An Algorithmic Approach to Investment

Author : AB PASKARAMOORTHY
Supervisor: PROF. T. GEBBIE

WITS
UNIVERSITY

A dissertation submitted in fulfilment of the requirements for the degree of Master of Science

in the

School of Computer Science and Applied Mathematics,
University of the Witwatersrand, Johannesburg.

February 20, 2019

Declaration

I, Andrew Bhavan Paskaramoorthy, declare that this dissertation titled, "*An Algorithmic Approach to Investment*" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

_____

Date:

_____

Acknowledgments

I would like to thank God, through Whom all things are possible, for the completion of this work. I would like to express my deep gratitude towards my parents for their care and financial support, without which this work would not be possible.

I would express my deep gratitude to my supervisor Prof. Tim Gebbie for his mentorship and guidance, and for the countless hours he has spent overseeing this work. I would like to thank Dr. Terence van Zyl, and Dr. Woolway for proof-reading portions of this work and for useful commentary. All errors are mine alone.

I would like to thank the DST-NRF Centre of Excellence in Mathematical and Statistical Sciences (CoE-MaSS) for their financial support. In addition, The financial assistance of the National Research Foundation (NRF) towards this research is hereby acknowledged. Opinions expressed and conclusions arrived at are those of the author and are not necessarily to be attributed to the CoE-MaSS or the NRF.

Nomenclature

Table 1: Nomenclature for Control Flow Charts

Symbol	Description	Literature References	Chapter References
r_t	Total return	2.4.7	3.2.2
r_t^e	Return in excess of the risk-free rate		
f_t	Factor realisations	2.4.2	3.2.2
z_t	Information Variables	2.4.3	3.2.2
β_t	Factor Loadings	2.4.2	3.2.2
$\mathbf{B}_t$	Matrix of factor loadings for all stocks		3.2.4
δ_t	Characteristic payoffs	2.4.7	3.2.2
ϵ_t	Idiosyncratic error		3.2.2
$\Sigma_{f,t}$	Covariance matrix of factor realisations		3.2.4
$\Sigma_{t\|\pi}$	Conditional covariance matrix	2.3.4	3.2.4
Ω_t	Expected return uncertainty	2.3.4	3.2.4
$\Sigma_{u,t}$	Unconditional covariance matrix	2.3.4	3.2.4
$\Sigma_{\epsilon,t}$	Idiosyncratic risk matrix		3.2.4
Σ_t^*	Shrinkage Target		43
Σ_t	Risk matrix used in portfolio construction		Alg. 2
e_t	Forecast error	2.5.2	43
λ	Exponential memory factor	2.5.2	43
π_t	Systematic expected return	2.3.1	3.2.3
μ_t	(Characteristic model) expected return	2.3.1	3.2.3
α_{t+1}	Active expected return		3.2.3, 3.2.4
Ψ_t	Active return shrinkage factor	2.3.4	3.2.4
$\alpha_{bl,t}$	Active blended expected return	2.3.4	3.2.4
κ	Shrinkage Intensity		43
γ	Risk tolerance	2.2.2	3.2.3
w_t	Vector of portfolio weights	2.2.2, 2.2.4	3.2.3
ϕ	Predictive function		Alg. 2
C	Portfolio constraint set		Alg. 2

The table contains the nomenclature used in Algorithm 2 and Figures 3.2 and 3.1. The *Literature References* Column contains the most pertitent sections from the literature review which introduce the respective quantities. The main contribution of this research is contained in Chapter 3. The *Chapter References* contains section references for each quantity, within Chapter 3. All quantities above refer to an arbitrary time period t.

The table does not present an exhaustive list for the entire thesis. Where new variables are introduced, they are immediately defined.

Matrices are indicated with bold capital letters, eg. the matrix of factor loadings for all stocks in the cross-section. $\mathbf{B}_t$. Vectors are indicated by bold lower case letters e,g. the vector of factor loadings for a single stock. β_t, whilst scalars are typseset in standard text, eg. the market beta for stock i, $\beta_{i,t}$.

Estimates and Predictions are both indicated with a hat, eg. $\hat{\beta}_t$. Realised observations are indicated with tilde, eg. realised returns at time t are denoted $\tilde{r}_t$.

Abstract

The objective of this research is to develop a framework for an online low-frequency investment algorithm and compare its performance against the equally-weighted and market-cap weighted benchmark portfolios. The research presents an algorithmic framework for a control system that automates quantitative active portfolio management. The control system sequentially constructs a strategic benchmark portfolio to earn systematic risk premiums, and an active tactical portfolio to exploit temporary opportunities to earn excess returns.

Portfolio controls are determined through mean-variance optimisation. The online property of the control system is achieved through recursively estimating the mean-variance inputs with an adaptive algorithm. Inputs for the systematic benchmark portfolio are determined using a rational asset pricing model. Active returns are determined with an ad-hoc predictive model. Risk factors are selected by the investor and excess returns are defined against these factors. Tactical bet size is determined by the relative realised predictive accuracy between systematic and active expected return predictions. Portfolios are robust to estimation errors through offline regularisation of the mean and covariance estimates.

