, and the new weight w'_{t+i+1} may have increased (Lemma 2.5.6). Deleting of points $b_t + i$ from the weighted exit sequence doesn't implies changes of the convex hull until $t + i \geq T$. After this point, deleting one point $b_T + i$, $i \geq 0$ from the weighted exit sequence followed by deletion of corresponding leftmost point of the convex hull. At the very end of the sequence of deletions, we have to update the MDD of point b_{t+i+1} from w_{t+i+1} to w'_{t+i+1}, this can be done by deletion of the point (w_{t+i+1}, b_{t+i+1}) and addition of new point (w'_{t+i+1}, b_{t+i+1}). The total number of such removals during the entire algorithm is at most $n - 1$. When condition $b_{t-1} < b_t$ is satisfied, proceed to the next stage.

ii. $b_{t-1} < b_t$.

   ii.1. $w_{t+1} \leq w_t$. We remove (leftmost) points b_{t+i}, $i \geq 0$, until $w_t < w_{t+i+1}$. Deleting of points $b_t + i$ from the weighted exit sequence doesn't implies changes of the convex hull until $t + i \geq T$. After this point, deleting one point $b_T + i$, $i \geq 0$ from the weighted exit sequence followed by deletion of corresponding leftmost point of the convex hull. By

Lemma 2.5.10, the total number of such removals cannot exceed $n-1$ over the course of the entire algorithm. When condition $w_t < w_{t+1}$ is satisfied, proceed to the next stage.

ii.2. $b_{t-1} < b_t$ and $w_t < w_{t+1}$.

(a) $f > w_t$. Add to the convex hull point (f, b_T).

(b) $f < w_t$. Add to the convex hull points (w_t, b_t) and (f, b_{t-1}).

The new source point is $p_{t-1} = (0, a_{t-1} + f_{sp})$, which just corresponds to a shift of $\mathbf{p}_t$, and so once the new convex hull for the new weighted exit sequence is computed, an additional $O(\log n)$ operations are needed to find the new upper tangent point.

The total number of removals in the entire algorithm is $O(n)$. For each new entry point, we have at most a constant number of additions, and since the number of entry points is $O(n)$, we see that the total number of additions is $O(n)$. By Lemma 2.5.12, we have that each operation takes $O(\log n)$ in the worst case, thus the total run time is $O(n \log n)$. Collecting all the results together, we can find the Sterling-optimal strategies making zero, one or more than one trade in $O(n \log n)$ time, completing the proof of the first part of Theorem 2.3.3.

Note that by only maintaining exit points with weight at most some given constant, MDD_0, i.e., by truncating some region of the points to the right, this algorithm is easily extended to computing the Sterling-optimal strategy that uses exactly one trade and has an $MDD \leq MDD_0$.

Proposition 2.5.13 *Given MDD_0, a Sterling-optimal strategy that uses exactly one trade and has $MDD \leq MDD_0$ can be computed in $O(n \log n)$ time.*

This result will be useful when we consider constrained Sterling-optimal strategies.

2.5.3 Constrained Sterling-Optimal Strategies

As with the return-optimal strategies, the unconstrained sterling-optimal strategies may make too many trades. Here, we consider the the Sterling-optimal strategy that makes at most K trades, $\mathcal{T}^K_{\mathsf{Strl}}$. We refer to such strategies as K-Sterling-optimal. First, we present some properties of this strategy, before giving an efficient algorithm to compute it.

A maximal return-optimal strategy $\mathcal{T}^*_\mu$ is a return-optimal whose trade intervals cannot be enlarged. Given any return-optimal strategy $\mathcal{T}_\mu$, in one (linear time) scan from left to right, we can enlarge any trade intervals maximally to the right as long as they keep the same return. Similarily, in one backward scan, we can extend all trade intervals maximally to the left. Since $\mathcal{T}_\mu$ can be computed in linear time, we conclude that

Lemma 2.5.14 *A maximal return-optimal strategy $\mathcal{T}^*_\mu$ can be computed in linear time.*

If any trade interval of a maximal return-optimal strategy is extended in either direction, then the total return must strictly decrease. In the previous section, we gave a $O(n \log n)$ algorithm for computing the Sterling-optimal strategy with exactly 1 trade. We also saw that if the unconstrained Sterling-optimal strategy contains more than 1 trade, then it is $\mathcal{T}^*_\mu$. Fix K, and let the number of trades that $\mathcal{T}^*_\mu$ makes be $K_0 \leq K$. In this case $\mathcal{T}^K_{\mathsf{Strl}} = \mathcal{T}^*_\mu$, and we are done. Thus we only need to consider the case that $1 < K < K_0$. Some important properties of $\mathcal{T}^*_\mu$ are summarized below. When it is clear, We also use $\mathcal{T}^*_\mu$ to refer to the set of trading intervals $\{I_r\}_{r=1}^{K_0}$. Let $C_i = \sum_{j=1}^{i} \hat{s}_j$ denote the cumulative return sequence of the excess returns.

Lemma 2.5.15 *Let $\mathcal{T}_\mu^*$ be maximal return-optimal. Let I be an interval $[t_i, t_j]$.*

i. If $I \in \mathcal{T}_\mu^*$, then, $\sum_{k=i}^{j} \hat{s}_k - f_{sp} \geq 0$ and $MDD(I) \leq f_{sp}$.

ii. Suppose I does not intersect with any interval in $\mathcal{T}_\mu^*$ and let the return of I ($\sum_{k=i}^{j} \hat{s}_k$) be $\mu(I)$. Then, $\mu(I) \leq f_{sp}$. If I is adjacent to some interval of $\mathcal{T}_\mu^*$, then $\mu(I) < 0$. If I is adjacent to two intervals in $\mathcal{T}_\mu^*$, then $\mu(I) < -f_{sp}$.

iii. Let $[t_l, t_r]$ and $[t_{l'}, t_{r'}]$ be two consecutive trade intervals in $\mathcal{T}_\mu^*$, $l \leq r < l' \leq r'$. Then, $C_r - C_{l'} > f_{sp}$, and for all $r < q < l'$, $C_{l'} < C_q < C_r$.

Let $\{(a_i, b_i)\}$ denote the local minima and maxima of $\{C_i\}$, as in the previous section. Any trade of $\mathcal{T}_\mu^*$ or $\mathcal{T}_{\mathsf{Strl}}^K$ must enter (exit) at a local minimum (maximum). Further, the entry (exit) point must be a minimum (maximum) in the trade, otherwise we can shrink the trade, strictly increasing the return without increasing the MDD.

Lemma 2.5.16 *Let $I = [t_l, t_r]$ be a trade interval of $\mathcal{T}_\mu^*$ or $\mathcal{T}_{\mathsf{Strl}}^K$. Then C_l is a local minimum, C_r is a local maximum, and for any k, with $l \leq k \leq r$, $C_l \leq C_k \leq C_r$*

We now give an important inclusion property of the $\mathcal{T}_{\mathsf{Strl}}^K$.

Proposition 2.5.17 *Let $\mathcal{T}_\mu^*$ be a maximal return-optimal trading strategy. There exists a K-Sterling-optimal strategy $\mathcal{T}_{\mathsf{Strl}}^K$, $K > 1$, with the following property: if $I = [t_l, t_r]$ is any trading interval in $\mathcal{T}_{\mathsf{Strl}}^K$, then a prefix of I and a suffix of I are trades in the maximal return-optimal strategy $\mathcal{T}_\mu^*$.*

Proof. First we show that for every trading interval $I^* = [t_a, t_b]$ in $\mathcal{T}_\mu^*$ with $I \cap I^* \neq \emptyset$, one can pick $\mathcal{T}_{\mathsf{Strl}}^K$ such that $I^* \subseteq I$. Suppose to the contrary, that for some I^*, either $t_a < t_l$ and $t_b \geq t_l$ or $t_a \leq t_r$ and $t_b > t_r$. We will extend I without decreasing the Sterling ratio of $\mathcal{T}_{\mathsf{Strl}}^K$ so that $I^* \subseteq I$. Suppose $t_a < t_l$ and $t_b \geq t_l$ (a similar argument holds for $t_a \leq t_r$ and $t_b > t_r$). There are two cases:

i. I^* does not intersects any other interval of $\mathcal{T}_{\mathsf{Strl}}^K$: Applying Lemma 2.5.16 to I^*, we have: $C_a \leq C_l$. Thus by extending I to $[t_a, t_r]$, the return of the interval cannot decrease. Since $MDD(I^*) \leq f_{sp}$, this extension cannot increase the $MDD(\mathcal{T}_{\mathsf{Strl}}^K)$, since we already have that $MDD(\mathcal{T}_{\mathsf{Strl}}^K) \geq f_{sp}$.

ii. I^* intersects with the previous trading interval of strategy $\mathcal{T}_{\mathsf{Strl}}^K$: $\exists I' = [t_{l'}, t_{r'}] \in \mathcal{T}_{\mathsf{Strl}}^K$ such that $t_a \leq t_{r'} < t_l$. Since $[t_{r'+1}, t_{l-1}]$ is a subinterval of I^*, $\sum_{j=r'+1}^{l-1} \hat{s}_j \geq -f_{sp}$ (Lemma 2.5.15). If we merge I and I' by adding the interval $[t_{r'+1}, t_{l-1}]$ into $\mathcal{T}_{\mathsf{Strl}}^K$, we save on the transaction cost of f_{sp}, and so the total return will not decrease. We show that the MDD has not increased. Since $C_{r'}$ is a maximum in $[t_{l'}, t_{r'}]$, the drawdown for all points in $[t_{r'+1}, t_l]$ is at most f_{sp}. Since C_l is a minimum in $[t_l, t_r]$, we conclude that the drawdown for any point in $[t_l, t_r]$ is at most $\max\{f_{sp}, MDD(I)\}$. Since $MDD(\mathcal{T}_{\mathsf{Strl}}^K) \geq f_{sp}$, we conclude that this merger does not increase the MDD.

Note that $\mu(I) \geq 0$ otherwise we improve the return of $\mathcal{T}_{\mathsf{Strl}}^K$ by removing I, without increasing the MDD, ans so $\mathcal{T}_{\mathsf{Strl}}^K$ cannot possibly be optimal. Thus, without loss of generality, we assume that the return of I is positive. Suppose that $I \cap \mathcal{T}_\mu^* = \emptyset$. Then, by adding I to $\mathcal{T}_\mu^*$, we strictly increase

the return, a contradiction on the optimality of $\mathcal{T}^*_\mu$. Thus, every interval of $\mathcal{T}^K_{\mathsf{Strl}}$ contains at least one interval of $\mathcal{T}^*_\mu$. Now consider the maximal prefix $P_{\max}$ of I that does not overlap with any interval of $\mathcal{T}^*_\mu$. Since we know that I contains some interval of $\mathcal{T}^*_\mu$, we conclude that this maximal prefix must be adjacent to some interval of $\mathcal{T}^*_\mu$. By Lemma 2.5.15, this interval has strictly negative return, so removing it from I strictly increase the return of $\mathcal{T}^K_{\mathsf{Strl}}$, without increasing its MDD. This contradicts the optimality of $\mathcal{T}^K_{\mathsf{Strl}}$, thus, $P_{\max}$ must be empty. Similarily, the maximal suffix of I that is non-intersecting with $\mathcal{T}^*_\mu$ must be empty, concluding the proof of Proposition 2.5.17. ■

As a result of Proposition 2.5.17, we assume from now on that every interval of the sterling optimal strategy $\mathcal{T}^K_{\mathsf{Strl}}$ is prefixed and suffixed by (not necessarily distinct) intervals from a maximal return-optimal strategy that makes K_0 trades.

Lemma 2.5.18 *If $1 < K \leq K_0$ then $\mathcal{T}^K_{\mathsf{Strl}}$ can be chosen to make exactly K trades.*

Proof: If $K = K_0$, then $\mathcal{T}^*_\mu$ itself is K-Sterling-optimal. If $K < K_0$, we show that if the number of trades made is less than K, we can always add one more interval without decreasing the Sterling ratio of the strategy. First, note that $\mathcal{T}^K_{\mathsf{Strl}}$ cannot contain all the intervals of $\mathcal{T}^*_\mu$, as otherwise (by the pigeonhole principle) at least one interval $I = [t_l, t_r]$ of $\mathcal{T}^K_{\mathsf{Strl}}$ contains two consecutive intervals $I_1 = [t_{l_1}, t_{r_1}]$ and $I_1 = [t_{l_2}, t_{r_2}]$ of $\mathcal{T}^*_\mu$. The region between these two intervals has return less than $-f_{sp}$ (Lemma 2.5.15), so breaking up I into the two intervals $[t_l, t_{r_1}]$ and $[t_{l_2}, t_r]$ will strictly increase the return, without increasing the MDD, contradicting the optimality of $\mathcal{T}^K_{\mathsf{Strl}}$. If $\mathcal{T}^K_{\mathsf{Strl}}$ does not contain some interval of $\mathcal{T}^*_\mu$, then by adding this interval, we do not decrease the return or the MDD (Lemma 2.5.15), since the MDD is already $\geq f_{sp}$. ■

Lemmas 2.5.17 and 2.5.18 indicate how $\mathcal{T}^K_{\mathsf{Strl}}$ can be constructed: start with all the intervals of a maximal return-optimal strategy $\mathcal{T}^*_\mu$ and then merge some neighbouring intervals, keeping the merging sequence that gave the best strategy. The number of possible merging sequences is exponential, however, we will now show that an efficient greedy merging algorithm gives the correct result.

Given two consecutive non-adjacent intervals $I_1 = [t_{l_1}, t_{r_1}]$, $I_2 = [t_{l_2}, t_{r_2}]$, where I_1 preceeds I_2, define the *bridge* $B(I_1, I_2) = [t_{r_1}, t_{l_2}]$ to be interval connecting I_1 with I_2. If I_1 and I_2 are intervals in a maximal return optimal strategy, then by Lemma 2.5.15, the MDD of the bridge is $C_{r_1} - C_{l_2}$. Since C_{r_1} is a maximum over the interval $[t_{l_1}, t_{l_2}]$, and C_{l_2} is a minimum over the interval $[t_{r_1}, t_{r_2}]$, we have that the MDD of the union of these three intervals, $[t_{l_1}, t_{r_2}]$ is given by $\max\{MDD(I_1), C_{r_1} - C_{l_2}, MDD(I_2)\}$.

For every bridge $B(I_1, I_2)$, define the *closure* $\mathcal{C}l(B(I_1, I_2))$ to be the *smallest* interval $J = [t_l, t_r]$, in the return sequence, satisfying the following three properties.

$\mathcal{C}l_1$. $C_l \leq C_m \leq C_r$ for $l \leq m \leq r$, i.e., C_l is a minimum and C_r is a maximum in $[t_l, t_r]$.

$\mathcal{C}l_2$. $I_1, I_2 \subset J$, i.e., J contains both I_1 and I_2.

$\mathcal{C}l_3$. J is prefixed and suffixed by intervals from a maximal return-optimal strategy $\mathcal{T}^*_\mu$.

Note that a bridge may not have closure. For example, if the two last intervals I_{l-1}, I_l in $\mathcal{T}^*_\mu$ are such that such that the end point I_l is below the end point of I_{l-1}, then $B(I_{l-1}, I_l)$ doesn't have a closure. The next lemma shows that if the closure J for a bridge exists, then not only is it unique, but any other interval satisfying $\mathcal{C}l_1$ - $\mathcal{C}l_3$ contains J.

Lemma 2.5.19 *For two intervals I_1, I_2, if $\mathcal{C}l(B(I_1, I_2))$ exists, then it is unique. Moreover, for any other interval I satisfying $\mathcal{C}l_1$ - $\mathcal{C}l_3$, $\mathcal{C}l(B(I_1, I_2)) \subseteq I$.*

Proof: Let $J_1 = [t_{l_1}, t_{r_1}]$ and $J_2 = [t_{l_2}, t_{r_2}]$ satisfy $\mathcal{C}l_1$ - $\mathcal{C}l_3$. Without loss of generality, assume that $t_{l_1} \leq t_{l_2} < t_{r_1} \leq t_{r_2}$. By construction, $J_1 \cap J_2 = [t_{l_2}, t_{r_1}]$ satisfies $\mathcal{C}l_1$ - $\mathcal{C}l_3$. Now let $\mathcal{C}l(B(I_1, I_2))$ be the intersection of all intervals that satisfy $\mathcal{C}l_1$ - $\mathcal{C}l_3$, concluding the proof. ■

Suppose that bridge B and B' are bridges in $\mathcal{T}_\mu^*$ and that $\mathcal{C}l(B)$ contains B'. Then $\mathcal{C}l(B)$ satisfies $\mathcal{C}l_1$ - $\mathcal{C}l_3$ with respect to B' and hence $\mathcal{C}l(B)$ also contains $\mathcal{C}l(B')$.

Lemma 2.5.20 *Let B and B' be bridges in $\mathcal{T}_\mu^*$. If $B' \subset \mathcal{C}l(B)$, then $\mathcal{C}l(B') \subset \mathcal{C}l(B)$.*

Any interval in $\mathcal{T}_{\mathsf{Strl}}^K$ containing bridge B satisfies properties $\mathcal{C}l_1$ - $\mathcal{C}l_3$ (Lemma 2.5.16 & Proposition 2.5.17), immediately yielding the following proposition.

Proposition 2.5.21 *Let $I \in \mathcal{T}_{\mathsf{Strl}}^K$ and let B be a bridge in $\mathcal{T}_\mu^*$.*

i. If $B \subset I$, then $\mathcal{C}l(B) \subset I$.

ii. If B does not have a closure, then no K-Sterling-optimal strategy can contain B.

iii. A K-Sterling-optimal strategy with more than one trading interval and no bridges of $\mathcal{T}_\mu^*$ has $MDD = f_{sp}$. If it contains one or more bridges B_i of $\mathcal{T}_\mu^*$, then $MDD = \max_i MDD(\mathcal{C}l(B_i))$.

iv. The MDD of a K-Sterling-optimal strategy with more than one trading interval can be one of at most $T + 1$ possible values where T is the number of bridges between the intervals of $\mathcal{T}_\mu^*$.

Proof: (i) and (ii) are immediate. (iv) follows from (iii), thus we only need to prove (iii). Let $I \in \mathcal{T}_{\mathsf{Strl}}^K$ contain the consecutive bridges $B_1, \ldots, B_K$, and hence their closures. From (i), it is clear that $MDD(I) \geq \max_i MDD(\mathcal{C}l(B_i))$. It is also clear that $I = \cup_i^K \mathcal{C}l(B_i)$. We show, by strong induction on K, a more general statement than we need: suppose that $I = \cup_i^K \mathcal{C}l(B_i)$, then $MDD(I) \leq \max_i MDD(\mathcal{C}l(B_i))$. If $K = 1$ then $I = \mathcal{C}l(B_1)$ and the result is trivial; suppose it is true for up to $K - 1$ consecutive bridges, $K > 1$, and suppose that I is the union of K closures of consecutive bridges. Consider the first closure $\mathcal{C}l(B_1)$. Let $I = [t_l, t_r]$ and $\mathcal{C}l(B_1) = [t_l, t_{r'}]$, $t_{r'} \leq t_r$. By definition of $\mathcal{C}l(B_1)$, $C_{r'}$ is a maximum over $[t_l, t_{r'}]$. Thus, $MDD(I) = \max\{MDD(\mathcal{C}l(B_1)), MDD([t_{r'}, t_r])\}$. If $r = r'$, then $I = \mathcal{C}l(B_1)$ and we are done. If $r < r'$, then $t_{r'+1}$ is the begining of some bridge B_κ. Let $I' = \cup_{i=\kappa}^K \mathcal{C}l(B_i)$. Then, $[t_{r'}, t_r] \subseteq I'$ and so $MDD([t_{r'}, t_r]) \leq MDD(I')$. But I' is the union of at most $K - 1$ closures, so by the induction hypothesis, $MDD(I') \leq \max_{i \geq \kappa} MDD(\mathcal{C}l(B_i))$, concluding the proof. ■

We will distinguish between four types of bridges. Let $I_1 = [t_{l_1}, t_{r_1}]$, $I_2 = [t_{l_2}, t_{r_2}]$ be consecutive intervals in $\mathcal{T}_\mu^*$. The bridge $B = B(I_1, I_2)$ can be one of four types:

regular.	$C_{l_1} \leq C_{l_2}$ and $C_{r_1} \leq C_{r_2}$, i.e, $\mathcal{C}l(B) = [l_1, r_2]$.
right irregular.	$C_{l_1} \leq C_{l_2}$ and $C_{r_1} > C_{r_2}$, i.e, $\mathcal{C}l(B)$ contains the next bridge.
left irregular.	$C_{l_1} > C_{l_2}$ and $C_{r_1} \leq C_{r_2}$, i.e, $\mathcal{C}l(B)$ contains the previous bridge.
irregular.	$C_{r_1} > C_{r_2}$ and $C_{l_1} > C_{l_2}$, i.e., $\mathcal{C}l(B)$ contains the next and previous bridges.

We define the *weight* of the bridge $W(B(I_1, I_2))$ as follows:

$$W(B(I_1,I_2)) = \begin{cases} C_{r_1} - C_{l_2} & \text{if } B(I_1,I_2) \text{ is regular,} \\ C_{r_1} - C_{l_2} & \text{if } B(I_1,I_2) \text{ is left irregular and the previous bridge is right irregular} \\ C_{r_1} - C_{l_2} & \text{if } B(I_1,I_2) \text{ is left irregular and the previous bridge is irregular} \\ +\infty & \text{otherwise.} \end{cases}$$

The general idea behind our algorithm is to start with a maximal return-optimal strategy and greedily merge pairs of intervals or pair of bridges according to the bridge weight, keeping track of the best K intervals each time. When no more merging can occur, because we are down to K intervals or all the bridge weights are ∞, we return the best K intervals we have seen so far. More precisely, let $\mathcal{T}_\mu^* = \{I_1, \ldots, I_{K_0}\}$ be a maximal return-optimal trading strategy making K_0 trades. We denote this pool of trade intervals by P_0, the *base pool*. From pool P_i, we obtain pool P_{i+1} by a single merge according to the following rule. Let $B = B(I_1, I_2)$ be the bridge with smallest weight. If $B = \infty$, stop (pool P_{i+1} does not exist). Otherwise, there are two cases.

i. *Regular merge*: if B is regular, merge B with I_1 and I_2 to get a larger interval $I_{new} = [t_{l_1}, t_{r_2}]$. We now update the status (type and weight) of any neighboring bridges as follows:

   - Previous bridge changes from right-irregular to regular (update type and weight).
   - Previous bridge B' changes irregular to left-irregular (update type). If the bridge previous to B' is right-irregular or irregular then update weight.
   - Next bridge changes from irregular to right-irregular (update type).
   - Next bridge changes from left-irregular to to regular (update type and weight).

ii. *Irregular merge*: if B is left irregular, denoting the previous bridge B^*, merge the two bridges and the interval between them into one bigger bridge $B_{new} = B^* \cup I_1 \cup B$. The status of bridges other than B have not changed. The status and weight of B may need updating.

Intervals are formed only by regular merges, so it is easy to see that intervals resulting from this merging procedure begin at a minimum and end at a maximum. New bridges are formed by irregular merges, and the resulting bridge must begin at a maximum and end at a minimum. The only bridge weights that could change are those that had weights of ∞. In such an event the weight will drop, but not below the weight of the original bridge used in the merge that led to the update.

Lemma 2.5.22 *Let bridge B with weight w be involved in a merge, and suppose that the weight of bridge B' is updated in the process from ∞ to u. Then, $w < u$.*

Proof. There are two cases:

i. The bridge involved in the merge was regular, i.e., two consecutive intervals $I_1 = [t_{l_1}, t_{r_1}]$ and $I_2 = [t_{l_2}, t_{r_2}]$ are merged with their bridge $B_{12} = B(I_1, I_2)$ with $W(B_{12}) = w$. Let the preceeding interval be $I_0 = [t_{l_0}, t_{r_0}]$ and the following interval be $I_3 = [t_{l_3}, t_{r_3}]$, and let the preceeding and following bridges be B_{01} and B_{23} respectively. If B_{01} obtained finite weight, it cannot be left irregular, as it would remain left irregular after the merge, and hence its weight would still be ∞. Thus, we need only consider B_{01} right irregular or irregular (i.e., $C_{r_0} > C_{r_1}$).

Its weight becomes $u = C_{r_0} - C_{l_1} > C_{r_1} - C_{l_1}$. Since B_{12} is regular, $w = C_{r_1} - C_{l_2} < C_{r_1} - C_{l_1}$ and so $w < u$. If B_{23} obtained finite weight, then it could not be right regular or irregular as it could not become regular or left irregular after the merge. Thus, we need only consider B_{23} left irregular ($C_{l_2} > C_{l_3}$). Its weight becomes $u = C_{r_2} - C_{l_3} > C_{r_2} - C_{l_2}$. Since B_{12} is regular, $C_{r_1} \leq C_{r_2}$, and so $u > C_{r_1} - C_{l_2} = w$.

ii. The bridge involved in the merge was left-irregular, i.e., $B_{12} = [t_{r_1}, t_{l_2}]$ is left irregular, and $B_{01} = [t_{r_0}, t_{l_1}]$ is either right-irregular or irregular (in both cases, $C_{r_0} > C_{r_1}$). Let $w = C_{r_1} - C_{l_2}$ be the weight of B_{12}. The merged bridge is $B = B_{01} I_1 B_{12}$. If B has finite weight (i.e. it is either regular or left-irregular), then its new weight is $u = C_{r_0} - C_{l_2} > C_{r_1} - C_{l_2} = w$. If B is left-irregular or irregular, then it does not change the weight of any other bridge. If, on the other hand, B became right-irregular or irregular, then it could affect the weight of the following bridge B_{23}, if B_{23} was left-irregular ($C_{l_2} > C_{l_3}$). In this case, the weight of $B_{23} = [t_{r_2}, t_{l_3}]$ becomes $v = C_{r_2} - C_{l_3} > C_{r_2} - C_{l_2}$. But since B_{12} was left-irregular, $C_{r_2} \geq C_{r_1}$, and so $v > C_{r_1} - C_{l_2} = w$. ■

The next lemma shows that if all the bridge weights become ∞, any further merged pairs of intervals can never be part of a K-Sterling-optimal strategy.