A specific configuration of the framework is implemented and tested out-of-sample using approximately ten years of weekly data from the JSE. The out-of-sample results did not demonstrate statistically significant outperformance over the benchmarks on a gross and net basis. Sharpe-Ratio point estimates over the out-of-sample period indicated benchmark outperformance on a gross basis, but benchmark underperformance after incorporating transaction costs.

Contents

List of Tables **vii**

List of Figures **viii**

1 Introduction **1**
1.1 Introduction . . . 1
1.1.1 Background . . . 1
1.1.2 Motivation . . . 2
1.1.3 Research Objectives . . . 3
1.1.4 Thesis Structure . . . 4

2 Literature Review **5**
2.1 Modeling Uncertainty . . . 5
2.1.1 Fundamental Analysis . . . 5
2.1.2 State-Contingent Payoffs . . . 6
2.2 Modern Portfolio Theory . . . 9
2.2.1 Diversification . . . 9
2.2.2 The Mean-Variance Efficient Frontier . . . 9
2.2.3 Neoclassical Portfolio Choice . . . 12
2.2.4 Intertemporal Portfolio Choice . . . 13
2.2.5 Towards Practical Portfolio Choice . . . 17
2.3 Practical Portfolio Choice . . . 18
2.3.1 Tactical Asset Allocation . . . 18
2.3.2 Error-Maximisation and Sensitivity . . . 19
2.3.3 Bayesian Portfolio Choice . . . 20
2.3.4 The Black-Litterman Model . . . 20
2.4 Asset Pricing Models . . . 23
2.4.1 Introduction . . . 23
2.4.2 No-arbitrage and the APT . . . 23
2.4.3 Extending the APT . . . 25
2.4.4 Identifying Factors . . . 28
2.4.5 Predictability . . . 29
2.4.6 Beyond Equilibrium and Efficiency . . . 31
2.4.7 Hierarchical Causality in Financial Economics . . . 32
2.5 System Identification for Automatic Control . . . 34
2.5.1 Adaptive Control . . . 34
2.5.2 Recursive System Identification . . . 34

3 An Online Framework for Factor-based Portfolio Control **41**
3.1 Introduction . . . 41
3.1.1 Theoretical Limitations and Practical Challenges . . . 42
3.2 Problem Formulation . . . 43

3.2.1 Objective . . . 43
3.2.2 Asset Price Dynamics . . . 44
3.2.3 Tactical Asset Allocation . . . 45
3.2.4 Regularization . . . 46
3.2.5 An Online Algorithm for Factor-Based Portfolio Control . . . 47

4 Implementation **56**
4.1 Data . . . 56
4.1.1 Market and Company Variables . . . 58
4.2 Methodology . . . 60
4.2.1 Data Preparation . . . 60
4.2.2 Offline User Choices . . . 61
4.2.3 Hyper-Parameter Tuning . . . 66
4.2.4 Covariance Prediction and Portfolio Construction . . . 71
4.3 Results . . . 74
4.3.1 Performance Measures . . . 74
4.3.2 Portfolio Performance . . . 80
4.3.3 Investigating the Out-of-sample Results . . . 83

5 Conclusion **86**
5.1 A Framework for Online Portfolio Management . . . 86
5.2 Implementation and Empirical Performance . . . 87
5.3 Shortcomings, and Recomendations for Future Work . . . 88

Bibliography **89**

A Literature Review **95**
A.1 MV Frontier . . . 95
A.2 Mean-Variance Frontier with Risk-Free Asset . . . 96
A.3 Tactical Asset Allocation . . . 96
A.4 CAPM derivation . . . 97
A.5 Woodbury's Matrix Identity . . . 98
A.6 Derivation of the Recursive M-Estimator . . . 98

B Code Patterns **100**
B.1 Workflow . . . 100
B.2 RLM Filter . . . 112
B.3 Cross-Sectional Regressions . . . 114
B.4 EW Mean . . . 115
B.5 EW Covariance . . . 116
B.6 EW Theil Statistics . . . 118
B.7 Invert Diagonal Matrix with NaNs . . . 119

List of Tables

1 Nomenclature for Control Flow Charts iii

4.1 Summary of Datasets 56
4.2 Characteristics Models 64
4.3 Optimal hyper-parameters for the algorithmic portfolio for each investible universe 80
4.4 In-sample Annualised Sharpe-Ratio Point Estimates 80
4.5 In-sample PSR Matrix 81
4.6 Out-of-sample Annualised Sharpe Ratio Point Estimates 82
4.7 Out-of-sample PSR matrix 82

List of Figures

2.1 A Two-Period State Diagram 7
2.2 The Mean-Variance Efficient Frontier 10
2.3 The Mean-Variance Efficient Frontier with a Risk-Free Asset 11
2.4 The Intertemporal Efficient Cone 16

3.1 Annotated Control Flow Chart for Factor-Based Portfolio Control Algorithm 50
3.2 Control Flow Chart for Factor-Based Portfolio Control Algorithm 51

4.1 Comparison of Gross Cumulative Returns Over Full Historical Sample 76
4.2 Comparison of Net Cumulative Returns Over Full Historical Sample 77
4.3 Out-of-Sample Comparison of Gross Cumulative Returns 78
4.4 Out-of-Sample Comparison of Net Cumulative Returns 79
4.5 Temporal Behaviour of Factor Loading Esimates 85

List of Algorithms

1 Online Parameter Estimation (OPE) 40

2 Factor-Based Portfolio Control (FBPC) Algorithm 49
3 Online Parameter Estimation 53

4 Implementation - Systematic Return Prediction 68
5 Implementation - Active Return Prediction 70
6 Implementation - Portfolio Construction 72

This page is intentionally left blank.