Lemma 2.5.23 *If all the bridge weights in a pool of intervals are ∞, then any further merged pairs of intervals from this pool can never be part of a K-Sterling-optimal strategy.*

Proof: (Lemma 2.5.23). Let P_r be pool obtained from P_0 by some sequence of merging intervals with bridges of finite weight, and suppose that all the bridges in P_r have infinite weight. In particular, this means that none of the bridges in P_r are regular. Denote the bridges by $B_1, \ldots, B_m$, and consider bridge B_k. If B_k is right irregular or irregular, then all following bridges are either right irregular or irregular since all bridges have finite weight. If a trading interval contains B_k, it must contain B_{k+1} (since B_k is right irregular or irregular), and so by induction, it must contain all the following bridges (and their closures). But, the last bridge does not have a closure (as it is right irregular or irregular), a contradiction. If on the other hand, B_k is left irregular, then all preceeding bridges are left irregular as all bridges have infinite weight. If a trading interval contains B_k, it must contain B_{k-1} (since B_k is left irregular), and so by induction, it must contain all the preceeding bridges (and their closures). But, the first bridge does not have a closure (as it is left irregular), a contradiction. We conclude that B_k cannot be in any trading interval. ■

Each merge decreases the number of intervals and number of bridges by one. If we merge down to pool P_{K_0-K}, we are left with exactly K intervals. We will show that $\mathcal{T}^K_{\mathsf{Strl}}$ can be chosen to be the best K trades (with respect to total return) in one of these pools. Specifically, define $\mathcal{T}^K_j$ to be the K intervals in P_j with the highest total return. We say that a strategy is *coarser* than pool P_i if the strategy can be obtained by a sequence of merges of some (or all) of the intervals in P_i. Clearly, $\forall i$, P_{i+1} is coarser than P_i, as P_{i+1} is obtained from P_i after a single merge. Note that for a strategy to be coarser than P_i, it need not contain every trade in P_i, however if it contains part of any trade in P_i, then it contains the entire trade. Next, we show that after a merge, the MDD of the remaining intervals is equal to the weight of the bridge involved in the merging.

Lemma 2.5.24 *If pool P_i, $i \geq 1$, was obtained from P_{i-1} by a merge involving a bridge of weight w, then the MDD of any interval in P_i is at most w. If the merge created a new interval (i.e., the bridge was regular), then the MDD of the new interval is equal to w.*

Proof In pool P_0, since any bridge is adjacent to two intervals of $\mathcal{T}^*_\mu$, its weight is at least f_{sp} (Lemma 2.5.15). Consider sequence of pools $P_0, P_1, \ldots, P_r$, where bridge B_i with weight $W(B_i)$ was the minimum weight bridge involved in the merge that resulted in pool P_i from from P_{i-1}. By Lemma 2.5.22 bridges weights are non-decreasing, i.e., $W(B_i) \leq W(B_{i+1})$.

We now use induction on the index i. For $i = 1$, from Lemma 2.5.15, every interval in P_0 has MDD at most f_{sp}. If P_1 was obtained from P_0 by an irregular merge, then all intervals of P_1 are intervals of P_0, with MDD at most f_{sp}. Since $W(B_1) \geq f_{sp}$, the claim holds. If the merge was regular, then the MDD is $W(B_1) \geq f_{sp}$ and the MDD of all other intervals is at most f_{sp}. Thus, the claim holds for P_1.

Suppose the claim holds for all $j < i$ and consider pool P_i which was obtained from P_{i-1} using a merge involving B_i. By the induction hypethesis, the MDD of any interval from P_{i-1} is at most $W(B_{i-1}) \leq W(B_i)$. If P_i that was obtained by an irregular merge, every interval of P_i is an interval of P_{i-1} and thus has MDD at most $W(B_{i-1}) \leq W(B_i)$. Suppose that P_i was obtained by a regular merge – all intervals except the merged interval are intervals of P_{i-1}. Consider the MDD of the new interval, which is obtained by the regular merge $I_1 \cup B_i \cup I_2$. Since new intervals are created only through regular merges, it is easy to see by induction that property $\mathcal{C}l_1$ holds for all the intervals in P_{i-1}, in particular it holds for I_1 and I_2. Since B_i was regular, the MDD of the new interval is $\max(MDD(I_1), W(B_i), MDD(I_2))$. By the induction hypothesis, $MDD(I_1) \leq W(B_{i-1})$ and $MDD(I_2) \leq W(B_{i-1})$, thus, $\max(MDD(I_1), W(B_i), MDD(I_2)) = W(B_i)$. ■

First, we show that if a K-Sterling-optimal strategy makes K trades, all of which are contained in intervals of one of the pools P_i, then a K-Sterling-optimal strategy exists which is composed of the K intervals with highest return in some pool P_j with $j \leq i$.

Lemma 2.5.25 *If K subintervals of the intervals of pool P_i are a K-Sterling-optimal strategy, then for some $j \leq i$, the K intervals with highest return of pool P_j are a K-Sterling-optimal strategy.*

Proof: If $P_i = P_0$, then the claim is trivial. Suppose that $i > 0$, and let $\mathcal{T} = \{I_1, \ldots, I_K\}$ be the K-Sterling-optimal strategy whose trades are all subintervals of intervals in P_i. Consider the set $\mathcal{B}$ of all bridges in $\mathcal{T}^*_\mu$ that are contained in $\mathcal{T}$, $\mathcal{B} = \{B_i\}_{i=1}^r$. We can assume that $\mathcal{B}$ is not empty because if it were, then $\mathcal{T}$ is composed of intervals in $\mathcal{T}^*_\mu$, in which case the top K intervals (with respect to return) in $\mathcal{T}^*_\mu$ are clearly optimal. Since P_i contains all the intervals in $\mathcal{T}$, P_i contains all the bridges in $\mathcal{B}$. Thus, there must exist $j \leq i$ such that P_j contains all the bridges in $\mathcal{B}$ and no pool P_k, with $k < j$ has this property, i.e., P_j was obtained from the previous pool by a regular merge involving a bridge B^* which must contain some bridge $B_l \in \mathcal{B}$. Let I be the interval in $\mathcal{T}$ that contains B_l. Then, I must contain the whole bridge B^*, since if B^* is the result of irregular merges, one of which involved bridge B_l, then $B^* \subset \mathcal{C}l(B_l)$, and $\mathcal{C}l(B_l) \subseteq I$ (Proposition 2.5.21). Since $B \subset I$, $MDD(\mathcal{T}) \geq MDD(I) \geq W(B^*)$. By Lemma 2.5.24, since B^* was the last bridge involved in a merge, the MDD of every interval in P_j is at most $W(B^*)$. Since every interval of $\mathcal{T}$ is a subinterval of some interval in P_j, we conclude that $\mathcal{T}$ is contained in at most K intervals of P_j. Let $\mathcal{T}_j^K$ be the top K intervals in P_j. Then, the return of $\mathcal{T}$ is at most the return of $\mathcal{T}_j^K$. Further, $MDD(\mathcal{T}_j^K) \leq W(B^*) \leq MDD(\mathcal{T})$, and so $\mathsf{Strl}(\mathcal{T}_j^K) \geq \mathsf{Strl}(\mathcal{T})$, concluding the proof. ■

We are now ready to prove the main result, which will lead to the greedy algorithm for constructing a K-Sterling optimal strategy.

Theorem 2.5.26 *Let j^* be such that $\mathsf{Strl}(\mathcal{T}_{j^*}^K) \geq \mathsf{Strl}(\mathcal{T}_j^K),\ \forall j$. Then $\mathcal{T}_{j^*}^K$ is K-Sterling optimal.*

Proof. Let S_0^K be a K-Sterling-optimal strategy that makes K trades – by Lemma 2.5.18, such a strategy must exist. If S_0^K has the same Sterling ratio as the trading strategy composed of the K most profitable trades in P_0, then we are done. If not, then we know from Proposition 2.5.17 that S_0^K is coarser than P_0. We prove the following statement for all $k \geq 1$

> $Q(k)$: Suppose there exists a K-Sterling-optimal strategy S_{k-1}^K that makes K trades and is coarser than P_{k-1}. Then either S_{k-1}^K is composed of K intervals of P_k, or there exists a K-Sterling-optimal strategy S_k^K that makes K trades and is coarser than P_k.

We know that $Q(1)$ is true. Suppose that $Q(k)$ is true for all $k \geq 1$, we then prove the proposition as follows. By an easy induction, we have that if none of the S_{j-1}^K are composed of K intervals in P_j for all $j \leq m$, then there is a K-Sterling-optimal strategy S_m^K making exactly K trades that is coarser than P_m. Suppose that we can construct a total of $\kappa+1$ pools, P_i for $0 \leq i \leq \kappa \leq K_0 - K$. If $\kappa < K_0 - K$ then all the bridge weights in P_κ are infinite. If $\kappa = K_0 - K$, then any further merging leads to fewer than K intervals. In both cases, there cannot exist a K-Sterling-optimal strategy that is coarser than P_κ. Therefore, for some $j^* \leq \kappa$, the K-Sterling-optimal strategy $S_{j^*-1}^K$ is composed of K intervals of P_{j^*}. By Lemma 2.5.25, there is a K-Sterling-optimal strategy T_{Strl}^K that is composed of the top K intervals of some pool P_l, where $l \leq j^*$.

What remains is to show that $Q(k)$ is true for all $k \geq 1$. Suppose that S_{k-1}^K is coarser than P_{k-1} and is not composed of K intervals in P_k. We show that there exists S_k^K that is coarser than P_k. Since S_{k-1}^K is coarser than P_{k-1}, it contains at least one bridge B in P_{k-1} with finite weight (because if it contains an infinite weight bridge, then it either contains the preceeding or following bridge; this argument continues analogously to the proof of Lemma 2.5.23 until we include a bridge of finite weight). Let I be the interval of S_{k-1}^K that contains B, and let I_l and I_r be intervals in P_{k-1} (which are subintervals of I) connected by B. Let B^* be the bridge in P_{k-1} with minimum weight that was involved in the merge to get P_k from P_{k-1}, and let I_l^* and I_r^* be the intervals in P_{k-1} connected by B^*. If $B^* = B$ then S_{k-1}^K is also coarser than P_k and we are done, so suppose $B^* \neq B$. There are two possibilities:

(i) *B^* is a regular bridge.* If S_{k-1}^K does not contain I_l^* or I_r^*, then $S_{k-1}^K \cap (I_l^* \cup B^* \cup I_r^*) = \emptyset$ and thus S_{k-1}^K itself is coarser than P_k, and can be chosen as S_k^K. Suppose that S_{k-1}^K contains I_l^* and not I_r^* (similar argument if it contains I_r^* and not I_l^*). Thus some interval $I' \in S_{k-1}^K$ has as a suffix I_l^*. Suppose we construct S_k^K by replacing interval I' by interval $I' \cup B^* \cup I_r^*$. S_k^K is then coarser than P_k. Since B^* is regular, the return of $I' \cup B^* \cup I_r^*$ is at least as big as the return of I'. I_r^* is either an interval of P_0 or was obtained by merging some intervals of P_0 through bridges with weight at most $W(B^*)$ (Lemma 2.5.24), and so $MDD(I_r^*) \leq W(B^*)$. Since the maximum cumulative return for I' is attained at its right endpoint (Lemma 2.5.16) and the left endpoint of I_r^* is a minimum in I_r^*, we have that $MDD(I' \cup B^* \cup I_r^*) = \max\{MDD(I'), W(B^*), MDD(I_r^*)\} = \max\{MDD(I'), W(B^*)\}$. Since $W(B^*) \leq W(B)$, we conclude that $MDD(S_k^K) \leq MDD(S_{k-1}^K)$, and thus $\mathsf{Strl}(S_k^K) \geq \mathsf{Strl}(S_{k-1}^K)$, which means that S_k^K is also K-Sterling-Optimal. Finally, suppose that S_{k-1}^K contains both I_l^* and I_r^*, and consider the strategy S_k^K obtained from S_{k-1}^K by removing bridge B and adding bridge B^*. $\mu(S_k^K) = \mu(S_{k-1}^K) + W(B) - W(B^*) \geq \mu(S_{k-1}^K)$. Since $W(B) \geq W(B^*)$, the MDD cannot have increased, and so S_k^K is K-Sterling-Optimal and coarser than P_k.

(ii) B^* *is an irregular bridge.* Since $B^* = B(I_l^*, I_r^*)$ has finite weight, we can conclude that B^* is left-irregular and the previous bridge $B_- = B(I_{l-1}^*, I_l^*)$ is right-irregular or irregular. Since S_{k-1}^K does not contain B^*, by Lemma 2.5.16, there are two possibilities: S_{k-1}^K does not contain I_l^*, in which case it also does not contain bridge B_- and so B_- and B^* can be merged into one bridge without influencing S_{k-1}^K, i.e., S_{k-1}^K is also more coarse than P_k; or, S_{k-1}^K contains I_l^* as one of its intervals. In this case, since B^* is left-irregular, $\mu(I_l^*) < W(B^*) \leq W(B)$, and so by dropping I_l^* from S_{k-1}^K and breaking I into two subintervals by removing B from I results in a profit increase of $W(B) - \mu(I_l^*) > 0$. Further, the MDD cannot increase, so the new strategy makes K trades and has strictly greater Sterling ratio than S_{k-1}^K, which contradicts optimality of S_{k-1}^K. Thus, S_{k-1}^K cannot contain I_l^* as one of its intervals.

Thus $Q(k)$ holds for all $k \geq 1$, concluding the proof. ■

We are now ready to give the $O(n \log n)$ algorithm that establishes Theorem 2.3.3. First, we can compute the optimal strategy that makes only one trade in $O(n \log n)$ (Section 2.5.1), and compare this with the trivial strategy that makes no trades. It remains to compute the K-Sterling-optimal strategy and pick the best. We show how to do this in $O(n \log n)$ time.

First we obtain $\mathcal{T}_\mu^*$ in linear time. Suppose $\mathcal{T}_\mu^*$ makes $K_0 > K$ trades (as otherwise $\mathcal{T}_\mu^*$ is our solution). By Theorem 2.5.26, we only need to construct the pools $P_0, P_1, \ldots$, maintaining the pool with the optimal Sterling ratio for its top K trades, as explained in the following algorithm.

1: Set $i = 0$; Sort (in decreasing order) the intervals from P_0 according to profit; Sort all the bridges with finite weight in increasing order. Let B_i be the minimum weight bridge in P_i; Let strategy S_i consist of the top K intervals, and let $\mathsf{Strl}_{opt} = \mathsf{Strl}(S_i)$;
2: **while** P_i contains at least K intervals and at least one finite weight bridge **do**
3: **if** $B_i = B(I_l, I_r)$ is regular **then**
4: Regular merge to obtain P_{i+1}: remove I_l, I_r, B_i from the interval and bridge orderings, and add back $I = I_l \cup B_i \cup I_r$ into the interval ordering; compute $\mu(I)$ and $MDD(I)$;
5: Update neighboring bridge weigths and re-insert them back into the bridge ordering.
6: **else if** $B_i = B(I_l, I_r)$ is left-regular **then**
7: Irregular merge to obtain P_{i+1}: Let B_- be the bridge left of B_i; remove I_l, B_-, B_i from the interval and bridge orderings. Create the new bridge $B = B_- \cup I_l \cup B_i$, compute $W(B)$ and insert B into the bridge ordering (note that $W(B)$ may be ∞).
8: **end if**
9: $i \leftarrow i + 1$; update Strl_i; if $\mathsf{Strl}_{opt} < \mathsf{Strl}_i$, then $\mathsf{Strl}_{opt} \leftarrow \mathsf{Strl}_i$.
10: **end while**

The correctness of the algorithm follows from Theorem 2.5.26. We now analyse the run time of an efficient implementation of the algorithm. P_0 contains at most n intervals and bridges. Each execution of the while loop reduces loop number of bridges and intervals by 1 each, so the while loop is executed at most $n/2$ times. Merging two intervals is a constant time operation. The profit of a new interval is the profit of the merged intervals minus the weight of the merging bridge (also computable in constant time). The MDD of a new interval is the maximum of the MDD of the merged intervals and the weight of the merging bridge (also computable in constant time). The weight of a new bridge takes constant time to compute, and updating the weights of the neighbour bridges is a constant time operation provided that pointers are maintained to them. These pointers can be updated in constant time as well. Thus the run time within the while loop is dominated by

inserting into the bridge or interval orderings. At most a constant number of such such inserts into a *sorted list* need to be done, and each is an $O(\log n)$ operation [17]. To efficiently implement step 9, we maintain two sorted lists of the top K intervals in the algorithm, sorted according to return and MDD. These can be maintained in $O(\log K)$ operations. The first allows us to update the total return of the top K intervals in constant time, and the second allows us to update the MDD of the top K intervals (by taking the interval with largest MDD) in constant time. Thus the total running time of the while loop is $O(n \log n + n \log K) = O(n \log n)$ The preprocessing (step 1) is $O(n)$, and so does not contribute to the asymptotic run time.

2.6 Collecting Training Dataset: Sharpe Optimal Trading Strategies

Another popular measure of the portfolio's risk-adjusted return is the Sharp Ratio. For trading strategy $\mathcal{T}$, we consider two versions of the Sharpe ratio, Shrp_1 and Shrp_2.

$$\mathsf{Shrp}_1(\mathcal{T}) = \frac{\mu(\mathcal{T})}{\sigma(\mathcal{T})}, \qquad \mathsf{Shrp}_2(\mathcal{T}) = \frac{\mu(\mathcal{T})}{\sigma^2(\mathcal{T})}. \tag{2.9}$$

Note that Shrp_2 is more conservative in that it penalizes large variances more heavily. We introduce a simplified Sharpe ratio (SSR) S that will be instrumental to finding the optimal strategies,

$$\mathsf{S} = \frac{\mu}{s^2}.$$

It is easy to check that maximizing Shrp_1 is equivalent to maximizing $\frac{\mu^2}{s^2}$, and that Shrp_2 is given by $\frac{\bar{r}}{\frac{1}{n}s^2 - \bar{r}^2}$, where $\bar{r}$ is the mean return. We will relate the maximization of Shrp_1 and Shrp_2 to the maximization of S.

Let $\mathcal{T}$ be a trading strategy that makes K trades, with trading intervals $I_1, \ldots, I_K$. Each trade contributes a transactions cost of $-f_{sp}$ to the return sequence. In general, a trade contributes $\hat{f}_S^2 + \hat{f}_B^2$ to s^2. However, we will assume that $f_{sp} \ll 1$ and so we will ignore the contribution of the transactions cost to s^2. Alternatively, we can justify this by assuming that the transactions cost is spread finely over many time intervals. The sum over these small time intervals is finite, equal to $-f_{sp}$, however, the sum of squares over these small time intervals can be made arbitrarily small. Define the total return and sum of squared returns for each trading interval,

$$\mu_i = \mu(I_i) = \sum_{j \in I_i} r[j], \qquad s_i^2 = s^2(I_i) = \sum_{j \in I_i} r[j]^2.$$

We define A_i as the contribution of trade i to the mean return, and B_i as the contribution of trade i to the mean squared return (ignoring the effect of the transactions cost), i.e., $A_i = \frac{1}{n}(\mu_i - f_{sp})$ and $B_i = \frac{1}{n}s_i^2$. We define $A(\mathcal{T}) = \sum_{k=1}^K A_i$ (note that $\bar{r} = A(\mathcal{T})$) and $B(\mathcal{T}) = \sum_{k=1}^K B_i$ (note that $\frac{1}{n}s^2 = B(\mathcal{T})$).

2.6.1 Maximizing the Simplified Sharpe Ratio S

We will need the following technical lemma, which can be proved by an easy induction.

Lemma 2.6.1 *Let $\mathcal{F} = \{\frac{a_1}{b_1}, \frac{a_2}{b_2}, \ldots, \frac{a_k}{b_k}\}$ be any set of fractions satisfying $b_i > 0$ and $\frac{c}{d} \le \frac{a_i}{b_i} \le \frac{a}{b}$, for all i, where $b, d > 0$. Then, $\frac{c}{d} \le \frac{a_1+a_2+\ldots+a_k}{b_1+b_2+\ldots+b_k} \le \frac{a}{b}$. The upper (resp. lower) bound is strict if at least one of the fractions in $\mathcal{F}$ is strictly upper (resp. lower) bounded by $\frac{a}{b}$ $\left(\text{resp } \frac{c}{d}\right)$.*

Let $\mathcal{T}^*$ be an SSR-optimal strategy making $K > 1$ trades with trading intervals $I_1, \ldots, I_K$.

$$\mathsf{S}(\mathcal{T}^*) = \frac{\sum_{i=1}^K A_i}{\sum_{i=1}^K B_i} = \frac{A(\mathcal{T}^*)}{B(\mathcal{T}^*)},$$

Lemma 2.6.2 *$\frac{A_i}{B_i}$ is a constant for every interval i, i.e., every trade is equivalent.*

Proof: Suppose that $\min_i \frac{A_i}{B_i} < \frac{A_j}{B_j}$ for some j (strict inequality), and without loss of generality, assume that the minimum is attained for interval I_1. By Lemma 2.6.1, if we remove I_1, we get that

$$\mathsf{S}(I_1 \cup \cdots \cup I_K) = \frac{\sum_{i=1}^K A_i}{\sum_{i=1}^K B_i} < \frac{\sum_{i=2}^K A_i}{\sum_{i=2}^K B_i} = \mathsf{S}(I_2 \cup \cdots \cup I_K),$$

which contradicts the optimality of $\mathcal{T}^*$ implying that $\min_i \frac{A_i}{B_i} = \frac{A_j}{B_j}$ for all j. ■

Corollary 2.6.3 *An SSR-optimal trading strategy making one trade exists.*

Proposition 2.6.4 *An SSR-optimal strategy making one trade can be found in $O(n \log n)$ time.*

Proof: By Corollary 2.6.3, we are guaranteed the existence of such a strategy. It suffices to find the single interval I maximizing $\sum_{i \in I} r[i] / \sum_{i \in I} r[i]^2$. Consider all intervals starting at position i and define $c_k = \sum_{j=i}^k r[j]$ and $d_k = \sum_{j=i}^k r[j]^2$. We wish to find k to maximize c_k/d_k. If we have done this for position i, we now consider position $i-1$. We show that the algorithm in Section 2.5.2 can be used. Trade intervals starting at $i-1$ correspond to shifting all the c_k by $r[i-1]$, and all the d_k by $r[i-1]^2$. Both these operations simply correspond to shifting the origin point $\mathbf{p}$ to $\mathbf{p}' = \mathbf{p} - (r[i-1], r[i-1]^2)$. We then add a new leftmost point at $\mathbf{p}$. Since each update takes $O(\log n)$, and the optimal interval for the new points can be found in $O(\log n)$, the entire algorithm runs in $O(n \log n)$. ■

2.6.2 Maximizing Shrp_2

Ignoring the ${f_{sp}}^2$ term in the denominator changes the denominator slightly, so we introduce the slightly different quantity $\overline{\mathsf{Shrp}}_2$. Specifically,

$$\mathsf{Shrp}_2(\mathcal{T}) = \frac{A(\mathcal{T})}{\frac{d}{n}{f_{sp}}^2 + B(\mathcal{T}) - A^2(\mathcal{T})}, \qquad \overline{\mathsf{Shrp}}_2(\mathcal{T}) = \frac{A(\mathcal{T})}{B(\mathcal{T}) - A^2(\mathcal{T})},$$

where d is the number of trades in $\mathcal{T}$. By the Cauchy-Schwarz inequality, for any trading strategy, $\sum r[i]^2 \geq \frac{1}{n}(\sum_i r[i])^2$. Since we are only interested in strategies for which $A(\mathcal{T}) \geq 0$, we have

Lemma 2.6.5 *For strategy $\mathcal{T}$, if $A(\mathcal{T}) > 0$ then $B(\mathcal{T}) - A^2(\mathcal{T}) > 0$*

We will show that maximizing $\overline{\mathsf{Shrp}}_2$ is closely related to a constrained optimization of the SSR, and that maximizing $\overline{\mathsf{Shrp}}_2$ is not too far from maximizing Shrp_2.

Let $\mathcal{T}_\mu^*$ be return optimal, with return $\mu(\mathcal{T}_\mu^*) = \mu^*$. For any $0 \leq \alpha \leq \mu^*$, we define the constrained SSR-optimal strategy $\mathcal{T}_\alpha$ as the strategy with maximum SSR among all strategies with return at least α, i.e., $A(\mathcal{T}_\alpha) \geq \alpha$ and for all strategies $\mathcal{T}$ with $A(\mathcal{T} \geq \alpha)$, $\mathsf{S}(\mathcal{T}_\alpha) \geq \mathsf{S}(\mathcal{T})$. Note that while an SSR-optimal strategy can be chosen with one trading interval, a constrained SSR-optimal strategy may require more than one trading interval. We show that for some appropriate threshold α, the constrained SSR-optimal strategy is a $\overline{\mathsf{Shrp}}_2$-optimal strategy.

Proposition 2.6.6 *$\exists 0 \leq \alpha \leq \mu^*$ such that the constrained SSR-optimal strategy $\mathcal{T}_\alpha$ is $\overline{\mathsf{Shrp}}_2$-optimal.*

Proof: Let $\mathcal{T}$ be any $\overline{\mathsf{Shrp}}_2$-optimal strategy, and let $\alpha^* = A(\mathcal{T})$. Let $\mathcal{T}_{\alpha^*}$ be any constrained SSR-optimal strategy. Then $A(\mathcal{T}_{\alpha^*}) \geq A(\mathcal{T})$ and since $\mathsf{S}(\mathcal{T}_{\alpha^*}) \geq \mathsf{S}(\mathcal{T})$, we have that

$$0 \leq A(\mathcal{T}_{\alpha^*})B(\mathcal{T}) - A(\mathcal{T})B(\mathcal{T}_{\alpha^*}).$$

Suppose that $\overline{\mathsf{Shrp}}_2(\mathcal{T}_{\alpha^*}) < \overline{\mathsf{Shrp}}_2(\mathcal{T})$, then

$$0 \leq A(\mathcal{T}_{\alpha^*})B(\mathcal{T}) - A(\mathcal{T})B(\mathcal{T}_{\alpha^*}) < A(\mathcal{T}_{\alpha^*})A(\mathcal{T}) \cdot (A(\mathcal{T}) - A(\mathcal{T}_{\alpha^*})).$$

Both $A(\mathcal{T}_{\alpha^*})$ and $A(\mathcal{T})$ are > 0, otherwise both strategies are inferior to $\mathcal{T}^*_\mu$; thus $A(\mathcal{T}) > A(\mathcal{T}_{\alpha^*})$, which is a contradiction. Therefore $\overline{\mathsf{Shrp}}_2(\mathcal{T}_{\alpha^*}) \geq \overline{\mathsf{Shrp}}_2(\mathcal{T})$ and so $\mathcal{T}_{\alpha^*}$ is $\overline{\mathsf{Shrp}}_2$-optimal. ■

We will need the following property of any SSR-optimal interval.

Proposition 2.6.7 *Let J be a subinterval of an SSR-optimal interval I. Then, $\mu(J) \geq -f_{sp}$. Further, if J is a prefix or suffix of I, then $\mu(J) > 0$.*

Proof: If J is a prefix or suffix of I and $\mu(J) \leq 0$, then deleting J from I gives at least as much return, with smaller sum of squared returns, contradicting the SSR-optimality of I. Suppose that $I = L \cup J \cup R$ where L and R are nonempty subintervals of I. If $\frac{\mu(J)+\mu(R)}{s^2(J)+s^2(R)} < \frac{-f_{sp}+\mu(L)}{s^2(L)}$, then by Lemma 2.6.1,

$$\mathsf{S}(I) = \frac{-f_{sp} + \mu(L) + \mu(J) + \mu(R)}{s^2(L) + s^2(J) + s^2(R)} < \frac{-f_{sp} + \mu(L)}{s^2(L)} = \mathsf{S}(L) \qquad (*)$$

This contradicts the optimality of I, so we have

$$\frac{\mu(J) + \mu(R)}{s^2(J) + s^2(R)} \geq \frac{-f_{sp} + \mu(L)}{s^2(L)}.$$

Now, suppose that $\mu(J) < -f_{sp}$. Using (*) and Lemma 2.6.1, we find that

$$\mathsf{S}(I) \leq \frac{\mu(J) + \mu(R)}{s^2(J) + s^2(R)} < \frac{-f_{sp} + \mu(R)}{s^2(J) + s^2(R)} < \frac{-f_{sp} + \mu(R)}{s^2(R)} = \mathsf{S}(R),$$

because $s^2(J) > 0$. This contradicts the SSR-optimality of I, so $\mu(J) \geq -f_{sp}$. ■

We show that adding an SSR-optimal interval to any trading strategy can only improve the strategy.

Proposition 2.6.8 *Let I_0 be any SSR-optimal interval, and let $\mathcal{T}$ be any trading strategy. Let $\mathcal{T}' = I_0 \cup \mathcal{T}$. Then $A(\mathcal{T}') \geq A(\mathcal{T})$ and $\mathsf{S}(\mathcal{T}') \geq \mathsf{S}(\mathcal{T})$*

Proof. If I_0 is contained in some interval of $\mathcal{T}$, then there is nothing to prove, so suppose that $\mathcal{T} \cup I_0 \neq \mathcal{T}$, and that $\mathcal{T}$ contains $d \geq 1$ trades $I_1, \ldots, I_d$. Note that $\mathsf{S}(T) = A(T)/B(T)$. If I_0 and $\mathcal{T}$ do not intersect, then $\mathcal{T}' = I_0 \cup \mathcal{T} = \{I_0, I_1, \ldots, I_d\}$. $A(\mathcal{T}') = A(\mathcal{T}) + A(I_0) \geq A(\mathcal{T})$, because $A(I_0) \geq 0$. Since I_0 is SSR-optimal, $\mathsf{S}(I_0) = \frac{A(I_0)}{B(I_0)} \geq \frac{A(\mathcal{T})}{B(\mathcal{T})} = \mathsf{S}(\mathcal{T})$, so by lemma 2.6.1,

$$\mathsf{S}(\mathcal{T}') = \frac{A(\mathcal{T}')}{B(\mathcal{T}')} = \frac{A(\mathcal{T}) + A(I_0)}{B(\mathcal{T}) + B(I_0)} \geq \frac{A(\mathcal{T})}{B(\mathcal{T})} = \mathsf{S}(\mathcal{T}).$$

Suppose that $\mathcal{T} \cap I_0 \neq \emptyset$. We can decompose $\mathcal{T}$ into four parts (each part could be empty): $\mathcal{T} = S_1 \cup S_2 \cup I_l \cup I_r$, where S_1 contains intervals that do not intersect with I_0, S_2 contains intervals that are contained in I_0, I_l is not contained in I_0 but overlaps I_0 on the left, and I_r is not contained in I_0 but overlaps I_0 on the right. $\mathcal{T}' = I_0 \cup \mathcal{T} = S_1 \cup I_l \cup I_0 \cup I_r$, i.e., adding I_0 combines all the trades in $\{S_2, I_l, I_r\}$ into one trade. Since the internal regions of I_0 have return at least $-f_{sp}$ and any prefix and suffix of I_0 has positive return (Proposition 2.6.7), we see that merging any two consecutive trades overlapping I_0 decreases the number of trades by one, hence increases the return by f_{sp} and the added interval loses at most f_{sp}, hence this merge can only increase $A(\mathcal{T})$. If either I_l or I_r are empty, then we are addionally adding a prefix or suffix of I_0 without changing the number of trades, which also increases $A(\mathcal{T})$, thus we see that $A(\mathcal{T}') \geq A(\mathcal{T})$.

Let's introduce the following definitions,

$$\begin{aligned}
A_1 &= A(S_1) + \frac{1}{n}(\mu(I_l \cap \overline{I_0}) + \mu(I_r \cap \overline{I_0})) \\
A_2 &= A(S_2) + \frac{1}{n}(\mu(I_l \cap I_0) - f_{sp} + \mu(I_r \cap I_0) - f_{sp}) \\
B_1 &= B(S_1) + \frac{1}{n}(s^2(I_l \cap \overline{I_0}) + s^2(I_r \cap \overline{I_0})) \\
B_2 &= B(S_2) + \frac{1}{n}(s^2(I_l \cap I_0) + s^2(I_r \cap I_0)),
\end{aligned}$$

where $\overline{I_0}$ is the complement of I_0. Letting $A_0 = A(I_0)$ and $B_0 = B(I_0)$, we then have

$$\mathsf{S}(\mathcal{T}) = \frac{A_1 + A_2}{B_1 + B_2}, \text{ and } \mathsf{S}(\mathcal{T}') = \frac{A_1 + A_0}{B_1 + B_0}.$$

Note that $\mathsf{S}(S_2 \cup (I_l \cap \overline{I_0}) \cup (I_r \cap \overline{I_0})) = \frac{A_2}{B_2}$, so by the optimality of I_0, $\frac{A_2}{B_2} \leq \frac{A_0}{B_0}$. We show that

$$\frac{\frac{1}{n}\mu(I_l \cap \overline{I_0})}{\frac{1}{n}s^2(I_l \cap \overline{I_0})} \leq \frac{A_0}{B_0}, \text{ and } \frac{\frac{1}{n}\mu(I_r \cap \overline{I_0})}{\frac{1}{n}s^2(I_r \cap \overline{I_0})} \leq \frac{A_0}{B_0}. \qquad (**)$$

If not, then suppose (for example) that $\frac{\frac{1}{n}\mu(I_l \cap \overline{I_0})}{\frac{1}{n}s^2(I_l \cap \overline{I_0})} > \frac{A_0}{B_0}$. Then,

$$\mathsf{S}(I_0 \cup I_l) = \frac{A_0 + \frac{1}{n}\mu(I_l \cap \overline{I_0})}{B_0 + \frac{1}{n}s^2(I_l \cap \overline{I_0})} > \frac{A_0}{B_0} = \mathsf{S}(I_0)$$

contradicting the SSR-optimality of I_0. Again, by the optimality of I_0, $\mathsf{S}(S_1) = \frac{A(S_1)}{B(S_2)} \leq \frac{A_0}{B_0} = \mathsf{S}(I_0)$. it also has to be sharper than strategy S_1. Thus, using (**) and Lemma 2.6.1 we have that $\frac{A_1}{B_1} \leq \frac{A_0}{B_0}$. Because A_2 is obtained from the returns of a collection of subintervals of I_0, it follows from Proposition 2.6.7 that $A_2 \leq A_0$. Now suppose that $\mathsf{S}(\mathcal{T}) > \mathsf{S}(\mathcal{T}')$, i.e.,

$$(A_1 + A_2)(B_1 + B_0) - (A_1 + A_0)(B_1 + B_2) > 0.$$

Since $A_2 \leq A_0$, it follows that $B_2 \leq B_0$. Rearranging terms in the equation above, we have that

$$\begin{aligned}
\frac{A_2B_1 - A_1B_2}{B_0(B_1 + B_2)} &> \frac{A_0}{B_0} - \frac{A_1 + A_2}{B_1 + B_2}, \\
&\geq \frac{A_2}{B_2} - \frac{A_1 + A_2}{B_1 + B_2} = \frac{A_2B_1 - A_1B_2}{B_2(B_1 + B_2)}.
\end{aligned}$$

Since $\mathsf{S}(I_0) \geq \mathsf{S}(\mathcal{T})$, the first inequality shows that $A_2B_1 - A_1B_2 > 0$. The second inequality then implies that $B_2 > B_0$, a contradiction. ∎

We can now give the intuition behind our algorithm. The starting point is Proposition 2.6.6, which says that it suffices to look for constrained SSR-optimal strategies. So the natural first choice is an unconstrained SSR-optimal interval $\mathcal{T}_0$. Either this will be $\overline{\mathsf{Shrp}}_2$ optimal or not. If not, it is because it has too small a return. So our next step is to add to this interval an new interval (possibly adjacent) with the property that the interval increases the return with *smallest* possible decrease in SSR, resulting in strategy $\mathcal{T}_1$. We repeat this process, constructing a sequence of trading strategies $\mathcal{T}_0, \mathcal{T}_1, \ldots$ with the property that $A(\mathcal{T}_i) > A(\mathcal{T}_{i-1})$, and among all other strategies $\mathcal{T}$ such that $A(\mathcal{T}) > A(\mathcal{T}_{i-1})$, $\mathsf{S}(\mathcal{T}_i) \geq \mathsf{S}(\mathcal{T})$. We then pick the strategy $\mathcal{T}_{i^*}$ with maximum $\overline{\mathsf{Shrp}}_2$ ratio among these strategies, which will be globally sharpe optimal.

Suppose that we have a current strategy, $\mathcal{T}_i$. We need to determine the next piece to add to this so that we increase the return, with smallest possible decrease in SSR. Let $\mathcal{T}_i$ be composed of the intervals $I_0, I_1, \ldots, I_d$. We replace each of these intervals by a special symbol, \$, to signify that these regions are already included in the strategy. We thus obtain a *generalized returns sequence*, one in which some intervals are replaced by the \$ symbol. A *generalized trading strategy* on the generalized return sequence must be composed of trades that do not contain the \$ symbol. However trades may be adjacent to the \$ symbol. A trade interval I in a generalized trading strategy can be *isolated* (not adjacent to any \$ symbol), *extending* (adjacent to one \$ symbol), or *bridging* (adjacent to two \$ symbols). In order to correctly account for the transactions cost, we need to change how we compute $A(I)$, so we introduce the new function $\mathcal{A}(I)$:

$$\mathcal{A}(I) = \begin{cases} A(I) & \text{I is isolated} \\ A(I) + \frac{f_{sp}}{n} & \text{I is extending} \\ A(I) + \frac{2f_{sp}}{n} & \text{I is bridging} \end{cases}$$

The *generalized simplified Sharp ratio (GSSR)* for generalized strategy $\mathcal{T} = \{I_1, \ldots, I_d\}$ is

$$\bar{\mathsf{S}}(\mathcal{T}) = \frac{\sum_{i=1\ldots d} \mathcal{A}(I_i)}{\sum_{i=1\ldots d} B(I_i)}$$

Similar to the notion of a maximal return optimal strategy, we introduce the notion of a maximal SSR-optimal (or GSSR-optimal) interval as one which cannot be extended in either direction without decreasing the SSR (or GSSR).

We now define generalized return sequences $\{R_0, R_1, \ldots\}$ as follows. R_0 is just the original returns sequence. Let I_i be a maximal GSSR-optimal interval for R_i. We obtain the generalized sequence R_{i+1} by replacing $I_i \subset R_i$ with the symbol \$. We define any set of generalized sequences obtained in this way as *monotone*. We also refer to a member of a monotone set as monotone. Let $R_0, R_1, \ldots, R_k$ be a monotone sequence of gerenalized returns sequences, and let $I_0, I_1, \ldots, I_k$ be the maximal GSSR-optimal intervals corresponding to each sequence. By construction, I_i is a maximal GSSR-optimal interval for R_i. We have defined $\mathcal{A}$ so that the SSR of the union of these intervals in R_0 is given by

$$\mathsf{S}_{R_0}(I_0 \cup I_1 \cup \cdots \cup I_k) = \frac{\sum_{i=1}^{d} \mathcal{A}_{R_i}(I_i)}{\sum_{i=1}^{d} B(I_i)},$$

where the subscript R_i indicates on which generalized return sequence the quantity is computed.

Lemma 2.6.9 $\bar{\mathsf{S}}_{R_i}(I_i) \geq \bar{\mathsf{S}}_{R_{i+1}}(I_{i+1})$

Proof: Suppose that $\bar{\mathsf{S}}_{R_i}(I_i) < \bar{\mathsf{S}}_{R_{i+1}}(I_{i+1})$, and let $\$_i$ be the symbol that replaced I_i in R_i to obtain R_{i+1}. If I_{i+1} is not adjacent with $\$_i$, then I_i is not GSSR-optimal in R_i, a contradiction. If I_{i+1} is adjacent with $\$_i$, then $I_i \cup I_{i+1}$ has higher GSSR (by Lemma 2.6.1), so once again I_i is not GSSR-optimal in R_i. ■

Now an easy induction, using Lemmas 2.6.1 and 2.6.9 gives,

Corollary 2.6.10 $\mathsf{S}_{R_0}(I_0 \cup I_1 \cup \cdots \cup I_k) \geq \bar{\mathsf{S}}_{R_k}(I_k)$ *for any* k.

Analogous to Propositions 2.6.4, 2.6.7, 2.6.8, we have the following three propositions. Their proofs are almost identical, so we omit them.

Proposition 2.6.11 *A GSSR-optimal strategy making one trade exists, and all maximal GSSR-optimal trades can be found in* $O(NlogN)$ *time.*

Proposition 2.6.12 *Let* J *be a subinterval of any GSSR-optimal interval* I*. Then* $\mu(J) \geq -f_{sp}$*. If* J *is a prefix or suffix of* I *that is not adjacent with the symbol "$", then* $\mu(J) > 0$*.*

Proposition 2.6.13 *Let* I_0 *be any GSSR-optimal interval, and let* $\mathcal{T}$ *be any generalized trading strategy. Let* $\mathcal{T}' = I_0 \cup \mathcal{T}$*. Then,* $\mathcal{A}(\mathcal{T}') \geq \mathcal{A}(\mathcal{T})$ *and* $\bar{\mathsf{S}}(\mathcal{T}') \geq \bar{\mathsf{S}}(\mathcal{T})$*.*

We now give the main result that will lead to the final algorithm to obtain the $\overline{\mathsf{Shrp}}_2$-optimal strategy. Its essential content is that given a monotone set of generalized returns sequences, $R_0, R_1, \ldots$, with corresponding GSSR-optimal intervals $I_0, I_1, \ldots$, for some k, $\mathcal{T} = I_0 \cup I_1 \cup \cdots \cup I_k$ is $\overline{\mathsf{Shrp}}_2$ optimal. We will need some preliminary results.

Proposition 2.6.14 *For some* k*,* $\mathcal{T}^* = I_0 \cup I_1 \cup \cdots \cup I_k$ *is* $\overline{\mathsf{Shrp}}_2$*-optimal., where* I_i *are the GSSR-optimal intervals corresponding to a monotone set of generalized returns sequences.*

Proof. First we show that there exists a $\overline{\mathsf{Shrp}}_2$-optimal strategy $\mathcal{T}_0$ that contains I_0. Indeed, let $\mathcal{T}$ be any $\overline{\mathsf{Shrp}}_2$-optimal strategy, and consider $\mathcal{T}_0 = \mathbb{I}_0 \cup \mathcal{T}$. By the Proposition 2.6.8, we have $\mathsf{S}(\mathcal{T}_0) \geq \mathsf{S}(\mathcal{T})$ and $A(\mathcal{T}_0) \geq A(\mathcal{T}) \geq 0$. Then,

$$\overline{\mathsf{Shrp}}_2(\mathcal{T}_0) - \overline{\mathsf{Shrp}}_2(\mathcal{T}) = \frac{A(\mathcal{T}_0)A(\mathcal{T})(A(\mathcal{T}_0) - A(\mathcal{T})) + B(\mathcal{T}_0)B(\mathcal{T})(\mathsf{S}(\mathcal{T}_0) - \mathsf{S}(\mathcal{T}))}{(B(\mathcal{T}_0) - A^2(\mathcal{T}_0))(B(\mathcal{T}) - A^2(\mathcal{T}))} \geq 0,$$

thus, $\mathcal{T}_0$ is $\overline{\mathsf{Shrp}}_2$-optimal.

Let $\mathcal{T}_k$ be a $\overline{\mathsf{Shrp}}_2$-optimal strategy that contains $I_0 \cup \cdots \cup I_k$. We know that $\mathcal{T}_0$ exists. If $\mathcal{T}_k = I_0 \cup \cdots \cup I_k$, then we are done. If , $\mathcal{T}_k = I_0 \cup \cdots \cup I_k \cup \mathcal{T}'$, with $\mathcal{T}' \neq \emptyset$, then we show that there must exist a $\overline{\mathsf{Shrp}}_2$-optimal strategy $\mathcal{T}_{k+1}$ which contains $I_0 \cup \cdots \cup I_{k+1}$, i.e., there is some other $\mathcal{T}'' \supseteq I_{k+1}$ such that $\mathcal{T}_{k+1} = I_0 \cup \cdots \cup I_k \cup \mathcal{T}''$ is $\overline{\mathsf{Shrp}}_2$-optimal. The proposition then follows by an easy induction.

Let $\mathcal{T}'' = \mathcal{T}' \cup I_{k+1}$. Then, $\mathcal{A}_{R_{k+1}}(\mathcal{T}'') \geq \mathcal{A}_{R_{k+1}}(\mathcal{T}')$ and $\bar{\mathsf{S}}_{R_{k+1}}(\mathcal{T}'') \geq \bar{\mathsf{S}}_{R_{k+1}}(\mathcal{T}')$ (Proposition 2.6.13). By Corollary 2.6.10 and the GSSR-optimality of I_{k+1}, we have that

$$\mathsf{S}(I_0 \cup \ldots \cup I_k) \geq \bar{\mathsf{S}}_{R_{k+1}}(I_{k+1}) \geq \bar{\mathsf{S}}_{R_{k+1}}(\mathcal{T}'') \geq \bar{\mathsf{S}}_{R_{k+1}}(\mathcal{T}')$$

From now on, we will drop the R_{k+1} subscript. Let $A = A(I_0 \cup \ldots \cup I_k)$, $B = B(I_0 \cup \ldots \cup I_k)$, $A' = \mathcal{A}(\mathcal{T}')$, $B' = B(\mathcal{T}')$, $A'' = \mathcal{A}(\mathcal{T}'')$ and $B'' = B(\mathcal{T}'')$. Let $\overline{\mathsf{Shrp}_2} = \overline{\mathsf{Shrp}_2}(I_0 \cup \ldots \cup I_k)$, $\overline{\mathsf{Shrp}_2}' = \overline{\mathsf{Shrp}_2}(I_0 \cup \ldots \cup I_k \cup \mathcal{T}')$ and $\overline{\mathsf{Shrp}_2}'' = \overline{\mathsf{Shrp}_2}(I_0 \cup \ldots \cup I_k \cup \mathcal{T}'')$. Thus,

$$\overline{\mathsf{Shrp}_2} = \frac{A}{B - A^2}, \qquad \overline{\mathsf{Shrp}_2}' = \frac{A + A'}{B + B' - (A + A')^2}, \text{ and} \qquad \overline{\mathsf{Shrp}_2}'' = \frac{A + A''}{B + B'' - (A + A'')^2}.$$

Let $A'' = \alpha'' A$ and $A' = \alpha' A$, where $\alpha'' \geq \alpha' > 0$. Then, by direct computation, one obtains

$$\overline{\mathsf{Shrp}_2}' = \frac{A}{B + A^2 + \frac{\alpha'}{1+\alpha'}(\frac{B'}{\alpha'} - B - A^2)}, \text{ and} \qquad \overline{\mathsf{Shrp}_2}'' = \frac{A}{B + A^2 + \frac{\alpha''}{1+\alpha''}(\frac{B''}{\alpha''} - B - A^2)},$$

Since $\overline{\mathsf{Shrp}_2}' > \overline{\mathsf{Shrp}_2}$, we conclude that $\frac{B'}{\alpha'} - B - A^2 < 0$. Since $\bar{\mathsf{S}}(\mathcal{T}'') \geq \bar{\mathsf{S}}(\mathcal{T}')$, we have that $\frac{B'}{\alpha'} \geq \frac{B''}{\alpha''}$, and since $\alpha'' \geq \alpha' > 0$, $\frac{\alpha''}{1+\alpha''} \geq \frac{\alpha'}{1+\alpha'} > 0$, therefore

$$\frac{\alpha''}{1+\alpha''}\left(\frac{B''}{\alpha''} - B - A^2\right) \leq \frac{\alpha'}{1+\alpha'}\left(\frac{B'}{\alpha'} - B - A^2\right) < 0,$$

and so $\overline{\mathsf{Shrp}_2}'' \geq \overline{\mathsf{Shrp}_2}'$, concluding the proof. ■

By Proposition 2.6.14, a $\overline{\mathsf{Shrp}_2}$-optimal trading strategy can be obtained by constructing the strategies $\mathcal{T}_k$, and then picking the one with the maximum value for $\overline{\mathsf{Shrp}_2}$. The next proposition shows that this can be done in $O(N^2 logN)$ time.

Proposition 2.6.15 *A $\overline{\mathsf{Shrp}_2}$-optimal trading strategy can be found in time $O(n^2 logn)$.*

Proof: I_i can be obtained in $O(n \log n)$ time (Proposition 2.6.11). Since there are at most n such intervals (since each must be non-empty), obtaining all the intervals is in $O(n^2 \log n)$.
Given the intervals, a single scan can be used to obtain the k for which $\mathcal{T}_k$ is $\overline{\mathsf{Shrp}_2}$-optimal. ■

One can improve the runtime to $O(n^2)$ if $O(n^2)$ memory is available, however, we do not discuss the details.