Chapter 1

Introduction

1.1 Introduction

1.1.1 Background

Since the inception of the Information Age, the finance industry has undergone widespread computerisation affecting how we invest in capital instruments. Technical advancement has facilitated growth in the study of finance as well as in its industrial application. Investment, as an academic and practical endeavour, is maturing from the art of valuation and stock-picking to the quantitative discipline of asset pricing and portfolio optimisation. The increased mathematical development of the field has brought further structure and discipline to investment decisions.

Whilst it may be argued that investment practice has always used metrics to inform investment decisions, the use of the term “quantitative" is typically reserved for systematic and sequential investment processes that are founded upon stochastic models of asset prices. The description “systematic and sequential" expresses the requirement that the investment process of a valid quantitative strategy provides consistent results across different datasets. In addition, quantitative strategies are characterised by modelling macro-properties of the market rather than considering individual assets. This approach is not arbitrary; it is a consequence of the principle of diversification, the conceptual cornerstone of modern finance. Diversification can be concisely described as the principle that covariance amongst assets, and not idiosyncratic variation, determines a portfolio’s risk. Thus, covariance between assets is the driver of value in a large market.

Stock-picking is by no means obsolete, but a strong argument against its application in well-functioning markets can be made based on one’s view of *market efficiency* [1]. Market efficiency relates to the information content in asset prices, and hence the ability to use this information to inform profitable trades. The literature concerning distribution of information in the market and the ability to profitably predict long-term price behaviour has not reached a consensus in over three decades of study [2].

The method of stock-picking is *Fundamental Analysis* which, in its original form, involves carrying out extensive research on an individual firm to project its future financial conditions, in order to assess if it is currently undervalued. This is a time-consuming task requiring predominantly subjective judgements, making it a costly process that is difficult to scale. In addition, to find an undervalued firm requires an informational advantage. Informational symmetry is reinforced by the empirical evidence of active managers struggling to outperform a market index benchmark on a net basis [3]. Interestingly, the market-cap portfolio can be considered as the first quantitative

strategy, justified by the first asset pricing model, the CAPM [4–6]. The evolution of quantitative equity strategies has followed the considerable progress of the academic discourse on financial markets. The corresponding investment practice has developed from passive "hold the market" strategies to quasi-active strategies that involve carefully tilting a portfolio to exploit mispriced pockets of risk [7].

Automation is amongst the current salient interests of the financial services industry [8]. The interest in automation is a consequence of the campaign to improve execution through cost reduction. Besides reducing costs, automated investment mitigates the risk of behavioural biases of investment managers. A related technology that is disrupting the sector is Machine Learning. Where automation can be thought of as primarily cost-reducing, Machine Learning can be used to gain informational advantages. Quantitative strategies lend themselves to automation owing to their systematic and sequential nature. Machine Learning comprises methods that are able to analyse large datasets. As information becomes more abundant, machine learning applications in finance will be increasingly sought after. Machine learning offers the potential to shift academic and industrial focus towards *quantitative meta-strategies*. A quantitative meta-strategy can be defined as the algorithmic management of investment strategies [9]. Indeed, with the possibility of testing and controlling over hundreds of candidate strategies, quantitative processes that manage alternate strategies becomes more important.

1.1.2 Motivation

Quantitative techniques aim to discover empirical regularities to determine the mechanisms that govern market dynamics in order to inform investment decisions. This is centred around differentiating signal from noise in asset prices. This is a sophisticated task since system noise largely dominates the true signal. The unique interactions between the masses of heterogeneous self-serving agents within the financial market creates an incredibly noisy environment. Furthermore, the signal is adaptive - the laws of the system change over time. Agents in the financial system have intelligence and are able to learn [10]. Accordingly, agents' interactions change over time resulting in evolving price dynamics. In a financial market, it is almost impossible to conduct repeated experiments under the same conditions to test hypotheses. In application, the market is in a state of flux, and the theoretical economic assumption of *ceteris paribus* is never true. These qualities inhibit the empirical study of finance and prevent the field from being characterised as a science.

Given the difficulty of conducting experiments on a complex adaptive system, how does one extract knowledge? Economics offers two distinct alternatives, a *Normative* approach and a *Positive* approach, both of which are often combined in practice. Positive economics can be briefly summarised as starting with facts and subsequently developing a theory. Given the adaptive behaviour of markets, so-called “facts" that are discovered are often temporary [11]. It is unclear whether profitable opportunities discovered in historical datasets are the result of statistical overfitting [12,13], or reflect an opportunity that existed but has been exhausted through competitive bidding [11]. The problem of overfitting is particularly pertinent in machine learning based investing, since it involves large-scale searches for an optimal strategy all tested on the same historical dataset [14]. Being overfit, the performance of this strategy is unlikely to persist in future. False discoveries are common, and for this reason it is important that “facts" have economic justification, and that economic insight precedes strategy development.