2.6.3 Approximation Ratio for Shrp_2

We have given an algorithm that obtains a $\overline{\mathsf{Shrp}_2}$-optimal strategy. A modification to the algorithm constructs the hierarchy $\mathcal{T}_i$ and pick the one with the highest one with value of Shrp_2. Suppose we have a $\overline{\mathsf{Shrp}_2}$-optimal strategy $\mathcal{T}$ and let $\mathcal{T}^*$ be a Shrp_2-optimal strategy. Then by Proposition 2.6.6, it must be that $A(\mathcal{T}^*) > A(\mathcal{T})$ and that $\mathsf{S}(\mathcal{T}^*) \leq \mathsf{S}(\mathcal{T})$. Since $\overline{\mathsf{Shrp}_2}(\mathcal{T}) \geq \overline{\mathsf{Shrp}_2}(\mathcal{T}^*)$, we have that $A^*(B - A^2) - A(B^* - A^{*2}) \leq 0$, where $A = A(\mathcal{T}), A^* = A(\mathcal{T}^*), B = B(\mathcal{T}), B^* = B(\mathcal{T}^*)$. We can evaluate $\mathsf{Shrp}_2(\mathcal{T}^*) - \mathsf{Shrp}_2(\mathcal{T})$ to obtain

$$0 \leq \mathsf{Shrp}_2(\mathcal{T}^*) - \mathsf{Shrp}_2(\mathcal{T}) \leq \mathsf{Shrp}_2^* \cdot \frac{\frac{d}{n} {f_{sp}}^2}{\frac{d}{n} {f_{sp}}^2 + B - A^2}$$

When $B - A^2 = O(1)$, which is usually the case, we see that this is a very accurate approximation (since $f_{sp} \ll 1$).

2.6.4 Maximizing Shrp_1

Once again, we introduce the slightly different quantity $\overline{\mathsf{Shrp}}_1$,

$$\mathsf{Shrp}_1(\mathcal{T}) = \frac{A(\mathcal{T})}{\sqrt{\frac{d}{n}{f_{sp}}^2 + B(\mathcal{T}) - A^2(\mathcal{T})}}, \qquad \overline{\mathsf{Shrp}}_1(\mathcal{T}) = \frac{A(\mathcal{T})}{\sqrt{B(\mathcal{T}) - A^2(\mathcal{T})}}.$$

We will optimize $\overline{\mathsf{Shrp}}_1(\mathcal{T})$. Since maximizing $\overline{\mathsf{Shrp}}_1(\mathcal{T})$ is equivalent to minimizing $1/\overline{\mathsf{Shrp}}_1^2(\mathcal{T})$ the problem reduces to maximizing

$$Q(\mathcal{T}) = \frac{A^2(\mathcal{T})}{B(T)}$$

The entire algorithm is analogous to that for maximizing $\overline{\mathsf{Shrp}}_2$ in the previous section, we only need to prove the analogs of Propositions 2.6.6 and 2.6.14.

Proposition 2.6.16 *$\exists 0 \leq \alpha \leq \mu^*$ such that the constrained SSR-optimal strategy $\mathcal{T}_\alpha$ is Q-optimal.*

Proof: Let $\mathcal{T}$ be $\overline{\mathsf{Shrp}}_1$-optimal, and let $\alpha^* = A(\mathcal{T})$. Let $\mathcal{T}_{\alpha^*}$ be a corresponding constrained SSR-optimal strategy. $A(\mathcal{T}_{\alpha^*}) \geq A(\mathcal{T})$ and $\frac{A(\mathcal{T}_{\alpha^*})}{B(\mathcal{T}_{\alpha^*})} \geq \frac{A(\mathcal{T})}{B(\mathcal{T})}$. Multiplying these two inequalities gives that $\frac{A^2(\mathcal{T}_{\alpha^*})}{B(\mathcal{T}_{\alpha^*})} \geq \frac{A^2(\mathcal{T})}{B(\mathcal{T})}$, i.e. $\mathcal{T}_{\alpha^*}$ is also Q-optimal. ■

Proposition 2.6.17 *For some k, $\mathcal{T}^* = I_0 \cup I_1 \cup \cdots \cup I_k$ is Q-optimal., where I_i are the GSSR-optimal intervals corresponding to a monotone set of generalized returns sequences.*

Proof: The proof is very similar to tho proof of Proposition 2.6.14. Let $\mathcal{T}$ be Q-optimal, and let $\mathcal{T}_0 = I_0 \cup \mathcal{T}$. Then $A(\mathcal{T}_0) \geq A(\mathcal{T})$ and $\mathsf{S}(\mathcal{T}_0) \geq \mathsf{S}(\mathcal{T})$. Multiplying these two inequalities give that $Q(\mathcal{T}_0) \geq Q(\mathcal{T})$, or that $\mathcal{T}_0$ is also Q-optimal.
Let $\mathcal{T}_k$ be a Q-optimal strategy that contains $I_0 \cup \cdots \cup I_k$. Introduce $\mathcal{T}', \mathcal{T}'' = I_{k+1} \cup \mathcal{T}'$ as in the proof of Proposition 2.6.14. Let $Q = Q(I_0 \cup \ldots \cup I_k)$, $Q' = Q(I_0 \cup \ldots \cup I_k \cup \mathcal{T}')$ and $Q'' = Q(I_0 \cup \ldots \cup I_k \cup \mathcal{T}'')$,

$$Q = \frac{A^2}{B}, \qquad Q' = \frac{(A + A')^2}{B + B'}, \text{ and} \qquad Q'' = \frac{(A + A'')^2}{B + B''}.$$

Following exactly the same logic as in the proof to Proposition 2.6.14, we only need to show that $Q'' \geq Q'$. Let $A'' = \alpha'' A$ and $A' = \alpha' A$, where $\alpha'' \geq \alpha' > 0$. $\frac{A''}{B''} \geq \frac{A'}{B'}$ implies that $\frac{B''}{\alpha''} \leq \frac{B'}{\alpha'}$, and so $\frac{B''}{\alpha''(\alpha''+2)} \leq \frac{B'}{\alpha'(\alpha'+1)}$. By direct computation, one obtains

$$Q' = \frac{A^2}{B + \left(1 - \frac{1}{(1+\alpha')^2}\right)\left(\frac{B'}{\alpha'(\alpha'+2)} - B\right)}, \qquad Q'' = \frac{A^2}{B + \left(1 - \frac{1}{(1+\alpha'')^2}\right)\left(\frac{B''}{\alpha''(\alpha''+2)} - B\right)}.$$

Since $Q' > Q$, it must be that $\frac{B'}{\alpha'(\alpha'+2)} - B < 0$. Since $\alpha'' \geq \alpha'$, $1 - \frac{1}{(1+\alpha'')^2} \geq 1 - \frac{1}{(1+\alpha')^2}$, so we have that

$$\left(1 - \frac{1}{(1+\alpha'')^2}\right)\left(\frac{B''}{\alpha'(\alpha''+2)} - B\right) \leq \left(1 - \frac{1}{(1+\alpha')^2}\right)\left(\frac{B'}{\alpha'(\alpha'+2)} - B\right) < 0,$$

which implies that $Q'' \geq Q'$. ■

2.6.5 Approximation Ratio for Shrp_1

Once again, a modification to the algorithm constructs the hierarchy $\mathcal{T}_i$ and picks the one with the highest one with value of Shrp_1. Suppose we have a $\overline{\mathsf{Shrp}}_1$-optimal strategy $\mathcal{T}$ and let $\mathcal{T}^*$ be a Shrp_1-optimal strategy. By direct computation, and using the fact that $\overline{\mathsf{Shrp}}_1(\mathcal{T}) \geq \overline{\mathsf{Shrp}}_1(\mathcal{T}^*) \implies A^{*2}B - A^2B^* \leq 0$, we get

$$0 \leq \mathsf{Shrp}_1^2(\mathcal{T}^*) - \mathsf{Shrp}_1^2(\mathcal{T}) \leq \mathsf{Shrp}_1^2(\mathcal{T}^*)\frac{\frac{{df_{sp}}^2}{n}}{\frac{{df_{sp}}^2}{n} + B}$$

which gives an approximation ratio of $\sqrt{1 - O({f_{sp}}^2)}$ when $B = O(1)$.

2.7 Conclusion and Discussion

The main goal of this chapter was to provide the theoretical basis for the computation of *a posteriori* optimal trading strategies, with respect to various criteria. The highlights of our contributions are that return and MDD-optimal strategies can be computed very efficiently, even with constraints on the number of trades. Sharpe optimal strategies prove to be much tougher to compute. However, for slightly modified notions of the Sharpe ratio, where one ignores the impact of bid-ask spread squared we can compute the optimal strategy efficiently. This is a reasonable approach since in most cases, the bid-ask spread is $\sim 10^{-4}$. We also show that this modified optimal strategy is not far from optimal with respect to the unmodified Sharpe ratio.

We have introduced a new technique for optimizing quotients over intervals of a sequence. This technique is based on relating the problem to convex set operations, and for our purposes has direct application to optimizing the MDD, the simplified Sharpe ratio (SSR), which is an integral component in optimizing the Sharpe ratio, and the Downside Deviation Ratio (DDR). This technique may be of more general use in optimizing other financially important criteria.

A natural open problem is whether Sharpe optimal strategies can be computed under constraints on the number of trades.

Chapter 3

Learning: Optimal Linear Separation

3.1 Introduction and Basic Definitions

3.1.1 Convex Hulls

The convex hull of a set of points is the smallest convex set that contains the points. The convex hull is a fundamental construction for mathematics and computational geometry. Other problems can be reduced to the convex hull - halfspace intersection, Delaunay triangulation, Voronoi diagrams and power diagrams. Aurenhammer in [2] describes applications of these structures in mesh generation, file searching, cluster analysis, collision detection, crystallography, metallurgy, urban planning, cartography, image processing, numerical integration, statistics, sphere packing and point location. More formally,

Definition 3.1.1 *A set* $S \subseteq \mathbb{R}^n$ *is* **convex** *if* $\forall \mathbf{x}, \mathbf{y} \in S$ *and* $0 \leq \lambda \leq 1$

$$\lambda \mathbf{x} + (1 - \lambda)\mathbf{y} \in \mathcal{S}$$

Definition 3.1.2 *Let* $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^2$. *The* **Convex hull** $\mathcal{H}(\mathcal{X})$ *is the convex set such that* $\mathcal{X} \subset \mathcal{H}(\mathcal{X})$ *and if* $\mathcal{Y}$ *is another convex set for which this is true then* $\mathcal{H}(\mathcal{X}) \subseteq \mathcal{Y}$, *in other words convex hull is defined as smallest convex set containing* $\mathcal{X}$.

In two dimensions, the convex hull takes on a specific form.

Definition 3.1.3 *A* **polygon** *is a closed path of straight line segments in* $\mathbb{R}^2$. *These segments are also called edges of the polygon, and the intersection of two adjacent edges is a vertex of the polygon. A* **simple** *polygon is one which has no intersecting non-adjacent edges. Every simple polygon divides* $\mathbb{R}^2$ *into an interior and an exterior region.*

It is easy to see that a simple polygon is **convex** if and only if the internal angle formed at each vertex is smaller than or equal to π. The **convex hull** of a set of points $\mathcal{S} \in \mathbb{R}^2$ is the convex polygon with smallest area that encloses S . Note that in two dimensions, the convex hull can be represented by ordered sequence $\mathbf{u}_1\mathbf{u}_2 \ldots \mathbf{u}_k\mathbf{u}_1 \in \mathcal{A}$ where the boundary of $\mathcal{H}(\mathcal{A})$ is the union of lines $\mathbf{u}_i\mathbf{u}_{i+1}$.

Definition 3.1.4 *Point $\mathbf{x} \in \mathcal{X}$ is a* **limit point** *of $\mathcal{X}$, if and only if $\forall \epsilon > 0$, the ϵ-neighborhood of $\mathbf{x}$ contains both points from $\mathcal{X}$ and points that do not belong to $\mathcal{X}$. Let $\mathcal{A} = \{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n\}$. The* **vertices** *of the convex hull $\mathcal{H}(\mathcal{A})$ are the points from $\mathcal{A}$ which are bound points of $\mathcal{H}(\mathcal{A})$.*

Convex hulls were studied extensively in the literature. Timothy Chan in [14] presented state-of-the-art algorithms for computing the convex hull of a set in two or three dimentions. He proved the following output-sensitive time bounds. By the reduction from sorting, these time bounds are optimal.

Fact 3.1.5 *The convex hull for a given set of points in two or three dimensions can be computed in time $O(N \log h)$, where h is the number of vertices of the convex hull.*

3.1.2 Linear Separators

Convex hulls are prominent in areas as Neural Networks, Support Vector Machines, clustering, pattern recognition, etc. One is usually interested in finding an optimal separating plane for two convex hulls (in n dimensions). In wide range of applications to robotics, computer graphics, simulation and computational geometry, intersection tests comprise some of the most fundamental issues.

The problem of learning - discovering hidden patterns in the collected data - often can be considered as the problem of finding a linear separator for two sets of points in multidimensional space. When the sets are not separable, we are interested in finding a subset of points such that after removing these points, the remaining points are separable. Such a set is called a separator set. If we assign a positive weight to every point (its fidelity), then we wish to find a separator set with the minimum possible weight. Optimal linear separator also can be used for optimal boosting of two classifiers [44]. We discuss the optimal boosting of two classifiers in section 3.4.

Optimal linear separation plays a key role in pattern recognition and computational geometry. In statistical pattern recognition, a resurgence of linear separability has emerged through its role in Support Vector Machines and other kernel based methods [18, 75]. In computational geometry, determining the intersection of sets (mostly in two or three dimensions) is of immense practical importance, for example in CAD systems, and thus linear separability is of fundamental use in this setting. To make the discussion more concrete, we need some definitions.

Throughout, we assume that we are in two or three dimensional Euclidean space.

Definition 3.1.6 *Sets $\mathcal{A} \subset \mathbb{R}^n, \mathcal{B} \subset \mathbb{R}^n$ are* **linearly separable** *iff $\exists v \in \mathbb{R}^n, v_0 \in \mathbb{R}$ such that*

$$\begin{aligned} \mathbf{v}^T \mathbf{x}_{\mathcal{A}} + v_0 > 0, \quad \forall \mathbf{x}_{\mathcal{A}} \in \mathcal{A} \\ \mathbf{v}^T \mathbf{x}_{\mathcal{B}} + v_0 < 0, \quad \forall \mathbf{x}_{\mathcal{B}} \in \mathcal{B} \end{aligned}$$

The pair $(\mathbf{v}, v_0)$ defines an *(oriented) separating hyperplane.*

Definition 3.1.7 *Let $\mathcal{A}, \mathcal{B} \subset \mathbb{R}^n$. The distance* $d(\mathcal{A}, \mathcal{B}) = \inf_{\mathbf{x} \in \mathcal{A}, \mathbf{y} \in \mathcal{B}} \| \mathbf{x} - \mathbf{y} \|$

If two sets, $\mathcal{A}, \mathcal{B}$, are linearly separable, then the *optimal* or *maximum margin* separating hyperplane is a separating hyperplane with maximum possible distance from both sets.

Definition 3.1.8 *Suppose that $\mathcal{A}, \mathcal{B} \subset \mathbb{R}^n$ are linearly separable. Then the **optimal separating hyperplane** $\sigma(\mathcal{A}, \mathcal{B})$ is a separating hyperplane with maximum possible distance from both sets. Pair of points $s \in \mathcal{A}, t \in \mathcal{B}$ is a **realization of separation** if $d(s,t) = d(\mathcal{A}, \mathcal{B})$ and $d(s, \sigma(\mathcal{A}, \mathcal{B})) = d(t, \sigma(\mathcal{A}, \mathcal{B})) = \frac{1}{2} d(s,t)$.*

Suppose that *weighting function* W assigns a weight $W(\mathbf{x}) > 0$ to point $\mathbf{x}$. For hyperplane $\ell = (\mathbf{v}, v_0)$, let $\mathcal{Q}(\ell) = \mathcal{Q}_{\mathcal{A}} \cup \mathcal{Q}_{\mathcal{B}}$ denote the mis-classified points, where $\mathcal{Q}_{\mathcal{A}} = \{\mathbf{x} \in \mathcal{A} | \mathbf{v}^T \mathbf{x} + v_0 \leq 0\}$ and $\mathcal{Q}_{\mathcal{B}} = \{\mathbf{x} \in \mathcal{B} | \mathbf{v}^T \mathbf{x} + v_0 \geq 0\}$. We define the weight (or error) $\mathcal{E}(\ell)$ as the weight of mis-classified points, $\mathcal{E}(\ell) = \sum_{\mathbf{x} \in \mathcal{Q}(\ell)} W(\mathbf{x})$. Note that the sets $\mathcal{A}' = \mathcal{A} \backslash \mathcal{Q}_{\mathcal{A}}$ and $\mathcal{B}' = \mathcal{B} \backslash \mathcal{Q}_{\mathcal{B}}$ are linearly separable, and ℓ is a separator for them.

Definition 3.1.9 (Optimal Fat Separator) *A hyperplane $\ell = (\mathbf{v}, v_0)$ is optimal if it has minimum weight. It is an optimal fat separator if it has minimum weight and separates $\mathcal{A}'$ and $\mathcal{B}'$ with maximum margin.*

Intuitively, the optimal fat separator ℓ is the hyperplane with minimum error such that if the mis-classified points are viewed as noisy and have their class flipped, then the entire set becomes separable, and ℓ is a maximum margin separator for the "noise-corrected" set.

The goal of this chaper is to develop an efficient algorithm for exact linear separation, where "exact" means globally optimal with respect to some (arbitrarily specified) error criterion. Constructed algorithms are applicable to the case where the data points are *not* linearly separable.

In the next sections we will discuss previous related work.

3.2 Related Work

3.2.1 Convex Hulls

One of the simplest $O(N \log N)$ – algorithms for computation of the convex hull for given a set of points on the plane is the **Graham's Scan Algorithm**. An online demonstration of the Graham's Scan Algorithm can be found at [54]. Under certain conditions on probability density function, this algorithm has $O(n)$ expected time complexity [20].

Algorithm 3.2.1 Graham's Scan Algorithm [30]
The algorithm works in three phases:

1. *Find an extreme point. This point will be the **pivot**, is guaranteed to be on the hull, and is chosen to be the point with largest y coordinate.*
2. *Sort the points in order of increasing angle about the pivot. We end up with a star-shaped polygon (one in which one special point, in this case the pivot, can "see" the whole polygon).*
3. *Build the hull, by marching around the star-shaped polygon, adding edges when we make a left turn, and back-tracking when we make a right turn.*

Another popular algorithm for computing the convex hull for given a set of points on the plane is the **Quick-Hull Algorithm**. An online demonstration of the Quick-Hull Algorithm can be found at [55]. Although it's worst-case complexity is quadratic, typically the algorithm works fast on random sets of points, and is similar to quick-sort:

- it is recursive
- each recursive step partitions data into several groups

Algorithm 3.2.2 Quick-Hull Algorithm [4]
The partitioning step does all the work. The basic idea is as follows:

1. *We are given a some points, and line segment AB which we know is a chord of the convex hull (i.e., it's endpoints are known to be on the convex hull). A good chord to start the algorithm goes from the leftmost to the rightmost point in the set.*
2. *Among the given points, find the one which is farthest from AB. Let's call this point C.*
3. *The points inside the triangle ABC cannot be on the hull. Put them in set S_0.*
4. *Put the points which lie outside edge AC in set $\mathcal{S}_1$, and points outside edge BC in set $\mathcal{S}_2$.*

Once the partitioning is done, we recursively invoke quick-hull on sets $\mathcal{S}_1$ and $\mathcal{S}_2$. The algorithm works fast on random sets of points because step 3 of the partition typically discards a large fraction of the points.

Timothy Chan in [14] presented state-of-the-art algorithms for computing the convex hull of a set in two or three dimentions. He proved the following output-sensitive time bounds.

Fact 3.2.3 *The convex hull for a given set of points in two or three dimensions can be computed in time $O(N \log h)$, where h is the number of vertices of the convex hull.*

The problem of randomized computation of the convex hull in two dimensions has been well-studied and several randomized incremental algorithms have been developed, with linear expected running time (when the probability distribution satisfies certain constraints) [21] [3].

3.2.2 Separable Sets

When two sets of points are separable, an approach to constructing the maximum margin separator is to first construct the convex hulls, and then construct the maximum margin separator for the convex hulls. In 2 and 3 dimensions, this approach is very efficient. The maximum margin separator can be specified as the orthogonal bisector of the line joining two points on the convex hulls of the two sets. These two points are sometimes refered to as a *realization* of the maximum margin separator.

Dobkin and Kirkpatrick, [23], introduced hierarchical representations for convex hulls and established many useful properties of such representations [25, 24, 26]. Specifically, given a standard representation of a convex hull (in 2 or 3 dimensions), a compact hierarchical representation of can be constructed in *linear* time. This representation has been exploited in a series of subsequent papers dealing with separation of polytopes[23], generalized extremal queries and applications, intersection of convex and non-convex polyhedra, intersection of convex bodies with curved edges and faces, parallel algorithms for manipulation of polytopes, applications in computer graphics etc. [26].

In particular, they construct a *sublinear* deterministic algorithm for obtaining the optimal linear separator for separable convex hulls in 2 and 3 dimensions (assuming that compact hierarchical representations for both convex hulls are available):

Fact 3.2.4 *([23]) The optimal linear separator $\sigma(\mathcal{P}, \mathcal{Q})$ (and its realization) of convex hulls on the plane $\mathcal{P}$, $\mathcal{Q}$ can be determined $O(log|\mathcal{P}| + log|\mathcal{Q}|)$ time from their hierarchical representations, where $|\mathcal{P}|$ ($|\mathcal{Q}|$) is the number of vertices of $\mathcal{P}$ ($\mathcal{Q}$).*

Using the linear algorithm for constructing the hierarchical representations combined with Fact 3.2.4, one obtains an efficient deterministic algorithm for constructing the maximum margin separator for *separable* sets in 2 and 3 dimensions:

Fact 3.2.5 ([23], [15]) *The optimal linear separator (in 2 and 3 dimensions), and its realization, for two separable sets $\mathcal{A}$ and $\mathcal{B}$ can be found in $O(n \log n)$ operations.*

Generalizing results of Dobkin and Kirkpatrick to $d > 3$ dimensions is difficult, and a more popular approach is to re-formulate the linear separability problem as a linear program or the maximum margin separator problem as a quadratic program. Such problems can be handled using linear/convex programming techniques such as: the simplex method [19, 16], with complexity $O(N^2)$ where the constant is exponential in d (in practice the simplex method has linear average-case complexity [71]); or, interior point methods [22, 38, 39, 52, 49].

3.2.3 Unseparable Sets

Our work addresses the case when $\mathcal{A}$ and $\mathcal{B}$ are not linearly separable (i.e., their convex hulls intersect). In this case we are interested in finding a subset $\mathcal{Q}$ of points such that after removing points from $\mathcal{Q}$ remaining points are separable.

Combinatorial approach plagued by the exponential growth og running time. Heuristics based on greedy search that seeks local inprovement may trap the solution in a local minimum which is much worse than the true global minimum [49].

Popular approaches are to formulate some differentiable error as a function of the distance of a mis-classified point from the hyperplane. One then seeks to minimize some heuristic function of this error, [75, 5]. If the resulting error function is convex, then convex optimization techniques can be brought to bear, for example in [75] one obtains a convex quadratic program. Bennet and Mangasarian [6] propose minimizing the average distance from mis-classified points using linear programming. Most often, however, such heuristic errors are minimized using iterative algorithms.

Another approach, suggested by Bennett and Bredensteiner in [5] use following heuristic to deal with this problem. If most points of one class are not in convex hull of the other, then we could restrict the influence of outlying points and solve the problem on reduced convex hulls. The intuition is that it is undesirable to let one point excessively influence the solution. Therefore, we want the solution to be based on a lot of points, not just a few bad ones. Say we want the solution to depend on at least K points. This can be done by contracting or reducing the convex hull by putting an upper bound on the multiplier in the convex combination for each point. *Reduced Convex Hull* of set of points $\mathcal{A}$ is the set of all convex combinations $c = Au$ of points in $\mathcal{A}$ where $e^T u = 1$, $0 \leq u \leq De$, $D < 1$. Typically we choose $D = \frac{1}{K}$ and $1 < K \leq |\mathcal{A}|$. K have to be chosen sufficiently large to ensure that the convex hulls do not intersect. Then problem solved for the reduced convex hulls via linear programming.

Interior point methods also can be applied to problems of discrete optimization. These methods do not confine their working to a discrete solution set, but instead view combinatorial objects

as limiting cases of continuous objects. Many computational breakthroughs in the approximate solution of large scale combinatorial optimization problems achieved recently are due to the development of interior point algorithms and implementations. Theoretical foundations and applications of interior point techniques to the development of algorithms for combinatorial optimization problems given in [49]. These techniques are superior to combinatorial heuristics and Simplex method for some problems. Interior point methods have provided a unified approach to create efficient algorithms for many different combinatorial problems. Successful approaches include an interior point branch and bound technique [11, 47], cutting plane technique [33, 40, 48], and semidefinite programming relaxations [35, 76].

In contrast to these existing approaches, our work focuses on producing globally optimal solutions in 2 dimensions: for an arbitrary weight function W, the problem cannot be represented as the minimization of some differentiable error (convex or not). Constructed algorithms output a minimum weight subset of the points $\mathcal{Q}$ such that after deleting these points, the remaining points are separable, and the algorithms given by Fact 3.2.5 can then be used.