In contrast to the Positive approach, a Normative theory starts with a series of assumptions which

are economically justifiable, to derive a idealised model that explains asset prices [15]. The model is then tested empirically. This is the approach favoured by Neoclassic financial economists, who tend to view all facts that are inconsistent with *rational* investor behaviour as the result of overfitting [1]. While this has been the dominant approach for studying financial markets, the archetypal Neoclassical models have been an empirical failure [16]. Various sub-fields, such as Behavioural Finance, have been developed in order account for empirical anomalies [17]. Amongst the major predictions made by Neoclassical finance is the inability to outperform the aggregate wealth (also known as the market) portfolio. Although this proposition may appear extreme, it is still useful since it advocates conservative portfolio advice: investors should invest in a market index portfolio. Thus, Neoclassic asset pricing theories can still be a useful approximation in spite of its empirical failures.

In an automated investment process, human intervention is greatly reduced. It is necessary to have a clearly defined model of the system, that is justified both empirically and economically, before constructing an automation. The adaptive behaviour exhibited by markets [10], creates the need to investigate the market's unobserved meta-processes. These are the combinations of micro and macro processes which generate the array of temporary anomalous facts. Accordingly, the market can be approximated by a hierarchy of interacting models, with each level representing the aggregate behaviour of market agents acting at different time-scales [18]. Within the hierarchical theoretical paradigm, this research is focussed on the layer inhabited by medium to long-term investors. At this layer, price behaviour reflects the aggregate market interpretation of market-wide (top-down) and stock-specific (bottom-up) information.

1.1.3 Research Objectives

The aim for this research is to develop an online algorithmic framework for a factor-based quantitative strategy that can respond to market adaptations, and is capable of extension to incorporate more advanced estimation techniques.

Specifically, the research seeks to recursively approximate an ex-ante mean-variance efficient portfolio; its dynamics consistent with the complex hierarchical theory presented by Wilcox and Gebbie [18]. Adopting the philosophical perspective that markets are adaptive, this research proposes a strategy that can exploit temporary opportunities resulting from the market's delayed assimilation of information. The success of this strategy will be measured against various passive strategies. Furthermore, the implementation of the factor-based strategy will be completely automated.

The research objectives of this project can be stated as:

- to develop a framework for an online factor-based portfolio strategy, accounting for the pertitent theoretical and practical issues encountered by systematic investors, so that it can be used for automated active quantitative investing
- to implement a particular iteration of the framework, making particular choices for each component, estimated on actual weekly market data from the JSE, so that the framework can be empirically tested and critiqued.

The success of the objectives will be assessed by testing the following hypothesis:

> "Can the implemented automated investment process demonstrate investment skill?".

The hypothesis test will be conducted using out-of-sample performance data. Investment skill is defined as persistent outperformance over a benchmark strategy, and is measured using the Sharpe Ratio. The benchmark strategies used in this research is the market-cap weighted portfolio (*the Market portfolio*), and the 1/n portfolio (*the Naive Diversification portfolio*).

Thus, the hypothesis can be more formally restated as:

"Is the Sharpe-ratio of the implemented automated strategy greater than that of the benchmark Strategy?"

The corresponding null hypothesis can be stated as "there is no significant difference in the Sharpe-Ratio between common diversification strategies and the automated strategy". The null hypothesis reflects the possibility that any historical outperformance can be explained by the fortunate random variation exhibited by the system. The hypothesis is tested both gross and net of trading costs.

1.1.4 Thesis Structure

Our goal is to automate a successful quantitative portfolio construction process. Chapter 2 contains the relevant literature motivating the design of the online algorithmic framework. The literature review contains discussion regarding the key theoretical developments relating to portfolio choice and asset pricing, as well as their shortcomings. The main contributions of this research is contained in chapters 3 and 4. Chapter 3 contains the design and description of the *Factor Based Portfolio Control Algorithm*, meeting the first of the stated objectives. The Factor Based Portfolio Control Algorithm is a novel contribution motivated by the studies [19–21], and is developed from first principles. Chapter 4 addresses the second objective, where a specific implementation of the framework is tested on JSE market data. Section 4.1 contains a description of the data and the data preparation process. Section 4.2 contains the methodology corresponding to the specific implementation. The corresponding code patterns can be found in Appendix B [1]. Section 4.3 contains the performance analysis, including testing whether the algorithm was able to demonstrate investment skill. Conclusion 5 concludes, and provides ideas for further research.

[1] Code patterns used in thise thesis are based on code patterns used in [19, 20], but are reformulated and extended to meet the objectives of thesis. Code patterns from [19, 20] can be retrieved from https://github.com/apaskara/Algo_Invest

Chapter 2

Literature Review

2.1 Modeling Uncertainty

2.1.1 Fundamental Analysis

Prior to the formulation of *Modern Portfolio Theory* [22], the investment practice was dominated by a set of valuation techniques known as *Security Analysis*. This subsection provides a brief review of the theory of fundamental analysis to define key relevant financial concepts and to provide historical context. This is a very limited depiction about the subject of valuation. A thorough treatment can be found in [23].