3.3 Contribution

In 2 dimensions, we give *exact* algorithms for obtaining an optimal fat separator ℓ for two sets $\mathcal{A}$ and $\mathcal{B}$ with respect to an arbitrary weighting function W. In particular, if $W(\mathbf{x}) = 1$, then the resulting separator minimizes the classification error. With respect to set intersection, $\mathcal{Q}(\ell)$ is the minimum sized set that must be removed in order to make the remaining points separable, and can be viewed as the intersection of the two objects. Further, constructed algorithms also furnish an exact calculation of the leave-one-out error with no increase in computational complexity. Traditionally, if $n = |\mathcal{A} \cup \mathcal{B}|$ is the number of data points, the computation of the leave-one-out error incurs an extra factor n in the runing time.

Let $|A| = m$ and $|B| = k$, and assume without loss of generality that $m \leq k$. Then, the computational complexity of constructed algorithm is given by $O(mn \log n)$.

3.3.1 Optimal Linear Separator for Non-Separable Sets

Let $\mathcal{A} = \{\mathbf{a}_i\}_{i=1}^m$ and $\mathcal{B} = \{\mathbf{b}_j\}_{j=1}^k$ be two sets of points, with $m \leq k$, and let $n = m + k$. It is traditional to assign class $+1$ to one of the sets (say) $\mathcal{A}$ and -1 to the other. Every point $\mathbf{x}$ has a weight $W(\mathbf{x}) > 0$. A *separator set* $\mathcal{Q} \subseteq \mathcal{A} \cup \mathcal{B}$ is a set with the following property: if the points in $\mathcal{Q}$ are deleted, the remaining points are linearly separable. Every separator set $\mathcal{Q}$ has a weight, $W(\mathcal{Q}) = \sum_{\mathbf{x} \in \mathcal{Q}} W(\mathbf{x})$. An *optimal separator set* $\mathcal{Q}^*$ is one with minimum weight, i.e., for any other separator set $\mathcal{Q}$, $W(\mathcal{Q}) \geq W(\mathcal{Q}^*)$. A brute force search through all subsets of $\mathcal{A} \cup \mathcal{B}$ is clearly an exponential algorithm, and we present here a polynomial time algorithm for the 2 dimensional case.

Theorem 3.3.1 (2-dimensions) *An optimal fat separator and its corresponding optimal separator set $\mathcal{Q}(\mathcal{A}, \mathcal{B})$ can be found in $O(mn \log n)$ time.*

Proof: We first discuss the correspondence between optimal separator sets and hyperplanes. As already discussed, to every oriented line ℓ, we can associate the separator set $Q(\ell)$. The converse is also true for an optimal separator set. Specifically, let $\mathcal{Q}^*$ be an optimal separator set. Then $\mathcal{A}' = \mathcal{A} \setminus \mathcal{Q}^*$ and $\mathcal{B}' = \mathcal{B} \setminus \mathcal{Q}^*$ are linearly separable, so let ℓ be any separating hyperplane. Then no points of $\mathcal{Q}^*$ must be mis-classified, as otherwise Q^* is unnecessarily large, contradicting its optimality. Further, if any points of $\mathcal{Q}^*$ lie on ℓ, then by shifting ℓ a little, we still separate $\mathcal{A}'$ and $\mathcal{B}'$, however now we correctly classify some points of $\mathcal{Q}^*$, once again contradicting the optimality of Q^*. Thus, we have the following lemma,

Lemma 3.3.2 *Let $\mathcal{Q}^*$ be any optimal separator set. Any hyperplane ℓ that separates $(\mathcal{A} \cup \mathcal{B}) \setminus \mathcal{Q}^*$ also separates $\mathcal{Q}^*$, i.e., $\mathcal{Q}^*$ itself is separable; further, $\mathcal{Q}(\ell) = \mathcal{Q}^*$.*

In the proof of Lemma 3.3.2, we used the fact that *any* hyperplane ℓ that separates $(\mathcal{A} \cup \mathcal{B}) \setminus \mathcal{Q}^*$ is such that $\mathcal{Q}(\ell) = \mathcal{Q}^*$, and so in particular the maximum margin separator for $(\mathcal{A} \cup \mathcal{B}) \setminus \mathcal{Q}^*$ will have minimum weight. Thus, once we have found the optimal separator set, we can easily construct the optimal fat separator using the result in Fact 3.2.5.

Lemma 3.3.2 implies that any optimal separator set is the separator set of some hyperplane. Thus, it suffices to consider all possible hyperplanes, and their separator sets. Though it appears that we have increased the difficulty of our enumeration problem, we will now show that not all possible hyperplanes need be considered. In fact, we can restrict ourself to hyperplanes passing

through at least two points. This is a big step, because there are only $\Theta(n^2)$ such hyperplanes. The separator set for a given hyperplane can be computed in $O(n)$ operations, and so we immediately have an $O(n^3)$ algorithm. By being careful about reusing computations, we can reduce the complexity to $O(n^2 \log n)$, which is essentially the content of the Theorem 3.3.1.

We need to consider more carefully the definition of a separator set, especially when points lie on the hyperplane ℓ. According to the strict definition of separability, we would need to include all the points on ℓ into $\mathcal{Q}(\ell)$. We relax this condition in the definition of the *positive separator set* associated to the hyperplane ℓ.

Definition 3.3.3 *For hyperplane ℓ, the positive separator set $\mathcal{Q}^+(\ell)$ contains all mis-classified points* **except** *the positive points (in $\mathcal{A}'$) that lie on ℓ. ℓ is denoted the positive separator hyperplane of $\mathcal{Q}^+(\ell)$.*

The only difference between the usual separator set and the positive separator set is in how we treat the points that reside directly on the hyperplane ($\mathcal{Q}^+(\ell) \subseteq \mathcal{Q}(\ell)$).

Let $\mathcal{Q}^*$ be an optimal separator set, and let ℓ' be a hyperplane that separates $\mathcal{A}'$ and $\mathcal{B}'$ constructed from $\mathcal{A} \cup \mathcal{B} \setminus \mathcal{Q}^*$. By Lemma 3.3.2 $\mathcal{Q}(\ell') = \mathcal{Q}^*$ and ℓ' separates $\mathcal{Q}^*$. Let $\mathbf{a}^+$ be the closest positive point in $\mathcal{A}'$ to ℓ. Then all hyperplanes ℓ'' parallel to ℓ' that are closer to $\mathbf{a}^+$ and correctly classify $\mathbf{a}^+$ also separate $\mathcal{A}'$ and $\mathcal{B}'$. Hence, by Lemma 3.3.2 all such hyperplanes also separate $\mathcal{Q}^*$, i.e., for all such hyperplanes ℓ'', $\mathcal{Q}(\ell'') = \mathcal{Q}^*$. This means there are no points that are on any of these hyperplanes ℓ''. Now consider the hyperplane ℓ parallel ℓ' and containing $\mathbf{a}^+$, and consider $\mathcal{Q}^+(\ell)$. Any negative points on ℓ already belong to Q^*. Thus, $\mathcal{Q}^+(\ell) = \mathcal{Q}^*$. Suppose that ℓ contains at least one negative point. If it contains no other positive points (other than $\mathbf{a}^+$), then by slightly rotating ℓ about $\mathbf{a}^+$, we can classify some of these negative points correctly, without altering the classifications of $\mathcal{A}'$ or $\mathcal{B}'$. This contradicts the optimality of $\mathcal{Q}^*$, which gives the following lemma,

Lemma 3.3.4 *Let $\mathcal{Q}^*$ be any optimal separator set. Then there exists hyperplane ℓ such that $\mathcal{Q}^+(\ell) = Q^*$ and either:*

i. two or more positive points from $\mathcal{A}$ reside on ℓ;

ii. exactly one positive point from the $\mathcal{A}$ resides on ℓ, and no others.

Lemma 3.3.4 shows that it suffices to consider only the positive separator sets of hyperplanes that pass through at least one positive point. This forms the basis of the algorithm. We try every positive point as a candidate "central" point and compute the best possible separator set for all hyperplanes that pass through this central point. We then keep the best separator set over all possible central points.

Let's consider how to efficiently find the best positive separator hyperplane that contains some fixed positive point $\mathbf{a}^+$. In order to do so efficiently, we introduce a *mirrored-radial* coordinate system in which all the points except $\mathbf{a}^+$ can be linearly ordered with respect to $\mathbf{a}^+$.

We start with an arbitrary (base) vector $\mathbf{u}$ that defines an axis as shown in Figure 3.1. The origin of $\mathbf{u}$ is at $\mathbf{a}^+$. With $\mathbf{a}^+$ as origin, we define the angle $\theta(\mathbf{x})$ of a point $\mathbf{x}$ with respect to the base vector $\mathbf{u}$ as the angle between the two vectors $\mathbf{x} - \mathbf{a}^+$ and $\mathbf{u}$. The upper hemisphere of the unit circle is the set of points on the unit circle with angle in the range $[0, \pi]$ (shaded in Figure 3.1). We define the mirrored-radial projection of a point $\mathbf{s}(\mathbf{x})$ as the projection of $\mathbf{x}$ onto the upper

hemisphere of the unit circle, through the origin $\mathbf{a}^+$. The mirrored-radial projections of $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ are illustrated by $\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3$ in the Figure 3.1). The mirrored-radial coordinate $\theta(\mathbf{x})$ is then the angle of $\mathbf{s}(\mathbf{x})$, i.e., $\theta(\mathbf{s}(\mathbf{x}))$. Notice that many points may have the same mirrored-radial projection, in which case, they all have the same mirrored-radial coordinate.

Suppose that the mirrored radial coordinates of all the points (except $\mathbf{a}^+$) have been sorted and that $\mathbf{u}$ has been chosen so that $0 < \theta_1 \leq \theta_2 \cdots \leq \theta_{n-1} < \pi$. For convenience, define $\theta_0 = 0$ and $\theta_n = \pi$. An oriented hyperplane ℓ can also be uniquely specified by giving its angle θ_ℓ (see Figure 3.1), together with its orientation (± 1). For a given orientation, all hyperplanes with $\theta_\ell \in (\theta_i, \theta_{i+1})$, $0 \leq i < n$, partition the points into the same two sets, and hence have the same positive separator set. The other possible values for θ_ℓ are the actual mirrored-radial coordinates θ_i. Since there only two possible orientation for a given θ_ℓ, we have the following lemma

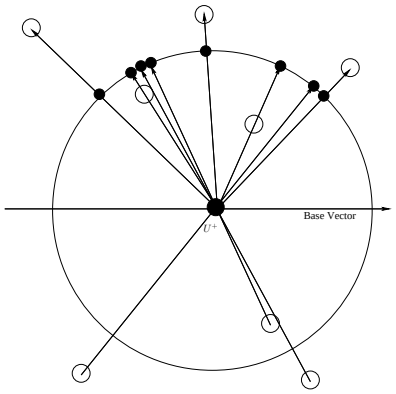


Figure 3.1: Mirrored-radial coordinates.

Lemma 3.3.5 *There are $4n-2$ possible equivalence classes of positive separator hyperplanes, corresponding to the following ranges for for θ_ℓ,*

$$\{(\theta_0, \theta_1), \theta_1, (\theta_1, \theta_2), \theta_2, \ldots, \theta_{n-1}, (\theta_{n-1}, \theta_n)\}.$$

For any two values of θ_ℓ from the same range, and for a given orientation, the positive separator sets are identical.

The importance of Lemma 3.3.5 is that we now only have to check one representative from each equivalence class. Further, this can be done very efficiently with two linear time scans (one for each orientation) as follows. Lets consider hyperplanes with orientation $+1$. First, we compute $\mathcal{Q}^+(\ell)$ for $\theta_\ell = 0$, and its weight. Now, we iteratively step through the values θ_i. When we process θ_i, some (or all) of the points with mirrored radial coordinates θ_i will move to the opposide side of ℓ, which will correspond to an update of $\mathcal{Q}^+(\ell)$ and the its weight $W(\mathcal{Q}^+(\ell))$. Assuming that set membership in $\mathcal{Q}^+(\ell)$ is maintained by an array of size $n-1$, if n_i points are moveed to the opposite side, then the update of $\mathcal{Q}^+(\ell)$ and $W(\mathcal{Q}^+(\ell))$ requires $O(n_i)$ operations. After processing θ_i, and come to process the range (θ_i, θ_{i+1}), once again at most n_i points shift sides, resulting in an update costing $O(n_i)$ operations. Thus the full scan takes $O(2\sum n_i) = O(n)$ operations. Since two scans need to be made, the total cost of these scans is $O(n)$.

Recap: For every positive point, $\mathbf{a}^+$, we first compute the mirrored-radial coordinates of all the other points, requiring $O(n)$ operations. We then sort these coordinates in $O(n \log n)$ operations. We now make two scans (in sorted order), one for each orientation of the hyperplane, updating the the best positive separator set and its weight as we scan. This requires $O(n)$ operations, Since the sorting operation has the dominant run time, this entire process is in $O(n \log n)$. Since this entire process has to be run for every positive point, and there are m positive points, we obtain a final computational complexity of $O(mn \log n)$, which completes the proof of the theorem. ■

3.3.2 Leave-One-Out Error

An important issue in the design of efficient machine learning systems is the estimation of the accuracy of learning algorithms, in particular its sensitivity to noisy inputs. One classical estimator

is *leave-one-out* error, which is commonly used in practice. Intuitively, the leave-one-out error is defined as the average error obtained by training a classifier on $n-1$ points and evaluating it on the point left out. For some learning algorithms, one can obtain estimates of the leave-one-out error, for example for Support Vector Machines, in the separable case, the leave one out error can be bounded in terms of the number of support vectors, [75]. Algorithmically, we remove one point, train the classifier and test in on the point left out. This process is repeated n times for every possible point that could be left out. The average error on the points left out is the leave-one-out error. More formally,

Let $\mathcal{X}$ denote all the points, $\mathcal{X} = \mathcal{A} \cup \mathcal{B}$. Let $\mathcal{X}^{(i)}$ denote the points with point $\mathbf{x}_i$ left out, $\mathcal{X}^{(i)} = \mathcal{X} \setminus \mathbf{x}_i$. Let $C^{(i)}$ denote the classifier built from $\mathcal{X}^{(i)}$ – in our case this is the optimal fat hyperplane. Let e_i denote the error of $C^{(i)}$ applied to the input point $\mathbf{x}_i$.

$$e_i = \begin{cases} 0 & \text{if } \mathbf{x}_i \text{ is classified correctly,} \\ W(\mathbf{x}_i) & \text{otherwise.} \end{cases}$$

The **leave-one-out** error, $\mathcal{E}_{loo}$ is given by $\mathcal{E}_{loo} = \frac{1}{n}\sum_{i=1}^{n} e_i$. We focus on the 2-dimensional case. A brute force computation of $\mathcal{E}_{loo}$ results in a factor of n increase in the run time, which would result in an $O(mn^2 \log n)$ algorithm. We show how to modify the algorithm so that it outputs $\mathcal{E}_{loo}$ with no increase in the computational complexity. We will establish the following theorem.

Theorem 3.3.6 (2-dimensions) *An optimal fat separator, together with its optimal separator set* $\mathcal{Q}(\mathcal{A}, \mathcal{B})$ **and** *the leave-one-out error can be be found in time* $O(n^2 \log n)$ *time.*

Let $\mathcal{Q}(\mathcal{X})$ be optimal separator set for set of points $\mathcal{X}$, and let $\mathcal{V}$ be any subset of $\mathcal{Q}(\mathcal{X})$. We consider the set $\mathcal{X}' = \mathcal{X} \setminus \mathcal{V}$, i.e., a set resulting from the removal of some part of an optimal separator set from the original set. Note that $\mathcal{Q}(\mathcal{X})$ is mis-classified by the optimal fat separator trained on $\mathcal{X}$. Consider $\mathcal{Q}(\mathcal{X}')$, i.e. an optimal separator set for the reduced set of points, and its corresponding fat separator hyperplane ℓ'. Certainly $\mathcal{Q}(\mathcal{X}) \setminus \mathcal{V}$ is a separator set for $\mathcal{X}'$, and so $W(\mathcal{Q}(\mathcal{X}')) \leq W(\mathcal{Q}(\mathcal{X})) - W(\mathcal{V})$. Now considering adding back the points in $\mathcal{V}$. If we add into $\mathcal{Q}(\mathcal{X}')$ all the points in $\mathcal{V}$ that ℓ' mis-classifies, then we get a separator set for $\mathcal{X}$. Suppose that ℓ' classifies any of the points in $\mathcal{V}$ correctly. Then the weight of the separator set $\mathcal{Q}(X')$ will increase by less than $W(\mathcal{V})$, which means we have constructed a separator set for $\mathcal{X}$ with smaller weight than $\mathcal{Q}(\mathcal{X})$, contradicting the optimality of $\mathcal{Q}(\mathcal{X})$. Thus, ℓ' must mis-classify every point in $\mathcal{V}$, and further, $W(\mathcal{Q}(\mathcal{X}')) = W(\mathcal{Q}(\mathcal{X})) - W(\mathcal{V})$.

Lemma 3.3.7 *Let* $\mathcal{V} \subseteq \mathcal{Q}(\mathcal{X})$ *and* $\mathcal{X}' = \mathcal{X} \setminus \mathcal{V}$. *Then* $W(\mathcal{Q}(\mathcal{X}')) = W(\mathcal{Q}(\mathcal{X})) - W(\mathcal{V})$, *and any separator hyperplane* ℓ' *associated to* $\mathcal{Q}(\mathcal{X}')$ *mis-classifies every point in* $\mathcal{V}$.

By lemma 3.3.7, we immediately have that

$$\mathcal{E}_{loo} = \frac{W(\mathcal{Q}(X))}{n} + \frac{1}{n} \sum_{x_i \in \mathcal{X} \setminus \mathcal{Q}(\mathcal{X})} e_i,$$

and so we immediately have a lower bound on $\mathcal{E}_{loo}$. Further, it suffices to compute e_i only for $\mathbf{x}_i \in \mathcal{X} \setminus \mathcal{Q}(\mathcal{X})$. With only a slight change in the algorithm given in the proof of the theorem 3.3.1 we can compute exactly the leave-one-out error. First observe that the following lemma holds.

Lemma 3.3.8 *If* $\mathcal{Q}$ *is a separator set for* $\mathcal{X}^{(i)}$, *then* $\mathcal{Q} \cup \mathbf{x}_i$ *is a separator set for* $\mathcal{X}$.

Thus, all separator sets of $\mathcal{X}^{(i)}$ are subsets of the separator sets of $\mathcal{X}$.

Point x_i from $\mathcal{X}$ can be one of three types:

Type 1. x_i always classified correctly by any optimal fat separator constructed for $\mathcal{X}^{(i)}$. Obviously, such a point x_i makes zero contribution to the leave-one-out error.

Type 2. x_i always misclassified by any optimal fat separator constructed for $\mathcal{X}^{(i)}$. Contribution of such a point to the leave-one-out error is equal to its weight $W(x_i)$.

Type 3. There are N_c representative lines for $\mathcal{X}^{(i)}$ such that optimal fat separators corresponding to these lines will classify x_i correctly and there are N_e representative lines for $\mathcal{X}^{(i)}$ such that optimal fat separators corresponding to these lines will misclassify x_i. Contribution of such a point to the leave-one-out error is equal to the weighted probability of choosing bad separator: $W(x_i)\frac{N_e}{N_e+N_c}$.

Thus, the leave-one-out error can be computed as follows: start with zero and add contribution of every point. Points of type 1 can be ignored, and we can concentrate only on points of type 2 and 3.

Lemma 3.3.9 *Let v be positive (negative) point from $\mathcal{X}$ such that there exists an optimal fat separator ℓ for $\mathcal{X}-v$ that misclassifies v. Then there exists positive (negative) separator line ℓ^* that passes through positive (negative) point $u \in \mathcal{X}, u \neq v$, $v \in \mathcal{Q}^+(\ell^*)$, and $W_{opt} \leq W_{\mathcal{X}}(\mathcal{Q}^+(\ell^*)) \leq W_{opt} + W(v)$, where W_{opt} is the weight of an optimal separator set for set $\mathcal{X}$.*

Proof: (Lemma 3.3.9). It is enough to prove the lemma for the case when point is positive. Let ℓ be an optimal fat separator for $\mathcal{X}-v$ that misclassifies v. Since ℓ is fat, no point from $\mathcal{X}-v$ can reside on it. Since v is positive and v is misclassified by ℓ, v is either reside directly on ℓ or in the negative side of ℓ. Thus, line ℓ^* that is parallel to ℓ, has same orientation and passing through the closest point u in the positive side of ℓ also will misclassify v. Since ℓ is optimal, u have to be a positive point.
$W_{opt} \leq W_{\mathcal{X}}(\mathcal{Q}^+(\ell^*))$ by the definition of optimal separator, and thus all we have to prove is that $W_{\mathcal{X}}(\mathcal{Q}^+(\ell^*)) \leq W_{opt} + W(v)$. It is easy to see that $W_{\mathcal{X}-v}(\mathcal{Q}(\ell^*)) = W_{\mathcal{X}-v}(\mathcal{Q}(\ell))$ and $W_{\mathcal{X}}(\mathcal{Q}(\ell^*)) = W_{\mathcal{X}-v}(\mathcal{Q}(\ell^*)) + W(v)$. Since an optimal separator for $\mathcal{X}$ is also a separator for $\mathcal{X}-v$ with weight at most W_{opt}, and ℓ is optimal for $\mathcal{X}-v$, we have $W_{\mathcal{X}-v}(\mathcal{Q}(\ell)) \leq W_{opt}$, so $W_{\mathcal{X}}(\mathcal{Q}(\ell^*)) \leq W_{opt} + W(v)$. ■

Lemma 3.3.10 *Let v be positive (negative) point from $\mathcal{X}$ and let ℓ^* be positive (negative) separator line that passes through positive (negative) point $u \in \mathcal{X}, u \neq v$, $v \in \mathcal{Q}^+(\ell^*)$, and $W_{opt} \leq W_{\mathcal{X}}(\mathcal{Q}^+(\ell^*)) < W_{opt} + W(v)$. Then any optimal fat separator for $\mathcal{X}-v$ will misclassify v.*

Proof: (Lemma 3.3.10). First, note that $W_{\mathcal{X}-v}(\mathcal{Q}(\ell^*)) = W_{\mathcal{X}}(\mathcal{Q}(\ell^*)) - W(v) < W_{opt}$. Since ℓ^* is a separator line for $\mathcal{X}-v$ with weight strictly less than W_{opt}, weight of any optimal separator for $\mathcal{X}-v$ is also strictly less than W_{opt}. If we suppose that there exists optimal fat separator for $\mathcal{X}-v$ that classifies v correctly, then this separator also would be a separator for $\mathcal{X}$ with weight smaller than W_{opt}. ■

Algorithm 1 Algorithm to compute type of points.

1: //**Input:** Set of points $\mathcal{X}$, Optimal fat separator ℓ^* for $\mathcal{X}$
2: //**Output:** $Type(u)$, $\forall u \in \mathcal{X}$
3: LOO = 0.
4: Let $W_{opt} = W(\ell^*)$ be the weight of the optimal separator.
5: **for** any positive (negative) point $u \in \mathcal{X}$ **do**
6: Sort all other points around u accordingly to the angle coordinates.
7: Consider all the different positive (negative) representative lines that pass through u.
8: **for** every representative line ℓ with weight W_ℓ **do**
9: $\forall$ positive (negative) $v \in \mathcal{Q}(\ell)$, such that $W(v) > W_\ell - W_{opt}$, $SET\ Type(v) = 2$
10: $\forall$ positive (negative) $v \in \mathcal{Q}(\ell)$, such that $W(v) == W_\ell - W_{opt}$, $SET\ Type(v) = 3$
11: **end for**
12: **end for**

The two lemmas above immediately give us Algorithm 1 for computing the type of all the points. Please note that point once marked as of type 2, cannot change it anymore. Point once marked as of type 3, can only change its type for 2. All the points left unmarked after the Algorithm 1 stopped are of type 1.

For a fixed central point u, execution of the loop $8-10$ can be performed in time $O(NlogN)$ as follows: for every representative line ℓ, place two marks on the circle that bound the positive (negative) arc of this line, and specify the representative line weight in these marks. After this is done for all the representatives, mark type of vertices in single ordered scan. Every time passing opening mark of some representative line, insert this mark into ordered set of active bounds (bounds sorted accordingly to the line weights). Every time passing closing mark of a representative line, remove corresponding mark from the ordered set of active bounds. Every time passing a point, update the type of the vertex accordingly to the smallest active bound.

In the Algorithm 1 above we can also keep track of number of times every vertex of type 3 was labelled as of type 3. Adding such a counter $C(v)$ for every vertex $v \in \mathcal{X}$ won't increase the algorithm's time complexity. It is clear that at the end of processing, counter for a vertex v will contain number of different representative lines in $\mathcal{X} - v$ such that v belongs to the separator set defined by the representative line: $C(v) = N_c(v) + N_e(v)$. If vertex v belongs to the separator set defined by the representative line, the optimal fat separator corresponding to this representative might or might not misclassify v. Following lemma establishes useful property of optimal fat separators that we will use to make our algorithm efficient.

Lemma 3.3.11 *Let ℓ be a representative line and $W(\mathcal{Q}(\ell)) = W_{opt} + \alpha$. Suppose there are k vertices in $\mathcal{Q}(\ell)$ with weight α: $V_P = \{v_1, \ldots, v_k\}$. Then optimal fat separator ℓ_o corresponding to ℓ can classify correctly at most one vertex in V_P. If ℓ_o classifies correctly any vertex from $\mathcal{Q}(\ell) - V_P$, it misclassifies all the vertices from V_P.*

Proof: (Lemma 3.3.11) Suppose ℓ_o classifies correctly vertex $v \in \mathcal{Q}(\ell)$. Then $\forall v_i \in V_P$, $W(v) + W(v_i) > \alpha$. Thus, if ℓ_o also classifies correctly v_i, then $W_{\mathcal{X}}(\mathcal{Q}(\ell_o)) \leq W(\mathcal{Q}(\ell)) - (W(v) + W(v_i)) < W_{opt}$, that contradicts with the optimality of W_{opt}. ■

Suppose we are given a representative line ℓ, optimal fat separator ℓ_o corresponding to ℓ and

convex hull built on $\mathcal{Q}(\ell)$. Then accordingly to the lemma 3.3.11, there exists at most one point in $\mathcal{Q}(\ell)$ with weight $W(\mathcal{Q}(\ell)) - W_{opt}$ can be classified correctly. This point or non-existence of such a point can be established in time $O(logN)$. Thus, exact value of $N_c(v)$ for every point v can be computed in time $O(N^2 logN)$ as follows: for every central point, sort all the other points according to the angle coordinates. Start enumerating all the representative lines, for every representative ℓ compute the optimal fat separator ℓ_o and convex hull on $\mathcal{Q}(\ell)$, then increase the N_c counter for the point from $\mathcal{Q}(\ell)$ with weight $W(\mathcal{Q}(\ell)) - W_{opt}$ (if any) that classified correctly by ℓ_o.

The last thing remaining to finish our proof of the proposition 3.3.6 is to show how to update the convex hull on $\mathcal{Q}(\ell)$ for a new representative line ℓ from convex hull for the separator of a previous representative line $\mathcal{Q}(\ell_{prev})$ in time O(N log N) for all representatives with a fixed center. It can be done as follows. It is well known that adding a point to a convex hull can be done in $O(logN)$ time. The only difficulty is with deleting a point from convex hull. But since we process representative lines in ordered manner, we know the order in which points will be deleted from the convex hull for the first representative. We can build the convex hull for the first representative iteratively adding points in reversed order and store all the intermediate convex hulls (not explicitly, we only need to store the information that will enable us restore previous convex hull after deleting next point in time $O(logN)$). This observation finishes our proof of proposition 3.3.6.

3.4 Discussion and Open Questions

As an example application of the linear separators in computational finance, consider the problem of optimal boosting of two classifiers. We have training set $\{x_i\}_{i\in\mathbb{I}}$ of points in multidimensional space $\mathbb{R}^m$. Every datapoint $x_k \in \mathbb{R}^m$ corresponds to values of a set of monitored features at time t_k. We also given values $\{y_i\}_{i\in\mathbb{I}}$ - change in the stock price from t_i to $t_i + \Delta t$. Our task is to construct an intelligent agent able to predict values of stock in the future $t_{cur} + \Delta t$ given the value of monitored features at current moment t_{cur}. Suppose also that we designed and trained two different intelligent agents g_1 and g_2 that for every point x_k predict the direction and magnitude of the price change $g_1(x_k)$ and $g_2(x_k)$. We can consider mapping from $\{x_i\}_{i\in\mathbb{I}}$ into 2-dimensional space $\mathcal{F}(x_i) = (g_1(x_i), g_2(x_i))$. In the ideal case of perfect predictors, we would end up with two separable sets on the plane (see Figure 3.2).

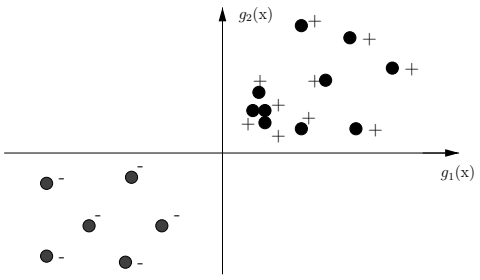


Figure 3.2: Case of two perfect predictors.

In reality, due to the noise in the data and models limitations we will obtain non-separable sets as depicted on Figure 3.3. Assuming that classifiers g_1 and g_2 are somewhat different, we want to construct a new classifier g_3 as a linear combination of g_1 and g_2 that combines strengths of both models. Then g_3 corresponds to an oriented line in the (g_1, g_2)- space. To every point from $\{x_i\}_{i\in\mathbb{I}}$ we can assign penalty for misclassification of this point $W(x_i)$. Penalties can be uniform ($W(x_i) = 1$, $\forall i \in \mathbb{I}$) or differentiated. One reasonable assignment of penalties is $W(x_i) = Prob(x_i|g_1(x_i), g_2(x_i))$, where $Prob(x_i|g_1(x_i), g_2(x_i))$ is the probability that observed value y_i is real given the values $g_1(x_i)$ and $g_2(x_i)$. We want g_3 to classify correctly as many points as possible and, futhermore, we want to pay more attention to the classification of points that are known to be more reliable. In other

words, we want to keep the summary likelihood of the misclassified points as small as possible. Then problem of finding the optimal linear combination g_3 is essentially the problem of finding the optimal separator set for a given set of weighted points in the 2-dimensional space. Earlier in this chapter we presented $O(n^2 logn)$ - algorithm for computing the optimal separator set for an arbitrary weighting function.

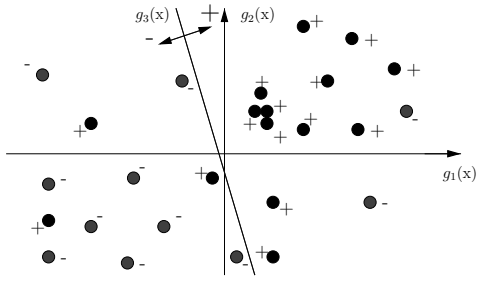


Figure 3.3: Optimal boosting of two classifiers.

Interesting open question is the probabilistic interpritation of the leave-one-out error in the case when points are weighted with the likelihood weighting function $Prob(x_i|g_1(x_i), g_2(x_i))$. Another open question is the constructing of similar algorithm for 3-dimenaional case with time complexity at most $O(n^2 logn)$. It is an interesting fact that for separable sets in 3 dimensions linear separator can be found with same time complexity as in 2 dimesions. The efficiency of the 3-D algorithm is based on properties of planar graphs [23]. It is open question if these properties can be utilized for constructing an efficient algorithm for finding the optimal separator set in 3-D case.

Chapter 4

Avoiding Overfitting: Isotonic and Unimodal Regression

4.1 Introduction and Basic Definitions

The problem of overfitting training data is well recognized in the machine learning community. Standard approach to deal with this threat is the early stopping of the training algorithm iterations. Important question is when the iterations should be stopped. Usually one monitores another error measure and stops iterations when the associated error starts growing. The associated error can be same error function measured on separate set of datapoints (validation set) or even completely different error function. Since the associated error measure is somewhat different from the primary, it does not necessary shows monotinical behavior but often appears as random fluctuations around unknown function. In order to pinpoint the exact location of the critical point, one can perform shape constrained optimization to fit function to the ibserved values of the associated error. Another application of the shape constrained regression arise in the pricing of financial instruments, for example the american put option [44].

Futher applications of isotonic regression can be found in [62, 61]. Isotonic and unimodal regression are both examples of nonparametric shape constrained regression. Such regressions are useful when prior knowledge about the shape but not the parametric form of a function are known. The importance of isotonic regression stems from the fact that it is often the case in statistical estimation or learning that one wishes to estimate a function that is known to be monotonically increasing (say), even though the data will not necessarily exhibit this behavior, on account of noise, [43, 67]. Examples include the probability of heart attack as a function of cholesterol level [43]; the "credit worthiness" as a function of income [67].

To illustrate, suppose that we would like to determine cholesterol level thresholds at which a heart attack becomes more prevalent, and we have a sequence of patients with cholesterol levels $c_1 < c_2 < \ldots < c_n$. Associated to each patient i, let x_i be the number of heart attacks they had within the following year, $x_i = 0, 1, 2, \ldots, K$ for some small value of K. The isotonic regression determines thresholds for the cholesterol levels that identify different severities for heart attack risk.

A futher example of application of isotonic regression is epidemiologic studies, where one may be interested in assessing the relationship between dose of a possibly toxic exposure and the probability of an adverse response [51]. In characterizing biologic and public health significance, and the need

for possible regulatory interventions, it is important to efficiently estimate dose response, allowing for flat regions in which increases in dose have no effect. In such applications, one can typically assume a priori that an adverse response does not occur less often as dose increases, adjusting for important confounding factors, such as age and race. It is well known that incorporating such monotonicity constraints can improve estimation efficiency and power to detect trends [62].

Isotonic regression in the L_p norm, $p > 0$, is defined as follows. Let $\mathbf{x} = [x_1, x_2, \ldots, x_n], x_i \in \mathbb{R}$, be given. The task is to construct a corresponding sequence $\mathbf{w} = [w_1 \leq w_2 \leq \ldots \leq w_n]$ so that $\mathcal{E}_p(\mathbf{w})$ is minimized for some given p, where

$$\mathcal{E}_p(\mathbf{w}) = \begin{cases} \frac{1}{n} \sum_{i=1}^{n} |x_i - w_i|^p & 1 \leq p < \infty, \\ \max_i |x_i - w_i| & p = \infty. \end{cases}$$

The regression is *unimodal* if $w_1 \leq w_2 \leq \cdots \leq w_i \geq w_{i+1} \geq \cdots \geq w_n$, for some i; x_i is denoted a *crossover* point. The prefix-isotonic regression problem is to construct the isotonic regression for all prefixes of $\mathbf{x}$. We study the cases $p = 1$ and $p = \infty$. The case $p = 1$ is sometimes denoted isotonic median regression. We will refer to $\mathcal{E}_1(\mathbf{w})$ or $\mathcal{E}_\infty(\mathbf{w})$ as the error of the regression when the context is clear. The efficiency of an algorithm is measured in terms of n.

4.2 Previous Work

L_2 isotonic regression can be performed efficiently in linear time using some variant of a Pooling Adjacent Violators (PAV) algorithm [1, 61, 62]. For L_1 isotonic regression, algorithms in the efficiency class $O(n \log n)$ are known. Some approaches to isotonic regression are given in [13, 57, 58, 62].

The L_1 and L_2 prefix-isotonic regression problems have been solved optimally in [72]. For L_2, the runtime is $O(n)$, which is clearly optimal, and for L_1 it is $O(n \log n)$, which, by a reduction from sorting, is optimal [72]. While $O(n \log n)$ is optimal for L_1 prefix-isotonic regression, it is not known whether the apparently simpler isotonic regression problem can be performed faster than $O(n \log n)$. We take a first step in this direction by obtaining a linear bound in terms of the size of the output (K).

Unimodal regression has been studied extensively in [72], where the author gives a linear time algorithm for the L_2 case, and an $O(n \log n)$ algorithm for the L_1 cases. This result was a significant improvement over the exponential and quadratic algorithms that existed prior to this work [27, 28, 50, 56].

A general PAV type algorithm, [72], relies on the ability to efficiently update a suitably defined "mean". Such an algorithm is easily applicable to the L_1 and L_2 cases, however, for $p > 2$, the "L_p-mean" is not conveniently updated. For the case $p = \infty$ it is not clear what this "mean" should be, and hence, the algorithm cannot be applied.

4.3 Contribution

We provide two output sensitive isotonic median regression algorithms and algorithms for L_∞ regression. More specifically,

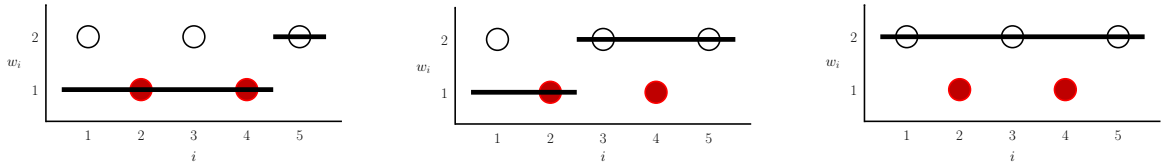
i. Suppose that $x_i \in \mathcal{X}$ where $|\mathcal{X}| = K$. Then, L_1-isotonic regression can be performed in $O(n \log K)$ time, linear in n. In the worst case, $K = n$ and we have $O(n \log n)$.

ii. Suppose that $x_i \in [a, b]$, $\forall i$. Given $\epsilon > 0$, we can construct an approximate L_1-isotonic regression with error at most the optimal plus ϵ in time $O(n \log(\frac{b-a}{\epsilon}))$.

iii. L_∞ prefix-isotonic and unimodal regressions can be constructed in linear time.

4.3.1 L_1-Isotonic Regression

We use the notation $[i, j]$ to refer to the interval of integers $\{i, i+1, \ldots, j\}$, and $\mathbf{x}[i, j]$ to represent the sequence $[x_i, \ldots, x_j]$. In this section, the isotonic regression will always refer to L_1-optimal isotonic regression. Without loss of generality, we can represent the isotonic regression by a collection of monotonically increasing *level sets*, or intervals to each of which is associated a value or *level*: $\mathcal{C} = \{I_\alpha, h_\alpha\}_{\alpha=1}^K$. Each I_α is an interval of the form $I_\alpha = [i_\alpha, j_\alpha]$. We assume that $i_1 = 1$, $j_K = n$, $i_{\alpha+1} = j_\alpha + 1$ and $h_\alpha < h_{\alpha+1}$ for $1 \leq \alpha < K$. The isotonic regression that is induced by $\mathcal{C}$ is given by assigning $w_i = h_\alpha$ for all $i \in I_\alpha$. We define the error for $\mathcal{C}$, $\mathcal{E}_1(\mathcal{C})$, as the error $\mathcal{E}_1(\mathbf{w})$ of the corresponding induced isotonic regression $\mathbf{w}$. Figure below below illustrates all this notation for the sequence $\mathbf{x} = [2, 1, 2, 1, 2]$.



$\mathcal{C} = \{([1, 4], 1), ([5, 5], 2)\}$
$\mathbf{w} = [1, 1, 1, 1, 2]$
$\mathcal{E}_1(\mathcal{C}) = 2$

$\mathcal{C} = \{([1, 2], 1), ([3, 5], 2)\}$
$\mathbf{w} = [1, 1, 2, 2, 2]$
$\mathcal{E}_1(\mathcal{C}) = 2$

$\mathcal{C} = \{([1, 5], 2)\}$
$\mathbf{w} = [2, 2, 2, 2, 2]$
$\mathcal{E}_1(\mathcal{C}) = 2$

Note that the isotonic regression is not unique. To remove this ambiguity, we will only consider the isotonic regression in which the sum of the w_i is minimized (the leftmost regression in the figure). We define the *weight* of the isotonic regression by $W(\mathcal{C}) = \sum_i w_i$, where $\{w_i\}$ is the isotonic regression induced by $\mathcal{C}$. Thus if $\mathcal{C}$ is an isotonic regression, and $\mathcal{C}'$ is any other monotonically increasing collection of level sets, then $\mathcal{E}_1(\mathcal{C}) \leq \mathcal{E}_1(\mathcal{C}')$, and if $\mathcal{E}_1(\mathcal{C}) = \mathcal{E}_1(\mathcal{C}')$, then $W(\mathcal{C}) < W(\mathcal{C}')$ (we will show later that the isotonic regression is indeed unique). Throughout, we will refer to the unique isotonic regression by $\mathcal{C} = \{I_\alpha = [i_\alpha, j_\alpha], h_\alpha\}_{\alpha=1}^K$, and in general, we will use I_α to refer both to the interval $[i_\alpha, j_\alpha]$, as well as to the set of points $\{x_{i_\alpha}, \ldots, x_{j_\alpha}\}$.

We define the median of a level set $I = [i, j]$, $M(I)$, to be the median of the points $\{x_i, x_{i+1}, \ldots, x_j\}$, where the median is defined in the usual way:

$$M(y_1 \leq y_2 \ldots \leq y_m) = y_{\lfloor \frac{m+1}{2} \rfloor}$$

Note that $M(I) = x_k$ for some $k \in [i, j]$. Further, note that if $M(S_1) \leq M(S_2)$ for any S_1, S_2, then $M(S_1) \leq M(S_1 \cup S_2) \leq M(S_2)$. It is also well known that the median is a minimizer of the L_1 error. Since we require the weight of the isotonic regression to be minimum, we conclude that the level of each level set has to be the median of the set:

Proposition 4.3.1 *$h_\alpha = M(I_\alpha)$ for all $\alpha \in [1, K]$.*

Proof: Suppose that $h_\alpha < M(I_\alpha)$ for some α. This means that there are strictly more points in I_α above h_α than below. By raising h_α we can decrease the error, contradicting the optimality of the isotonic regression. Suppose that $h_\alpha > M(I_\alpha)$ for some α. In this case, by the definition of the median, there are at least as many points below h_α as there are above. In this case, we guarantee not to increase the error by lowering h_α, and at the same time decrease the sum of the w_i contradicting the minimality of $W(\mathcal{C})$. ■

In particular, $h_\alpha = x_k$ for some $k \in [i_\alpha, j_\alpha]$, i.e., every level is one of the x_i's. Note that since, $h_\alpha < h_{\alpha+1}$, we immediately have that the sequence of medians must be increasing.

Corollary 4.3.2 *$M(I_\alpha) < M(I_\beta)$ for $1 \leq \alpha < \beta \leq K$.*

The next proposition is one of the crucial properties that we will use. It essentially states that the isotonic regression for a set of points is the union of the isotonic regressions for two disjoint subsets of the points. Consider *any* level set I_α in the isotonic regression and define the left and right subsets of the points with respect to this level set by $S_l = \{x_1, \ldots, x_{i_\alpha - 1}\}$ and $S_r = \{x_{i_\alpha}, \ldots, x_n\}$. We define the left and right isotonic regressions $\mathcal{C}_l$ and $\mathcal{C}_r$ as the isotonic regressions for the respective left and right subsets. Then $\mathcal{C} = \mathcal{C}_l \cup \mathcal{C}_r$. We will need the following lemma to prove the proposition,

Lemma 4.3.3 *For any α, with $I_\alpha = [i_\alpha, j_\alpha] \in \mathcal{C}$,*

$$\begin{array}{ll} (i) & M(\{x_{i_\alpha}, \ldots, x_j\}) \geq M(I_\alpha), \text{ for all } j \geq i_\alpha. \\ (ii) & M(\{x_i, \ldots, x_{j_\alpha}\}) \leq M(I_\alpha), \text{ for all } i \leq j_\alpha. \end{array}$$

Proof: *(i)* Let I_α be the last level set for which there exists a $j \geq i_\alpha$, with $M(\{x_{i_\alpha}, \ldots, x_j\}) < M(I_\alpha)$. Suppose $j > j_\alpha$. Then, $M(x_{i_{\alpha+1}}, \ldots, x_j) < M(I_\alpha) < M(I_{\alpha+1})$ and so I_α is not the last level set with this property. Thus, $j < j_\alpha$. Decompose (I_α, h_α) into two level sets: $(I_1 = \{x_{i_\alpha}, \ldots, x_j\}, \max(h_{\alpha-1}, M(I_1)))$ and $(I_2 = \{x_{j+1}, \ldots, x_{j_\alpha}\}, h_\alpha)$. The decomposition guarantees not to increase the error, while lowering the weight of the regression, contradicting the fact that $\mathcal{C}$ has minimum weight among optimal isotonic regressions.
(ii) Let I_α be the first level set for which there exists an $i \leq j_\alpha$, with $M(\{x_i, \ldots, x_{j_\alpha}\}) > M(I_\alpha)$. Suppose $i < i_\alpha$. Then, $M(x_i, \ldots, x_{j_{\alpha-1}}) > M(I_\alpha) > M(I_{\alpha-1})$ and so I_α is not the first level set with this property. Thus, $i > i_\alpha$. Decompose (I_α, h_α) into two level sets: $(I_1 = \{x_{i_\alpha}, \ldots, x_{i-1}\}, h_\alpha)$. and $(I_2 = \{x_i, \ldots, x_{j_\alpha}\}, \min(h_{\alpha+1}, M(I_2)))$. The decomposition strictly decreases the error, contradicting the fact that $\mathcal{C}$ has minimum error.

■

Proposition 4.3.4 *$\mathcal{C} = \mathcal{C}_l \cup \mathcal{C}_r$*

Note that the proposition is valid for any level set I_α that is used to construct S_l, S_r.

Proof: Let $\mathcal{C}' = \mathcal{C}_l \cup C_r = \{I'_\beta, h'_\beta\}_{\beta=1}^{K'}$. Since $h'_\beta = M(I'_\beta)$, it will suffice to show that $I'_\alpha = I_\alpha$ for all $\alpha \in [1, K]$. Suppose to the contrary and let α^* be the first level set for which $I_{\alpha^*} \neq I'_{\alpha^*}$. Further, suppose without loss of generality that $|I_{\alpha^*}| > |I'_{\alpha^*}|$ (a similar argument holds for $|I_{\alpha^*}| < |I'_{\alpha^*}|$). Therefore,

$$I_{\alpha^*} = I'_{\alpha^*} \cup I'_{\alpha^*+1} \cup \cdots \cup I'_{\alpha^*+L} \cup P,$$

where P is a prefix of I'_{α^*+L+1}. Note that $I_{\alpha^*}, \ldots, I'_{\alpha^*+L+1}$ are either all in $\mathcal{C}_l$ or all in $\mathcal{C}_r$. Without loss of generality, assume they are all in $\mathcal{C}_l$. We know that $h'_{\alpha^*+i} = M(I'_{\alpha^*+i})$ for $i \in [0, L+1]$ and by construction, $h'_{\alpha^*+i} < h'_{\alpha^*+i+1}$ for $i \in [0, L]$. From Lemma 4.3.3, we know that $M(P) \geq M(I'_{\alpha^*+L+1})$ (since $\mathcal{C}_l$ is the isotonic regression for S_l). By Lemma 4.3.3, we also have that $M(I_{\alpha^*}) \geq M(I'_{\alpha^*})$, and similarly from the optimality of $\mathcal{C}$, we have that $M(I'_{\alpha^*}) \geq M(I_{\alpha^*})$, hence that $M(I'_{\alpha^*}) = M(I_{\alpha^*})$. Therefore, we have that

$$M(I_{\alpha^*}) = M(I'_{\alpha^*}) < M(I'_{\alpha^*+1}) < \cdots < M(I'_{\alpha^*+L}) < M(I'_{\alpha^*+L+1}) \leq M(P).$$

Since P is a suffix of I_{α^*}, by the optimality of $\mathcal{C}$ and Lemma 4.3.3, we have that $M(P) \leq M(I_{\alpha^*})$ which is the desired contradiction. ■

An immediate consequence of this proposition is that the isotonic regression is unique, by choosing (for example) $S_l = \mathbf{x}$ and $S_r = \{\}$.

Corollary 4.3.5 *The isotonic regression is unique.*

Suppose we are given a constant γ, we would like to find the first level set whose height is at least γ. In particular, we would like to find the first point of this level set. We call this point a pivot point for γ. More specifically, let $\mathcal{C}$ be the isotonic regression, and let α be such that $h_\alpha \geq \gamma$ and if $\alpha > 1$, then $h_{\alpha-1} < \gamma$. We would like to find x_{i_α}. Note that if all the levels are $< \gamma$, then x_{i_α} does not exist, in which case we can default to $i_\alpha = n + 1$. We know from Lemma 4.3.3 that it is necessary for x_{i_α} to satisfy two conditions:

i. for every sequence S begining at x_{i_α}, $M(S) \geq h_\alpha \geq \gamma$;

ii. for every sequence S' ending at $x_{i_\alpha - 1}$, $M(S') \leq h_{\alpha-1} < \gamma$.

The content of the next proposition is that these conditions are also sufficient.

Theorem 4.3.6 *Let $\mathcal{C}$ be the isotonic regression. Given γ, let I_α be the first level set with $h_\alpha \geq \gamma$. Then, x_i is the first point in I_α (i.e., $x_i = x_{i_\alpha}$) if and only if for any sequence S begining at x_i and any sequence S' ending at x_{i-1}, $M(S') < \gamma \leq M(S)$.*

Proof: It only remains to prove that if $M(S') < \gamma \leq M(S)$ for any two sequences as described, then $i = i_\alpha$. We know that i must belong to one of the level sets, $i \in I_\beta$ for some β with $1 \leq \beta \leq K$. We need to show three things: (i) $h_\beta \geq \gamma$; (ii) $i = i_\beta$; (iii) $h_{\beta-1} < \gamma$.
(i) Suppose that $h_\beta < \gamma$. Then, consider $S = \{x_i, \ldots, x_{j_\beta}\}$. By Lemma 4.3.3, $M(S) \leq h_\beta < \gamma$. By construction of x_i, $M(S) \geq \gamma$, a contradiction.
(ii) Suppose that i is not the first point in I_β. Then consider $S' = \{x_{i_\beta}, \ldots, x_{i-1}\}$. By Lemma 4.3.3, $M(S') \geq h_\beta \geq \gamma$ (by (i)). By construction of x_i, $M(S') < \gamma$, a contradiction.
(iii) Suppose that $h_{\beta-1} \geq \gamma$. Consider $S' = \{x_{i_{\beta-1}}, \ldots, x_{i-1}\}$. From (ii), this is exactly $I_{\beta-1}$. By construction of x_i, $M(S') = M(I_{\beta-1}) = h_{\beta-1} < \gamma$, a contradiction.