The objective of security analysis is to determine the *intrinsic value* of a security, and exploit differences between the intrinsic values and market prices of securities. Intrinsic value, also known as fundamental value and sometimes fair value, represents the present value of a firm to an omniscient investor who had foreknowledge of the future financial conditions and performance of a firm, indefinitely. Security analysis comprises techniques to estimate intrinsic value through making judgements about the future cashflows arising from an asset.

Intrinsic value can be estimated by discounting the future cashflows generated by a security at an “appropriate" discount rate:

$$V = \sum_{t=1}^{n} \frac{CF_t}{(1+r)^t} \tag{2.1}$$

where V is the present value of the asset at time t, CF_t is the estimated cashflow at time t, r is the discount rate. There are n cashflows representing the lifetime of the asset, but this need not be finite. A discounted cashflow analysis exercise provides a subjective valuation that depends on the investors information about a firm's future prospects, the manner in which he uses this information to form judgements about cashflows, and crucially, his choice of discount rate. The choice of discount rate is equivocal. Alternative appropriate discount rates include the investors borrowing rate, his opportunity cost, the risk-free rate, or a more general threshold rate. The discount rate can also be adjusted to reflect the riskiness of an investment.

The success of a strategy built on security analysis depends on the accuracy of assumptions concerning future cashflows. The future profitability of a firm is inherently uncertain and depends on a variety of environmental and business-related factors such as macro-economic conditions and the quality of the firm's employees. Classical discounted cashflow analysis, being a predominantly accounting-centric discipline, does not explicitly incorporate uncertainty through the use of stochastic variables. Rather, variation in experience and uncertainty in assumptions is incorporated by deflating valuations. The adjustment made to the valuation is called the *margin-*

of-safety and is analagous to a crude computation of an economic risk-premium. Like a risk-premium, the margin-of-safety reflects that investors are averse to risk. Furthermore, investors intuitively understood that a diversification stabilises portfolio returns, although this was not formally incorporated into the portfolio selection process.

Estimates Intrinsic value reflect the current available information, and change over-time in response to changes in market outlook, and realized historical performance. Intrinsic value is unobservable and is typically stable over short time horizons, since it reflects long-term profitably. In contrast, the market price of security is observable and is subject to short term fluctuations, perhaps driven by irrational views. It is this irrationality that provides opportunities to the fundamental analyst, since it causes securities to be mispriced. A security is mispriced when there is a difference between its intrinsic value and its market price.

2.1.2 State-Contingent Payoffs

The intellectual leap from security analysis to Modern Portfolio Theory was the explicit recognition of uncertainty, and the discovery of its implications for portfolio choice and asset valuation. Since the 1980s, uncertainty in financial markets has been represented through state-contingent outcomes. This characterisation is a natural extension to the discounted cashflow analysis, where the future cashflows and discount rates are mathematically represented as random variables, defined over a set of mutually-exclusive *states*.

Mathematically, states comparable to individual outcomes that form the foundational events of a sample space. Economically, states represent alternative possible future economic scenarios that affect asset returns, investor decisions, and their exogenous sources of wealth. Furthermore, states may reflect the economic conditions at the time, or may reflect news of future economic conditions.

For example, the release of national economic statistics may indicate that the country has entered a recession (the conditions at the time) and may indicate that the recession may persist for some time (the future conditions). Realized returns from the most recent historical period may be poor due to the recession. Poor returns limits an investor's wealth and current consumption. Furthermore, the investor will adjust his future expectations for returns, affecting his future optimal consumption policy. The distinction between a state reflecting contemporaneous conditions and a state reflecting future conditions implies that the intertemporal consumption problem can not, in general, be treated in the same way as the single-period problem.

To define intertemporal uncertainty, we restrict attention to the finite horizon, discrete-time, discrete state-space case. Suppose that at each time period the future states of the world are modelled by a finite number of mutually-exclusive exogenously determined states $\S = \{s_1, s_2, ..., s_m\}$. Then, the possible outcomes over time can be described by a multinomial tree with m^t nodes at time t, where each node in the tree represents a state-realization. The market is assumed to be *complete*. Market completeness requires that a payoff is defined at every node in the tree.

Each branch is a possible sample path and represents a potential evolution of the information set. The corresponding payoffs describe all cashflows arising from an asset including its sale price. In the case that the set of future states are path dependent, then this necessarily implies some predictability. Knowledge of the sample path will provide knowledge of the possible future states.