■

Thus to find the first point of the first level set with height at least a given γ, we only need to search for an x_i that satisfies the conditions of Theorem 4.3.6. The remainder of this section is devoted to developing a linear time algorithm to find this point. This algorithm will be the basis of our isotonic regression algorithms that we discuss in the next section.

Define the following three quantities for any interval $[i, j]$.

$$\begin{aligned} N^+(i,j): &\quad \text{the number of points} \geq \gamma \text{ in the set } S_{[i,j]} = \{x_i, \ldots, x_j\}. \\ N^-(i,j): &\quad \text{the number of points} < \gamma \text{ in the set } S_{[i,j]} = \{x_i, \ldots, x_j\}. \\ \Delta_r(i,j): &\quad \min_{t \in [i,j]} (N^+(i,t) - N^-(i,t)). \\ \Delta_l(i,j): &\quad \max_{t \in [i,j]} (N^+(t,j) - N^-(t,j)). \end{aligned}$$

Note that the median of the set $S_{[i,j]}$ is $\geq \gamma$ if and only if $N^+(i,j) - N^-(i,j) > 0$. From this observation, we get the following lemma.

Lemma 4.3.7 *x_k satisfies the conditions of Theorem 4.3.6 if and only if one of the following hold:*

i. $k = 1$ *and* $\Delta_r(k, n) > 0$*;*

ii. $k > 1$, $\Delta_r(k, n) > 0$ *and* $\Delta_l(1, k-1) \leq 0$.

If no such x_k exists, then the levels of all the level sets are $< \gamma$.

We show how to find such an x_i in *linear* time. Start two pointers $p_l = 0$ and $p_r = n + 1$. The initial conditions of the algorithm are:

$$\begin{aligned} N^+(p_r, n) = 0; &\quad N^-(p_r, n) = 0, \\ N^+(1, p_l) = 0; &\quad N^-(1, p_l) = 0. \end{aligned}$$

Let $\mathbf{x}_l = \mathbf{x}[1, p_l]$, $\mathbf{x}_r = \mathbf{x}[p_r, n]$, and $S = \mathbf{x}[p_l + 1, p_r - 1]$. Initially, $\mathbf{x}_r = \mathbf{x}_l = \{\}$, and $S = \mathbf{x}$. If $M(S) \geq \gamma$, then we know that x_{p_r} is not our solution, so we decrement p_r by 1 and update $\mathbf{x}_l, \mathbf{x}_r, S$. On the other, if $M(S) < \gamma$, then x_{p_l+1} is not our solution, so we increment p_l by 1 and update $\mathbf{x}_l, \mathbf{x}_r, S$. We continue this process of decreasing p_r or increasing p_l until $p_r = p_l + 1$. We now prove that this algorithm correctly computes the pivot point. The nature of the algorithm is to move p_r (resp. p_l) until $M(S)$ switches from $\geq \gamma$ (resp. $< \gamma$) to $< \gamma$ (resp. $\geq \gamma$). Denote a phase in the algorithm as the period when one of the pointers begins to move and then stops.

Lemma 4.3.8 *The following invariants are maintained at the end of every phase.*

i. The median of every prefix of $\mathbf{x}_r$ is $\geq \gamma$.

ii. The median of every suffix of $\mathbf{x}_l$ is $< \gamma$.

Algorithm 2 Algorithm to compute a pivot point.

1: //**Input:** $\mathbf{x} = \{x_i | i \in [1, n]\}$ and $\gamma \in \mathbb{R}$.
2: //**Output:** i such that x_i is the pivot point for $\geq \gamma$.
3: Set $p_l = 0$, $p_r = n + 1$ and using a single scan compute $N^{\pm}(p_l + 1, p_r - 1)$;
4: **while** $p_l + 1 \neq p_r$ **do**
5: **if** $N^+(p_l + 1, p_r - 1) - N^-(p_l + 1, p_r - 1) > 0$ **then**
6: $p_r \leftarrow p_r - 1$, and update $N^{\pm}(p_l + 1, p_r - 1)$;
7: **else**
8: $p_l \leftarrow p_l + 1$, and update $N^{\pm}(p_l + 1, p_r - 1)$;
9: **end if**
10: **end while**
11: **return** p_r;{$p_r = n + 1$ if all levels are $< \gamma$.}

Proof: We prove the claim by induction on the phase number. Initially the invariants hold by default since $\mathbf{x}_r$ and $\mathbf{x}_l$ are empty. Suppose the invariants hold up to some phase, and consider the next phase, i.e., $p_l + 1 < p_r$.
Suppose that $p_l \rightarrow p'_l$ and $\mathbf{x}_l \rightarrow \mathbf{x}'_l$ in this phase. By construction, $M(\mathbf{x}[k, p_r - 1]) < \gamma$ for $p_l + 1 \leq k \leq p'_l$. Since p_l stopped moving, there are two cases. (i) $p'_l = p_r - 1$, in which case the median of every suffix of $\mathbf{x}[p_l + 1, p'_l]$ is $< \gamma$. (ii) $p'_l < p_r - 1$, in which case $M(\mathbf{x}[p'_l + 1, p_r - 1]) \geq \gamma$. But since $M(\mathbf{x}[k, p_r - 1]) < \gamma$ for $p_l + 1 \leq k \leq p'_l$, it follows that $M(\mathbf{x}[k, p'_l]) < \gamma$, or once again, the median of every suffix of $\mathbf{x}[p_l + 1, p'_l]$ is $< \gamma$. Every suffix of $\mathbf{x}'_l$ is either a suffix of $\mathbf{x}[p_l + 1, p'_l]$ or the union of $\mathbf{x}[p_l + 1, p'_l]$ with a suffix of $\mathbf{x}_l$. Since $M(S_1) < \gamma$ and $M(S_2) < \gamma$ implies $M(S_1 \cup S_2) < \gamma$ for any S_1, S_2, invariant (ii) now follows, i.e., the median of every suffix of $\mathbf{x}'_l$ is $< \gamma$. Since p_r did not move in this phase, invariant (i) was unchanged.
Similarily, suppose instead that $p_r \rightarrow p'_r$ and $\mathbf{x}_r \rightarrow \mathbf{x}'_r$ in this phase. This means that $M(\mathbf{x}[p_l + 1, k])ge\gamma$ for $p'_r \leq k \leq p_r - 1$. Once again, there are two cases, $p'_r = p_l + 1$ and $p'_r > p_l + 1$. In both cases it follows using similar arguments that the median of every prefix of $\mathbf{x}[p'_r, p_r - 1]$ is $\geq \gamma$. Invariant (i) follows from the facts that any prefix of $\mathbf{x}'_r$ is the union of prefixes of $\mathbf{x}[p'_r, p_r - 1]$ and $\mathbf{x}_r$, and $M(S_1) \geq \gamma,\ M(S_2) \geq \gamma \implies M(S_1 \cup S_2) \geq \gamma$. Since p_l did not move in this phase, invariant (ii) was unchanged. ■

Thus when the algorithm concludes, $\Delta_r(p_r, n) > 0$ and $\Delta_l(p_1, l) \leq 0$ and we have the pivot point. The efficiency of the algorithm hinges on being able to determine if $M(S)$ is larger or smaller than γ. Since $M(\mathbf{x}[i, j]) \geq \gamma$ if and only if $N^+(i, j) - N^-(i, j) > 0$, we need to maintain $N^{\pm}(p_l + 1, p_r - 1)$. The following update rules allow us to do this efficiently. Suppose we have computed $N^{\pm}(i, j)$ for $1 \leq i < j \leq n$

$$\begin{array}{lll} N^+(i+1,j) = N^+(i,j) - 1; & N^-(i+1,j) = N^-(i,j) & \text{if } x_i \geq \gamma. \\ N^+(i+1,j) = N^+(i,j); & N^-(i+1,j) = N^-(i,j) - 1 & \text{if } x_i < \gamma. \end{array}$$

$$\begin{array}{lll} N^+(i,j-1) = N^+(i,j) - 1; & N^-(i,j-1) = N^-(i,j) & \text{if } x_j \geq \gamma. \\ N^+(i,j-1) = N^+(i,j); & N^-(i,j-1) = N^-(i,j) - 1 & \text{if } x_j < \gamma. \end{array}$$

The entire algorithm is summarised in Algorithm 2.

We define an operation as a comparison, a floating point operation or an assignment. Step 3 can be computed in $3n$ operations. An update (steps 6,8) takes 6 operations, and n updates need to be made. We thus have the following theorem.

Theorem 4.3.9 *Given $\mathbf{x} = \{x_i | i \in [1, n]\}$ and $\gamma \in \mathbb{R}$, the pivot point for γ can be found using at most Cn operations, where $C \approx 9$.*

Summary. The pivot point x_i for any value γ can be found in linear time. $\mathbf{x}$ can then be partitioned into two disjoint subsets, $\mathbf{x}_l = \mathbf{x}[1, i-1]$ and $\mathbf{x}_r = \mathbf{x}[i, n]$. The isotonic regression $\mathcal{C}_l$ of $\mathbf{x}_l$ will have level sets all of whose levels are $< \gamma$, and the isotonic regression $\mathcal{C}_r$ of $\mathbf{x}_r$ will have level sets all of whose levels are $\geq \gamma$. Further, the isotonic regression $\mathcal{C}$ of $\mathbf{x}$ is given by $\mathcal{C} = \mathcal{C}_l \cup \mathcal{C}_r$. This result already has applications. Suppose we would simply determine a threshold x where the response function exceeds a given value, γ. This can be accomplished by finding the pivot point for γ.

4.3.2 L_1-Isotonic Regression: Algorithms

The importance of Proposition 4.3.4 and Theorem 4.3.9 from the algorithmic point of view can be summarised as follows. Suppose we have the input $\mathbf{x}$ for which the isotonic regression can only have levels in the set $\{m_1 < m_2 < \cdots < m_K\}$ – for example, this would be the case if x_i can only take values in this set. Let p be the index of the pivot point for $\gamma = m_i$, $i \in [1, K]$. This pivot point, which can be found in linear time, partitions $\mathbf{x}$ into $\mathbf{x}_l = \mathbf{x}[1, p-1]$ and $\mathbf{x}_r = \mathbf{x}[p, n]$ (one of these may be empty). By Proposition 4.3.4, it then suffices to recursively compute the isotonic regressions for $\mathbf{x}_l$ and $\mathbf{x}_r$. Further, by construction of p, all the levels in $\mathbf{x}_l$ will be $< \gamma = m_i$, and all the levels in $\mathbf{x}_r$ will be $\geq \gamma$. We obtain an efficient algorithm by choosing γ to be the median of the available levels each time in the recursion. The full algorithm is given in Algorithm 3.

The correctness of this algorithm follows from the results in the previous section, specifically Proposition 4.3.4. What remains is to analyse the run time. It is enough to analyse the runtime of ISOTONIC$(\mathbf{x}, \mathbf{m}, [i, j], [k, l])$. Let $T(n, K)$ be the worst case runtime when $|[i, j]| = n$ and $|[k, l]| = K$. Then in the worst case, the algorithm will call itself on a left set of size δ with $\lceil K/2 \rceil$ levels and on a right set of size $n - \delta$ with $\lfloor K/2 \rfloor$ levels, for some $0 \leq \delta \leq n$. As already discussed, the pivot step to perform this partition takes at most Cn operations (step 9), so we have the following recursion for $T(n, K)$:

$$T(n, K) \leq \max_{\delta \in [0,n]} \left(T(\delta, \lceil \tfrac{K}{2} \rceil) + T(n - \delta, \lfloor \tfrac{K}{2} \rfloor) \right) + Cn.$$

For $K = 2^l$, a straight forward induction shows that $T(n, K) \leq Cn \log K$. By monotonicity, $T(n, K) \leq T(n, 2^{\lceil \log K \rceil})$, which gives $T(n, K) \leq Cn \lceil \log K \rceil$, yielding the following theorem.

Theorem 4.3.10 *The isotonic regression for n points with K possible levels can be obtained in $O(n \log K)$ time.*

If the K levels are not known ahead of time, they can be determined and sorted using standard data structures, such as a balanced binary seach tree in $O(n \log K)$ time, [17]. This does not affect the asymptotic running time. In the worst case, $K = n$ and our algorithm is no worse than existing algorithms. However, there can be significant improvement in the efficiency when K is fixed and small.

Algorithm 3 Algorithm to perform the full isotonic regression.

1: // **Wrapper to call the recursive function.**
2: //**Input:** $\mathbf{x} = \{x_i | i \in [1, n]\}$ and $\mathbf{m} = \{m_1 < m_2 < \cdots < m_K\}$.
3: //**Output:** Isotonic regression, $\mathcal{C} = \{(I_\alpha, h_\alpha)\}$
4: Call ISOTONIC($\mathbf{x}, \mathbf{m}, [1, n], [1, K]$);

1: **ISOTONIC**($\mathbf{x}, \mathbf{m}, [i, j], [k, l]$)
2: //**Output:** Isotonic regression $\mathcal{C} = \{(I_\alpha, h_\alpha)\}$ for $\mathbf{x}[i, j]$, given all levels are in $\mathbf{m}[k, l]$.
3: **if** $j < i$ **then**
4: **return** $\{\}$;
5: **else if** $k = l$ **then**
6: **return** $\{([i, j], \mathbf{m}[k])\}$
7: **else**
8: Let $q = k + 1 + \left\lfloor \frac{l-k}{2} \right\rfloor$; {$q$ is 1+the median of $[k, l]$}
9: Let p=index of pivot point for $\mathbf{x}[i, j]$ with $\gamma = \mathbf{m}[q]$;
10: $\mathcal{C}_l$ =ISOTONIC($\mathbf{x}, \mathbf{m}, [i, p-1], [k, q-1]$); $\mathcal{C}_r$ =ISOTONIC($\mathbf{x}, \mathbf{m}, [p, j], [q, l]$);
11: **return** $\mathcal{C}_l \cup \mathcal{C}_r$;
12: **end if**

Approximate isotonic regression. The algorithm that we have given can be run with any set of levels supplied – the pivot point is defined for any γ. It is not required that the true isotonic regression levels all be from this set in order to run the algorithm. Ofcourse, if the true levels are not from the set of levels supplied to the algorithm, then the result cannot be the true isotonic regression. If the levels chosen are close to the true levels, then the approximate isotonic regression should be close to the true one.

In particular, suppose that $a \leq x_i \leq b$ for all $i \in [1, n]$. Consider the levels $m_i = a + i\epsilon$, where $\epsilon = (b - a)/K$ and $i \in [0, K]$. Suppose that $[i_\alpha, j_\alpha], h_\alpha$ is a (non-empty) level set output by the algorithm, $h_\alpha = a + i_\alpha \epsilon$. Then x_{i_α} is a pivot point, for which all the levels of the *true* isotonic regression to the right are $\geq h_\alpha$. Further, all the levels to the left of the next level set that is output are $< h_\alpha + \epsilon$. Therefore, the error of a point from its corresponding level output by the algorithm differs from its error with respect to the true isotonic regression level by at most ϵ. Thus, the additional error contributed by every point is at most ϵ, for a total error increase of at most $n\epsilon$, increasing $\mathcal{E}_1$ by at most ϵ. Further, the runtime is $O(n \log K) = O(n \log((b - a)/\epsilon))$, establishing the following theorem.

Corollary 4.3.11 *Suppose that $a \leq x_i \leq b$ for $i \in [1, n]$ and let $\mathbf{w}$ be the isotonic regression. Then, an approximate isotonic regression $\mathbf{w}'$ can be computed in $O(n \log((b - a)/\epsilon))$ time with $\mathcal{E}_1(\mathbf{w}') - \mathcal{E}_1(\mathbf{w}) \leq \epsilon$.*

4.3.3 L_∞-Prefix-Isotonic Regression

In this section, we will refer to the L_∞-optimal isotonic regression more simply as the isotonic regression (which is not necessarily unique). For any sequence of points $\mathbf{x} = [x_1, x_2, ..., x_n]$, define a *Maximally Violating Pair (MVP)* to be a pair of points that maximally violates the monotonicity requirement, i.e., an MVP is a pair (x_l, x_r) with $l < r$, $x_l > x_r$, and $\forall i < j$, $x_l - x_r \geq x_i - x_j$. If

$x_i \leq x_j$ for all $i < j$, then no such pair exists. If $\mathbf{x}$ has an MVP (x_l, x_r), we define the *distortion* of $\mathbf{x}$, $D(\mathbf{x})$, to be $(x_l - x_r)$, and $D(\mathbf{x}) = 0$ if $\mathbf{x}$ does not have an MVP. Note that by definition of an MVP, $x_i \leq x_l$ for all $i < r$ and $x_j \geq x_r$ for all $j > l$.

Let $\mathcal{C}$ be an isotonic regression for $\mathbf{x}$ and let (x_l, x_r) be an MVP. Either $w_l \leq (x_l + x_r)/2$ or $w_r \geq w_l > (x_l + x_r)/2$, so we conclude that $\mathcal{E}_\infty(\mathcal{C})$ cannot be less that $D(\mathbf{x})/2$. The next proposition shows that this lower bound is achievable.

Proposition 4.3.12 *Let $\mathcal{C}$ be an isotonic regression for $\mathbf{x}$. Then $\mathcal{E}_\infty(\mathcal{C}) = D(\mathbf{x})/2$. Further, if (x_l, x_r) is an MVP, then $w_l = w_r = (x_l + x_r)/2$.*

Proof: If $D(\mathbf{x}) = 0$, then $\mathbf{x}$ is a monotonically nondecreasing sequence. $w_i = x_i$ is the optimal regression with $\mathcal{E}_\infty = 0$. Suppose that $D(\mathbf{x}) > 0$. We will construct (by induction) an isotonic regression with error $D(\mathbf{x})/2$. It then follows immediately that $w_l \geq (x_l + x_r)/2 \geq w_r$, and by monotonicity, $w_r \geq w_l$ from which we get $w_l = w_r = (x_l + x_r)/2$.
The induction basis is when $\mathbf{x} = \{\}$, $\mathbf{x} = [x_1]$ or $\mathbf{x} = [x_1, x_2]$, in which cases the claim is obvious. Suppose that an optimal regression exists with error $D(\mathbf{x})/2$ whenever $|\mathbf{x}| \leq N$, and consider any sequence $\mathbf{x}$ with $|\mathbf{x}| = N + 1$ and $D(\mathbf{x}) > 0$. Let (x_l, x_r) be an MVP, and define the left and right sequences: $\mathbf{x}_l = [x_1, x_2, \ldots, x_{l-1}]$; and $\mathbf{x}_r = [x_{r+1}, x_{r+2}, \ldots, x_{N+1}]$. Note that $D(\mathbf{x}_l) \leq D(\mathbf{x})$ and $D(\mathbf{x}_r) \leq D(\mathbf{x})$. Let $\mathcal{C}_l$ and $\mathcal{C}_r$ be the isotonic regressions for $\mathbf{x}_l$ and $\mathbf{x}_r$ respectively. Since the left and right sequences are strictly shorter than $\mathbf{x}$, by the induction hypothesis, we have that $\mathcal{E}_\infty(\mathcal{C}_l) = D(\mathbf{x}_l)/2 \leq D(\mathbf{x})/2$ and $\mathcal{E}_\infty(\mathcal{C}_r) = D(\mathbf{x}_r)/2 \leq D(\mathbf{x})/2$.
We now show how to construct the isotonic regression for $\mathbf{x}$ with error $D(\mathbf{x})/2$ from $\mathcal{C}_l$, $\mathcal{C}_r$ and one additional level set $\mathcal{C}^* = \{(I = [l, r], h = (x_l + x_r)/2)\}$. Consider all level sets in $\mathcal{C}_l$ with level $\geq h$. Reduce *all* these levels to h, and call this new isotonic regression $\mathcal{C}'_l$. We claim that $\mathcal{E}_\infty(\mathcal{C}'_l) \leq D(\mathbf{x})/2$. We only need to consider the level sets whose levels were altered. Let x be any point in such a level set with height $h' \geq h$. $x \leq x_l$ by definition of the MVP (x_l, x_r). $x \geq x_r$, because if $x < x_r$, then $D(\mathbf{x}_l)/2 \geq h' - x > h - x_r = D(\mathbf{x})/2 \geq D(\mathbf{x}_l)/2$, which is a contradiction. Thus $x_r \leq x \leq x_l$ and so the error for any such point is at most $D(\mathbf{x})/2$ for the regression $\mathcal{C}'_l$. The error for all other points has remained unchanged and was originally at most $\mathcal{E}_\infty(\mathcal{C}_l) = D(\mathbf{x}_l)/2 \leq D(\mathbf{x})/2$, so we conclude that $\mathcal{E}_\infty(C'_l) \leq D(\mathbf{x})/2$. Similarily, consider all level sets of $\mathcal{C}_r$ with level $\leq h$. Increase *all* these levels to h and call this new isotonic regression $\mathcal{C}'_r$. Once again any point x in any level set with a level change must satisfy $x_r \leq x \leq x_l$ and so we conclude that $\mathcal{E}_\infty(\mathcal{C}'_r) \leq D(\mathbf{x})/2$.
Consider the regression $\mathcal{C}' = \mathcal{C}'_l \cup \mathcal{C}^* \cup \mathcal{C}'_r$. $\mathcal{E}_\infty(\mathcal{C}') = \max\{\mathcal{E}_\infty(\mathcal{C}'_l), \mathcal{E}_\infty(\mathcal{C}^*), \mathcal{E}_\infty(\mathcal{C}'_r)\} = D(\mathbf{x})/2$. The isotonic regression $\mathcal{C}$ is constructed from $\mathcal{C}'$ by taking the union of all level sets with the height h (these must be consecutive level sets), which does not alter the error. ■

Proposition 4.3.12 immediately yields a recursive algorithm to compute the isotonic regression. Unfortunately, this recursive algorithm would have a run time that is quadratic in n. We now show how to construct this regression from left to right, using a single pass. This will lead to a linear time algorithm for the prefix-isotonic regression problem. Let $\mathbf{x}_i = \mathbf{x}[1, i]$. Let $\mathcal{C}_i$ be an isotonic regression for $\mathbf{x}_i$. The prefix-isotonic regression is given by $\{\mathcal{C}_i\}_{i=1}^n$. Note that $\mathcal{E}_\infty(\mathcal{C}_{i+1}) \geq \mathcal{E}_\infty(\mathcal{C}_i)$ since $D(\mathbf{x}_{i+1}) \geq D(\mathbf{x}_i)$. We will construct $\mathcal{C}_{i+1}$ from $\mathcal{C}_i$.

Let $\mathcal{C}_i = \{I_\alpha = [i_\alpha, j_\alpha], h_\alpha\}_{\alpha=1}^K$. Let $\inf_\alpha = \min_{k \in I_\alpha} x_k$, and $\sup_\alpha = \max_{k \in I_\alpha} x_k$. Define the distortion of level set I_α, $D(I_\alpha)$ as the distortion of the sequence $\mathbf{x}[i_\alpha, j_\alpha]$. The $\mathcal{C}_i$ that we construct will all satisfy the following properties:

P1: $\forall \alpha \in [1, K]$, $h_\alpha = \frac{1}{2}(\sup_\alpha + \inf_\alpha)$.

P2: $\forall \alpha \in [1, K]$, $D(I_\alpha) = \sup_\alpha - \inf_\alpha$.

P3: $\forall \alpha \in [2, K]$, $h_{\alpha-1} < h_\alpha$.

Property $P3$ is just a restatement of the monotonicity condition. From property $P2$ it follows that for any $i \in I_\alpha$, $|x_i - h_\alpha| \leq D(I_\alpha)/2$. Since $D(I_\alpha) \leq D(\mathbf{x})$, it follows from Proposition 4.3.12 that any regression that has properties $P2$ and $P3$ is necessarily optimal. Therefore, properties $P1$-$P3$ are sufficient conditions for an isotonic regression. Suppose that $\mathcal{C}_i$ has been constructed, satisfying $P1$-$P3$. Now consider adding the point x_{i+1}. Let $I_{K+1} = \{i+1\}$, $h_{K+1} = x_{i+1}$. Note that $D(I_{K+1}) = 0$, and by construction, I_{K+1} satisfies $P1$ and $P2$.