A Two-Period State Diagram

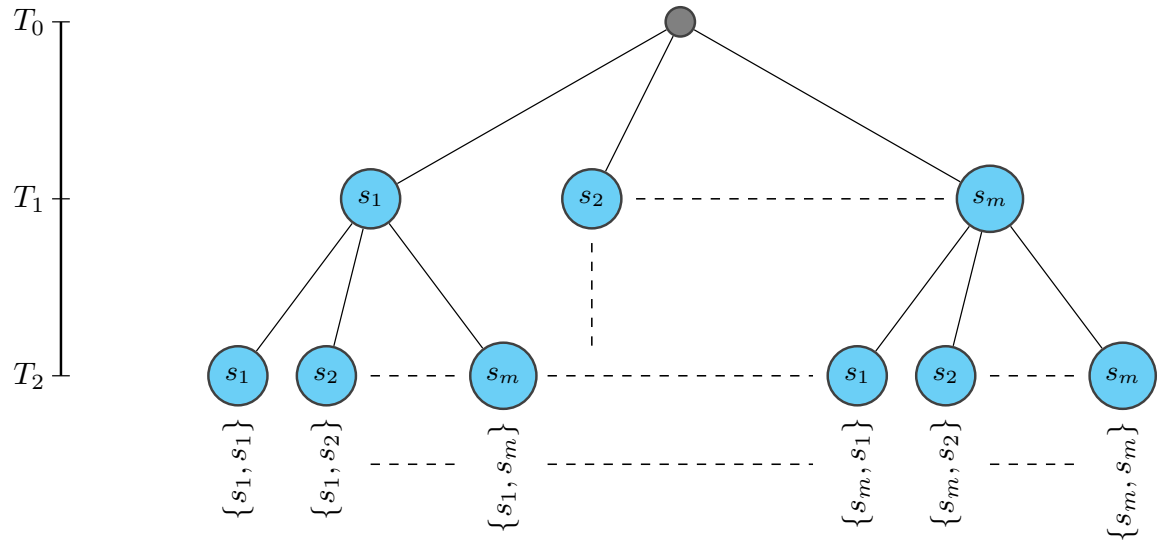


Figure 2.1: The diagram shows the possible state-realisations at each time-step for a single-period time horizon. At each time step, there are m possible outcomes. Asset prices are defined as a function of the m outcomes. If the outcomes at T_2 or the function defining asset prices differed for each node at T_1, then asset prices would be predictable.

However, even in the case that states are path independent, predictability may be possible through path dependent probability measures. Predictive variables are called state-variables, which may be the real-world state-realizations themselves or a proxy-variable if the state-realization is unobservable.

Each asset has a set of payoffs populated over the entire tree. In the case of equities, these payoffs are either the results of a dividend or reflect the price of the asset. Different assets have different payoff random variables. By forming a portfolio of assets in a complete market, an investor can own any of the payoffs branches. Diversification is the result that portfolio composition can be used to alter payoff distributions, such that payoff variation is reduced across states. Markowitz develops an optimal portfolio rule to minimise variation of portfolio returns for a specified level of average return under the assumption of that asset returns are normally distributed. This is discussed at greater length in the next section.

Contemporary asset pricing is the stochastic analogue to the discounted cashflow equation (Eq. 2.1) seen previously . Each sample path describes a possible set of payoffs which can be discounted to determine a value. The discount rates themselves may also be path dependent. The price over each sample path can be composed to determine a single value for an asset that explicitly incorporates uncertainty. The stochastic discounted cashflow equation, collapsed into the single-period case, is described mathematically as:

$$P = \mathbb{E}[MX] \tag{2.2}$$

where P is the price of the asset, M is the stochastic discount factor, and X is the asset's future payoffs.

Diversification implies that asset values can not be determined in isolation in a stochastic environment. The relationship between the valuations of different assets is captured by a common *stochastic discount factor*, whose specification is the principal focus of asset pricing theories. Asset

pricing is discussed in section 2.4.

2.2 Modern Portfolio Theory

2.2.1 Diversification

Recognising that investors seek returns but are averse to risk, Markowitz [22] considered how an investor could explicitly incorporate risk in the portfolio construction process. His idea was to represent securities as alternative production possibilities between average return and volatility, thus simplifying the portfolio selection problem to a textbook microeconomics problem. Analogous to a production-possibilities curve, portfolios of securities could be formed to define an *efficient frontier* of minimum risk for a given level of return.

The derivation of the efficient frontier formalises the intuition of diversification in mathematical terms. Diversification is the principle that dividing wealth across imperfectly correlated assets reduces the overall risk of the portfolio. The volatility of an asset is equivalent to computing the $L2-norm$ of centered asset returns. In this case, diversification is an economic restatement of the triangle inequality. Demonstrating this mathematically,

$$\sigma_p = \sqrt{\boldsymbol{w}^\top \Sigma \boldsymbol{w}} = \sqrt{\sum_i \sum_j w_i w_j \sigma_i \sigma_j \rho_{ij}} < \sum_i w_i \sigma_i \tag{2.3}$$

$$\text{when} \quad \rho_{ij} < 1 \tag{2.4}$$

More generally, since measures of risk are typically convex functions, a convex combination of assets (a portfolio) will have lower combined risk than the convex combination of risks.

Furthermore, an investor's particular portfolio choice will depend on his utility function, which explicitly captures his preferences for risk-adjusted returns. The functional form and parametrisation of the investor's utility function will determine the investor's appetite for risk. If it is the case that means and variances and insufficient to characterise the distribution of returns, Markowitz's formulation could be defended by assuming investors have quadratic or logarithmic utility functions.

Neither the required distributional assumption, nor the required utility assumption are borne out in reality. The distribution of asset returns has been empirically shown to have skewness [24]. Quadratic utility functions have satiation points, where additional wealth beyond this point decreases the investor's utility. Nonetheless, the mean-variance frontier can be viewed as a local approximation to an optimal portfolio, and is widely used as the backbone to an industrial investment process owing to its simplicity. The derivation of the mean-variance frontier follows.

2.2.2 The Mean-Variance Efficient Frontier

Suppose an investor aims to form a fully-invested portfolio, $\boldsymbol{w}$, to maximise his risk-adjusted return at the end of a single-period, subject to his risk tolerance γ. For the time being, the set of assets does not include the risk-free asset. The derivation of the optimal portfolio where a risk-free asset is included is discussed in section 2.2.2. The distribution of asset returns are assumed to be sufficiently explained by the mean, $\boldsymbol{\mu}$ and covariance parameters, Σ, which are constant through the investment horizon. It is assumed that the investor's subjective estimates of the return distribution parameters are equal to the true parameters. This assumption is known as investor *rationality*, and ensures that the average ex-post performance of an optimal portfolio is the same as the ex-ante expectation.

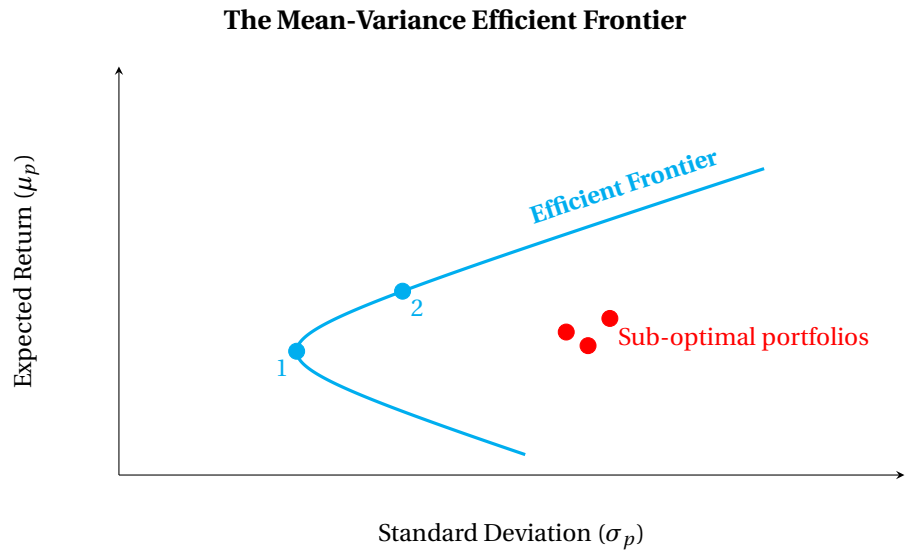


Figure 2.2: The dots on the efficient frontier represent optimal portfolios for different levels of risk tolerance. The dot labelled 1 indicates the global minimum variance portfolio. The dot labelled 2 represents an arbitrary optimal portfolio. The difference in standard deviation between the GMV and any other portfolio represents idiosyncratic risk.[1]

Mathematically, the investor's faces the following constrained optimization problem:

$$\max_{w} \boldsymbol{w}^\top \boldsymbol{\mu} - \frac{\gamma}{2} \boldsymbol{w}^\top \Sigma \boldsymbol{w} \tag{2.5}$$

subject to the full investment constraint,

$$\boldsymbol{w}^\top \mathbf{1} = 1 \tag{2.6}$$

Note that the optimization is specified in terms of returns, rather than absolute levels of wealth.

Solving the constrained optimization problem yields the following solution for a fully-invested optimal portfolio:

$$\boldsymbol{w}^* = \left(1 - \frac{\mathbf{1}^\top \Sigma^{-1} \boldsymbol{\mu}}{\gamma}\right) \frac{\Sigma^{-1} \mathbf{1}}{\mathbf{1}^\top \Sigma^{-1} \mathbf{1}} + \left(\frac{\mathbf{1}^\top \Sigma^{-1} \boldsymbol{\mu}}{\gamma}\right) \frac{\Sigma^{-1} \boldsymbol{\mu}}{\mathbf{1}^\top \Sigma^{-1} \boldsymbol{\mu}} \tag{2.7}$$

The constrained optimization problem can be solved by optimizing the Lagrangian. The proof is contained in appendix A.1.

The composition of the optimal portfolio changes for different levels of risk tolerance. The set of all possible optimal portfolios is called the efficient frontier. Varying the risk tolerance parameter will draw out the efficient frontier, which is a hyperbola in the mean-standard deviation space. The hyperbola of is sometimes referred to as the *efficient bullet*.

The Two-Fund Separation Theorem

The solution in Equation 2.7 has the form $(1-x)\,\boldsymbol{w}_g + x\boldsymbol{w}_r$, where each component $\boldsymbol{w}_g$ and $\boldsymbol{w}_r$ has the property that $\boldsymbol{w}'\mathbf{1} = 1$. Thus, the efficient portfolio is a convex combination of two portfolios: the global the minimum variance (GMV) portfolio and an optimal portfolio of risky assets (the maximum risk-adjusted return portfolio). The efficient frontier is defined as the set

[1] figure adapted from http://loualiche.gitlab.io/www/tikz-pgf/teaching/investments.html