Lemma 4.3.13 *If $h_{K+1} > h_K$, let $\mathcal{C}_{i+1} = \mathcal{C}_i \cup \{(I_{K+1}, h_{K+1})\}$. Then $\mathcal{C}_{i+1}$ satisfies $P1$-$P3$.*

If $h_{K+1} \leq h_K$, then to get $\mathcal{C}_{i+1}$, we merge I_{K+1} with I_K. We need to ensure that properties $P1$ and $P2$ continue to hold. We will prove this in general for any two consecutive level sets. Suppose that (I_k, h_k) and (I_{k+1}, h_{k+1}) both satisfy properties $P1$ and $P2$, and suppose that $h_{k+1} \leq h_k$. Define the new level set I'_k by

$$I'_k = I_k \cup I_{k+1} \qquad \inf'_k = \min(\inf_k, \inf_{k+1}) \qquad \sup'_k = \max(\sup_k, \sup_k)$$
$$h'_k = \tfrac{1}{2}(\inf'_k + \sup'_k)$$

Lemma 4.3.14 *I'_k satisfies properties $P1$ and $P2$.*

Proof: By construction, $P1$ is satisfied. We show that $D(I'_k) = \sup'_k - \inf'_k$, from which $P2$ follows.
Suppose that $\inf_{k+1} \leq \inf_k$. Thus, $\inf'_k = \inf_{k+1}$. Since the first maximum in I_{k+1} occurs before the last minimum in I_{k+1} (as I_{k+1} satisfies $P2$), and the maximum in I_k occurs before any point in I_{k+1}, it follows that the first maximum in I'_k occurs before its last minimum, thus I'_k satisfies $P2$.
Suppose, on the other hand, that $\inf_{k+1} > \inf_k$. Thus, $\inf'_k = \inf_k$. Since $h_{k+1} \leq h_k$, we have that $\sup_{k+1} + \inf_{k+1} \leq \sup_k + \inf_k \implies \sup_{k+1} < \sup_k$, and so $\sup'_k = \sup_k$. Thus, the first maximum in I'_k is the first maximum in I_k and the last minimum in I'_k is the last minimum in I_k. Since I_k satisfies $P2$ then so does I'_k. ■

The idea of the algorithm should now be clear. The addition of a new point creates a new level set satisfying $P1$ and $P2$. If this new level set also satisfies $P3$, then we are done, and have constructed the isotonic regression for the sequence augmented by this one point. If not, then we merge the last two level sets, maintaining $P2$ and $P3$, and not altering any of the other level sets. We continue to merge until $P3$ is satisfied for the last level set, which must eventually happen. At this point we have a regression that satisfies $P1$-$P3$ and so it is the isotonic regression for the augmented sequence.

Note that I_K is the right most level set of $\mathcal{C}_i$, i.e., $I_K = [i_K, i]$. This rightmost level set is the union of i with some number (possibly zero) of the level sets (from right to left) of $\mathcal{C}_{i-1}$. The remaining level sets of $\mathcal{C}_i$ will be the level sets of $\mathcal{C}_{i-1}$ that remain after the merging. In fact, the remaining level sets will be exactly the level sets of $\mathcal{C}_{i_K-1}$, where it is understood that $\mathcal{C}_{i_K-1} = \{\}$ if $i_K = 1$.

Proposition 4.3.15 $\mathcal{C}_i$=$\mathcal{C}_{i_K-1} \cup \{I_K, h_K\}$.

Proof: If $i = 1$, there is nothing to prove. Assume that $i > 1$ and that the claim holds for all $\mathcal{C}_j$ with $j < i$. Let $\mathcal{C}_i = \{I_\alpha = [i_\alpha, j_\alpha], h_\alpha\}_{\alpha=1}^{K}$. By construction, C_{i-1} is given by

$$\mathcal{C}_{i-1} = \{(I_1, h_1), \ldots, (I_{K-1}, h_{K-1}), (S_1, h'_1), \ldots, (S_M, h'_M)\}, \qquad (*)$$

where M is possibly zero, and $I_K = \cup_i S_i \cup \{i\}$. Let $S_i = [\alpha_i, \beta_i]$, where $\alpha_1 = i_K$ and $\beta_M = i - 1$. By the induction hypothesis,

$$\begin{aligned}
\mathcal{C}_{i-1} &= \mathcal{C}_{\alpha_M - 1} \cup \{S_M, h'_M\}, \\
\mathcal{C}_{\alpha_M - 1} &= \mathcal{C}_{\alpha_{M-1} - 1} \cup \{S_{M-1}, h'_{M-1}\}, \\
\mathcal{C}_{\alpha_{M-1} - 1} &= \mathcal{C}_{\alpha_{M-2} - 1} \cup \{S_{M-2}, h'_{M-2}\}, \\
&\vdots \\
\mathcal{C}_{\alpha_2 - 1} &= \mathcal{C}_{\alpha_1 - 1} \cup \{S_1, h'_1\}.
\end{aligned}$$

Combining these equalities and using the fact that $\alpha_1 = i_K$, we get that

$$\mathcal{C}_{i-1} = \mathcal{C}_{i_K - 1} \cup_i \{S_i, h'_i\}.$$

using (*), we identify that $\mathcal{C}_{i_K-1} = \{(I_1, h_1), \ldots, (I_{K-1}, h_{K-1})\}$, concluding the proof. ■

4.3.4 L_∞-Prefix-Isotonic Regression: Algorithms

Here, we will give the linear time algorithm for L_∞-prefix-isotonic regression that follows from the results of the previous section, along with the analysis of its run time. Our algorithm will process points from left to right. After processing the new point x_i, we will have constructed the isotonic regression $\mathcal{C}_i$ as discussed in the previous section by merging the rightmost two intervals until $P1$-$P3$ are satisfied.

By Proposition 4.3.15, to reconstruct $\mathcal{C}_i$, we only need to know l_i, the index of the first point of its rightmost level set, the level, h_i, of this rightmost level set, and how to construct C_{l_i-1}. This can be recursively achieved by only storing the parameters l_i and h_i, for every i. The algorithms are given in Algorithm 4. The correctness of this algorithm follows from the results of the previous section, specifically Lemmas 4.3.13, 4.3.14. Further, $\mathcal{E}_\infty(\mathcal{C}_i)$ is stored in $D[i]$. By Proposition 4.3.15, the output of the algorithm stores all the necessary information to extract $\mathcal{C}_m$ as shown in the recursive function $RECONSTRUCT$.

What remains is to analyse the computational complexity of the algorithms. First consider the prefix-isotonic regression. lines 7,8,13 constitute 8 operations, thus contributing about $8n$ operations to the total run time. The merging while loop, lines 9-12, uses 6 operations. The maximum number of intervals is n. Each time a merge occurs, this maximum drops by 1. Since this maximum is bounded below by 1, this means that there are at most $n - 1$ merges, so the *total* time spent merging is about $6n$ operations, and the condition of the while loop is checked at most $2n$ times, so the runtime of this algorithm is bounded by Cn where $C \approx 14$. There are at most n level sets at any time, and each level set needs to store 5 numbers, $i_\alpha, j_\alpha, \inf_\alpha, \sup_\alpha, h_\alpha$. The additional space for L, H, D is $3n$, for a total memory requirement bounded by $C'n$, where $C' \approx 8$.

Algorithm 4 Algorithms for L_∞ prefix-isotonic regression.

// **Algorithm to perform L_∞-Prefix-Isotonic Regression.**
// **Input:** $\mathbf{x} = \{x_i | i \in [1, n]\}$.
// **Output:** L, H, D. $\{L[i] = l_i$, $H[i]$=level of $[l_i, i]$ in $\mathcal{C}_i$, $D[i]$=distortion of $\mathbf{x}_i\}$
$I_1 = [1, 1]$, $\inf_1 = x_1$, $\sup_1 = x_1$, $h_1 = x_1$, $K = 1$; {Initialization}
$L[1] = 1$, $H[1] = h_1$, $D[1] = 0$; {Initialization of outputs}
for $i = 2$ to n **do**
  $K \leftarrow K + 1$
  $I_K = [i, i]$, $\inf_K = x_i$, $\sup_K = x_i$, $h_K = x_i$, $D[i] = D[i-1]$;
  while $h_K \leq h_{K-1}$ and $1 < K$ **do**
    $I_{K-1} \leftarrow I_{K-1} \cup I_K$; $\inf_{K-1} \leftarrow \min(\inf_{K-1}, \inf_K)$; $\sup_{K-1} \leftarrow \max(\sup_{K-1}, \sup_K)$;
    $K \leftarrow K - 1$; $h_K = \frac{1}{2}(\inf_K + \sup_K)$; $D[i] = \max(D[i], \sup_K - \inf_K)$;
  end while
  $L[i]$=left endpoint of I_K; $H[i] = h_K$;
end for

$RECONSTRUCT(m)$
// **Output** $\mathcal{C}_m$, the isotonig regression for $\mathbf{x}_m$, assuming L, H are global.
if $m = 0$ **then**
  return $\{\}$;
end if
return $RECONSTRUCT(L[m] - 1) \cup \{[L[m], m], H[m]\}$;

It is not hard to analyse the recursion for $RECONSTRUCT$, and a straightforward induction shows that the runtime is $O(m)$.

4.3.5 L_∞ Unimodal Regression

As pointed out in [72], a prefix-isotonic regression can easily be modified to yield the optimal unimodal regression. The next proposition shows that the crossover point in the L_∞ unimodal regression can always be chosen at a maximum in the sequence (any maximum). Thus, a simpler algorithm that follows directly from the prefix-isotonic regression is to first find a maximum in $\mathbf{x}$ (linear time). Now perform isotonic regression on the sequence to the left of the maximum and the reversal of the sequence to the right. More specifically, suppose that the maximum is x_m. Now consider the sequences $\mathbf{x}_l = \mathbf{x}[1, m]$, $\mathbf{x}_r = \mathbf{x}[m, n]$, and let $\mathbf{x}_r^R$ be the reversal of $\mathbf{x}_r$. Let $\mathcal{C}_l$ and $\mathcal{C}_r^R$ be the isotonic regressions for $\mathbf{x}_l$ and $\mathbf{x}_r^R$ respectively. Then the union, $\mathcal{C}_l \cup \mathcal{C}_r$ (where $\mathcal{C}_r$ is the reversal of $\mathcal{C}_r^R$) is the unimodal regression, with the merging of the last level set of $\mathcal{C}_l$ and the first level set of $\mathcal{C}_r$, as they will have the same level, equal to the maximum. All that remains is to prove that the crossover point can always be chosen at a maximum.

Proposition 4.3.16 *The crossover point in the unimodal regression of* $\mathbf{x}$ *can always be chosen to be a maximum (any maximum) of* $\mathbf{x}$.

Proof: Let $\mathcal{C}$ be the unimodal regression, and let x_i be the crossover point, so

$$w_1 \leq w_2 \leq \cdots \leq w_i \geq w_{i+1} \geq \cdots \geq w_n.$$

Let $\mathbf{x}_l = \mathbf{x}[1,i]$, $\mathbf{x}_r = \mathbf{x}[i,n]$. Since $\mathbf{w}[1,i]$ is an isotonic regression for $\mathbf{x}_l$ and $\mathbf{w}^R[1,n-i+1]$ is an isotonic regression for $\mathbf{x}_r^R$, the error of the regression is $\mathcal{E}_\infty(\mathcal{C}) \geq \frac{1}{2}\max(D(\mathbf{x}_l), D(\mathbf{x}_r^R))$. Let x_m be any maximum not equal to x_i (if x_i is a unique maximum, then we are done, otherwise x_m exists). Without loss of generality, since a unimodal regression for $\mathbf{x}^R$ is $\mathcal{C}^R$, we can suppose that $m > i$. Let $\mathbf{x}_1 = \mathbf{x}[1,m]$, let $\mathbf{x}_2 = \mathbf{x}[m,n]$, and let $\mathbf{x}_c = \mathbf{x}[i,m]$. For the unimodal regression constructed from the two isotonic regressions on $\mathbf{x}_1$ and $\mathbf{x}_2^R$, x_m will be a crossover point. We show that the error of this regression cannot be more than the error of $\mathcal{C}$. The error of this regression is given by $\max(D(\mathbf{x}_1), D(\mathbf{x}_2^R))$. Since x_m is a maximum, $D(\mathbf{x}_r^R) = \max(D(\mathbf{x}_c^R), D(\mathbf{x}_2^R))$, so $\mathcal{E}_\infty(\mathcal{C}) = \max(D(\mathbf{x}_l), D(\mathbf{x}_c^R), D(\mathbf{x}_2^R))$. $D(\mathbf{x}_1)$ is given by

$$\begin{aligned} D(\mathbf{x}_1) = &= \max_{1\leq k\leq m}\{\max(\mathbf{x}[1,k]) - x_k\} \\ &= \max\left(\max_{1\leq k\leq i}\{\max(\mathbf{x}[1,k]) - x_k\}, \max_{i\leq k\leq m}\{\max(\mathbf{x}[1,k]) - x_k\}\right) \end{aligned}$$

The first term on the right hand side is $D(\mathbf{x}_l)$. Since x_m is a maximum, the second term is bounded by $\max_{i\leq k\leq m}\{x_m - x_k\} = D(\mathbf{x}_c^R)$. Thus $D(\mathbf{x}_1) \leq \max(D(\mathbf{x}_l), D(\mathbf{x}_c^R))$, and so

$$\max(D(\mathbf{x}_1), D(\mathbf{x}_2^R)) \leq \max(D(\mathbf{x}_l), D(\mathbf{x}_c^R), D(\mathbf{x}_2^R)) = \mathcal{E}_\infty(\mathcal{C}).$$

■

4.4 Conclusion and Discussion

For L_1-isotonic regression we presented an output sensitive algorithm whose running time is linear in n when the number of possible values that the levels of the isotonic regression can take is bounded by K. In the worst case, $K = n$ and the algorithm is no worse than existing algorithms. The open question that remains is whether the median isotonic regression can be computed in linear time, or to prove that it cannot. Presented algorithms can be extended without much effort to the case of minimizing a weighted L_1 error. In this case, all the results remain true, with minor modifications, by replacing the standard median with the weighted median.

For L_∞ isotonic and unimodal regression, we have given simple (not requiring sophisticated data structures) linear time algorithms. We are unaware of any other published results relating to the L_∞ regression.

Bibliography

[1] M. B. Ayer, H. D., G. M. Ewing, W. T. Reid, and E. Silverman. An empirical distribution function for sampling with incomplete information. *Annals of Mathematical Statistics*, 1955.

[2] F. Aurenhammer. Voronoi diagrams - a survey of a fundamental geometric data structure. *ACM Symposium on Computational Geometry*, 20-28, 1995.

[3] C. Bradford Barber, David P. Dobkin, Hannu Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, Volume 22, Issue 4, 469 - 483, 1996.

[4] C. Barber, D. Dobkin, H. Huhdanpaa. The Quickhull Algorithm for Convex Hull. *Geometry Center Technical Report GCG53*, Univ. of Minnesota, MN, 1993.

[5] K. P. Bennett and E. J. Bredensteiner. Duality and geometry in SVM classifiers. In *Proc. 17th International Conf. on Machine Learning*, pages 57–64, 2000.

[6] K. P. Bennett and O. L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, pages 23–34, 1992.

[7] Arjan Berkelaar and Roy Kouwenberg. Dynamic asset allocation and downside-risk aversion. citeseer.ist.psu.edu/berkelaar00dynamic.html.

[8] Tomasz R. Bielecki, Stanley Pliska, Jiongmin Yong. Optimal investment decisions for a Portfolio with a Rolling Horizon Bond and a Discount Bond. *To appear*, 2004.

[9] Tomasz R. Bielecki, Jean-Philippe Chancelier, Stanley Pliska, Agnes Sulem. Risk sensitive portfolio optimization with transaction costs. *To appear*, 2004.

[10] Fischer Black. Treynor, Jack L. How to use security analysis to improve portfolio selection. *Journal of Business*, pages 66–85, January 1973.

[11] B. Borchers and J. E. Mitchell. Using an interior point method in a branch and bound algorithm for integer programming. *Technical report* 195, Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY, 12180, July 1992.

[12] O. Bousquet and A. Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2002. To appear.

[13] N. Chakravarti. Isotonic median regression, a linear programming approach. *Math. of Oper. Research*, 1989.

[14] T. M. Chan. Output-sensitive results on convex hulls, extreme points, and related problems. In *Proc. 11th Annual Symposium on Computational Geometry*, pages 10–19, 1995.

[15] B. Chazelle. An optimal algorithm for intersecting three-dimensional convex polyhedra. In *IEEE Sympos. on Found. of Comp. Sci. (FOCS)*, volume 30, pages 586–591, 1989.

[16] V. Chvtal. *Linear Programming*. W. H. Freeman and Company, New York, 1983.

[17] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. Mcgraw-Hill, 2001.

[18] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines (and other kernel-based learning methods)*. Cambridge University Press, 2000.

[19] G. B. Dantzig. Maximization of a linear function of variables subject to linear inequalities. *Activity Analysis of Production and Allocation*, Wiley, New York, 1951, pp 339-347.

[20] L. Devroye and T. Klincsek. Average time behavior of distributive sorting algorithms. *Computing*, 26: 1-7, 1980.

[21] L. Devroye and G. T. Toussaint. A note on linear expected time algorithms for finding convex hulls. *Computing*, vol. 26, pp. 361-366, 1981.

[22] I. I. Dikin. Iterative solution of problems of linear and quadratic programming. *Sov. Math. Doklady*, 8(66):674–675, 1967.

[23] D. Dobkin and D. Kirkpatrick. A linear algorithm for determining the separation of convex polyhedra,. *J. Algorithms*, 6:381–392, 1985.

[24] D. P. Dobkin and D. G. Kirkpatrick. Fast detection of polyhedral intersections. *Theoretical Computer Science*, 27:241–253, 1983.

[25] David P. Dobkin, David G. Kirkpatrick. Fast detection of polyhedral intersections, *Lecture Notes in Computer Science*, 140 (1982) 154-165.

[26] D. P. Dobkin and D. G. Kirkpatrick. Determining the separation of preprocessed polyhedra - a unified approach. In *Proc. 17th International Colloquium on Automata, Languages and Programming*, pages 400 – 413, 1990.

[27] M. Frisén. Unimodal regression. *The Statistician*, 1980.

[28] Z. Geng and N.-Z. Shi. Isotonic regression for umbrella orderings. *Applied Statistics*, 1990.

[29] Thomas H. Goodwin. The information ratio. *Financial Analysis Journal*, 54(4):43–43, 1998.

[30] R. L. Graham. An efficient algorithm for determining of the convex hull of a planar set. *Information Processing Letters*, 132-133, 1972.

[31] Bhaswar Gupta and Manolis Chatiras. The interdependence of managed futures risk measures. pages 203–220. Center for International Securities and Derivatives Markets, October 2003.

[32] Jiqing Han. Yang Liu, Xiaohui Yu. Sharp ratio-oriented active trading: A learning approach. *SSRN Electronic Paper Collection*, 2001.

[33] M. Junger, G. Reinelt, and S. Thienel. Practical problem solving with cutting plane algorithms in combinatorial optimization. *Combinatorial Optimization: DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pp. 111 - 152. AMS, 1995.

[34] Thomas Hellstrom. Optimizing the sharpe ratio for a rank based trading system. In *Portuguese Conference on Artificial Intelligence*, pages 130–141, 2001.

[35] C. Helmberg and F. Rendl. Solving quadratic (0,1)-problems by semidefinite programs and cutting planes. *Technical report* SC-95-35, Konrad-Zuse-Zentrum fuer Informationstechnik, Berlin, 1995.

[36] Sergei Issaenko. Domenico Cuoco, Hua He. Optimal dynamic trading strategies with risk limits. *SSRN Electronic Paper Collection*, 2001.

[37] G. Kalai. A subexponential randomized simplex algorithm. *Proc. 24th Annual ACM Sympos. Theory Comput.*, 475-482 (1992).

[38] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984.

[39] L. G. Khachiyan. A polynomial algorithm in linear programming (in russian). *Doklady Akademii Nauk SSSR*, 244:1093–1096, 1979.

[40] Eva K. Lee, John E. Mitchell. Branch-and-Bound Methods for Integer Programming. *Encyclopedia of Optimization*, Kluwer Academic Publishers, August 2001.

[41] Jiming Liu. Samuel P. M. Choi. Optimal time-constrained trading strategies for autonomous agents. *In Proceedings of International ICSC Symposium on Multi-agents and Mobile Agents in Virtual Organizations and E-Commerce (MAMA 2000)*, 2000.

[42] Hong Liu. Optimal Consumption and Investment with Transaction Costs and Multiple Risky Assets. *The Journal of Finance*, Vol. 59 Issue 1 Page 289 February 2004.

[43] M. Magdon-Ismail, J. H.-C. Chen, and Y. S. Abu-Mostafa. The multilevel classification problem and a monotonicity hint. *Intelligent Data Engineering and Learning (IDEAL 02), Third International Conference*, August 2002.

[44] M. Magdon-Ismail, Personal communication.

[45] J. Matousek, M. Sharir, E. Welzl. A subexponential bound for linear programming. *Proc. 8th Annual Sympos. Comput. Geometry, ACM*, 1-8 (1992).

[46] Oliver Mihatsch and Ralph Neuneier. Risk-sensitive reinforcement learning. *Mach. Learn.*, 49(2-3):267–290, 2002.

[47] John E. Mitchell. Branch-and-Cut Algorithms for Combinatorial Optimization Problems. *Handbook of Applied Optimization*, pp. 65-77, Oxford University Press, January 2002. ISBN: 0-19-512594-0.

[48] John E. Mitchell. Cutting plane algorithms for integer programming. *Encyclopedia of Optimization*, Kluwer Academic Press, August 2001.

[49] John E. Mitchell, Panos Pardalos, Mauricio G. C. Resende. Interior point methods for combinatorial optimization. *Chapter in Handbook of Combinatorial Optimization*, Kluwer Academic Publishers, 1998.

[50] R. A. Mureika, T. R. Turner, and P. C. Wollan. An algorithm for unimodal isotonic regression with application to locating a maximum. Technical report, Department of Mathematics and Statistics, University of New Brunswick, 1992.

[51] Brian Neelon, David B. Dunson. Bayesian Isotonic Regression and Trend Analysis. *To Appear, 2003.*

[52] Y. Nesterov and A. Nemirovsky. *Interior Point Polynomial Algorithms in Convex Programming*. SIAM Studies in Applied Mathematics. Society for Industrial and Applied Mathematics, 1994.

[53] Online encyclopedia of financial terms. *http://www.investopedia.com*, 2004.

[54] *http://www.cs.princeton.edu/∼ah/alg_anim/version1/GrahamScan.html.* Online demonstration of Graham's scan algorithm.

[55] *http://www.cs.princeton.edu/∼ah/alg_anim/version1/QuickHull.html.* Online demonstration of Quick-Hull algorithm.

[56] G. Pan. Subset selection with additional order information. *Biometrics*, 1996.

[57] P. M. Pardalos and G.-L. Xue. Algorithms for a class of isotonic regression problems. *Algorithmica*, 1999.

[58] P. M. Pardalos, G.-L. Xue, and L. Yong. Efficient computation of an isotonic median regression. *Appl. Math. Lett.*, 1995.

[59] C.D. Perttunen D. Jones and B.E. Stuckman. Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Application*, 79:157–181, 1993.

[60] F. P. Preparata and M. I. Shamos. *Computational Geometry: an Introduction*. Springer-Verlag, New York, 1985.

[61] T. Robertson and P. Waltman. On estimating monotone parameters. *Ann. Math. Stat.*, pages 1030–1039, 1968.

[62] T. Robertson, F. T. Wright, and R. L. Dykstra. *Order Restricted Statistical Inference*. Wiley Series in Probability and Statistics. Wiley, new York, 1988.

[63] Rudholm-Alfvin T. Pedersen C.S. Selecting a risk-adjusted shareholder performance measure. *Journal of Asset Management*, 4(3):152–172, September 2003.

[64] Matthew Saffell. John Moody. Learning to trade via direct reinforcement. *IEEE Transactions on Neural Networks*, 12(4):875–889, 2001.

[65] William F. Sharpe. Mutual fund performance. *Journal of Business*, pages 119–138, January 1966.

[66] William F. Sharpe. Adjusting for risk in portfolio performance measurement. *Journal of Portfolio Management*, pages 29–34, 1975.

[67] J. Sill and Y. S. Abu-Mostafa. Monotonicity hints. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 9, pages 634–640. Morgan Kaufmann, 1997.

[68] David M. Smith, Christophe Faugere, Hany A. Shawky. Sell discipline and institutional money management. *Journal of Portfolio Management*, 30(3), 2004.

[69] Miguel Sousa Lobo, Maryam Fazel, Stephen Boyd. Portfolio optimization with linear and fixed transaction costs. *Submitted to Operations Research*, Oct 2002.

[70] Chester S. Spatt, Robert M. Dammon. The optimal trading and pricing of securities with asymmetric capital gains taxes and transaction costs. *The Review of Financial Studies*, 9(3):921–952, 1996.

[71] Sabine Stifter, Eugen Ardeleanu. Intersection algorithms: a comparative study. *http://citeseer.ist.psu.edu/6603.html*, 1994.

[72] Q. F. Stout. Optimal algorithms for unimodal regression. *Computing Science and Statistics*, 32, 2000.

[73] D. Tasche and L. Tibiletti. Ap-proximations for the value-at-risk approach to risk-return analy-sis. *Working paper. Technische Universiat Unchen. http://citeseer.ist.psu.-edu/tasche01approximations.html*, 2001.

[74] G. W. P. Thompson. Optimal trading of an asset driven by a hidden Markov process in the presence of fixed transaction cost. *Collection of research papers*, Judge Institute of Management Working Papers, 2002.

[75] V. N. Vapnik. *The Nature of Statistical Learning Theory.* Springer–Verlag, 1995.

[76] H. Wolkowicz, R. Saigal, and L. Vandenberghe, editors. *Handbook of Seimidefinite Programming: Theory, Algorithms, and Applications.* Kluwer Academic Publishers, Dordrecht, The Netherlands, 2000.

[77] Valeri I. Zakamouline. Optimal portfolio selection with both fixed and proportional transaction costs for a CRRA investor with finite horizon. *Web-site of Norwegian School of Economics and Business Administration*, 2002.