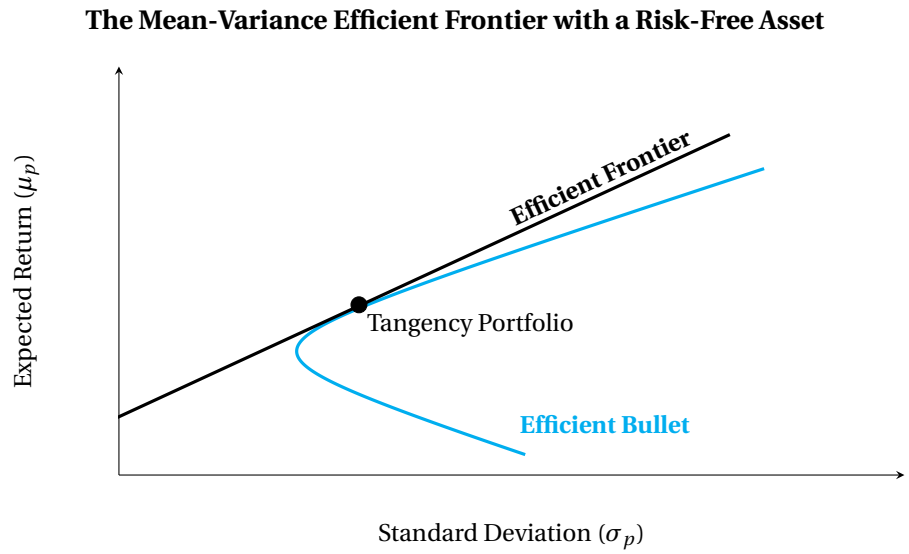


Figure 2.3: The efficient frontier including the risk-free asset is a straight line which intersects the efficient bullet at a single point in M-V space. All efficient portfolios are a mixture of the tangency portfolio and the risk-free asset. The tangency portfolio contains the optimal mix of risky assets. [2]

of maximally diversified portfolios at different levels of return. As the required return increases, some diversification is sacrificed in the pursuit of higher average returns. The global minimum variance represents the maximally diversified portfolio and is located at the apex of the mean-variance frontier. When there is no risk-free asset, idiosyncratic risk increases by traversing along the frontier towards more risky optimal portfolios.

Thus, an investor can hold any optimal portfolio by holding proportions of any two other optimal portfolios. Rather than determining the allocation between a large number of assets, the optimal portfolio choice for any investor could be made by allocating wealth over two optimal portfolios. This result is known as the *mutual-fund separation theorem*. The mutual-fund separation theorem reduces the objective of the single-period investor to identify two optimal funds.

The task of forming an efficient portfolio is further simplified in the case where the set of available assets includes a risk-free asset. A risk-free asset is defined to have zero volatility, and zero covariance with any other asset. The set of convex combinations between the risk-free asset and any other asset is a straight line in the mean-standard deviation space. The derivation of the efficient frontier including a risk-free asset is provided in the appendix. The efficient frontier including the risk-free asset is a straight line that stretches from the risk-free asset and is tangent to the efficient bullet.

The tangency of the new efficient frontier can be intuitively understood through considering the alternatives. Portfolios that lie above the tangency line are not possible, thus the efficient frontier must touch the efficient bullet. If the straight line intersected the efficient bullet, there would exit portfolios on the efficient bullet that dominated portfolios on the straight line contradicting the definition of an efficient frontier. Thus, inclusion of a risk-free asset reduces the task of forming an efficient portfolio to identifying the tangency portfolio.

In the subsequent sections, we explore intertemporal portfolio choice. Extensions to the mean-

[2] figure adapted from http://loualiche.gitlab.io/www/tikz-pgf/teaching/investments.html

variance portfolio choice problem were cast in the framework of Neoclassical Economics to allow for economic interpretations to investor behaviour, and to describe the portfolio choice problem consistently with prevailing economic theory of non-financial assets.

2.2.3 Neoclassical Portfolio Choice

In the Neoclassic framework, investors are equivalent to households whose preferences are defined with regard to consumption only. Wealth carries no utility to a household beyond its ability to be used for consumption. Defining a utility function over wealth may be convenient in certain contexts, but strictly speaking, this is economically valid only to the extent that wealth acts as a proxy for consumption or as a variable that influences the consumption decision. By defining the investor's problem in terms of optimal consumption, it is possible to derive an equilibrium theory for financial assets that is consistent with the general equilibrium theory for the entire economy [25].

Under the Neoclassic framework, the portfolio choice problem is approached from the perspective of an intertemporal investor who must determine his optimal consumption policy order to maximise his expected utility subject to some budget constraint. At each time period, the rational agent can purchase assets using his available wealth to provide a future payoff. The payoffs from his invested wealth can be consumed, or reinvested for the next period. From the perspective of the single investor, asset prices and payoffs (and, equivalently, returns) are exogenous. Thus, the investor is said to be a *price taker*.

If we consider the discrete-time intertemporal formulation over an infinite horizon, the investors optimization problem can be mathematically described as follows:

$$\max_{w_t, C_t} \mathscr{U}(C_t, C_{t+1}, ...) = \mathbb{E}\left[\sum_{t=1}^{\infty} \beta^t U(\omega_t, C_t, t)\right] \tag{2.8}$$

$$\text{st} \quad W_{t+1} = \left(1 + r_{p,t+1}\right)(W_t - C_t) \tag{2.9}$$

$$r_{p,t+1} = \boldsymbol{w}_t' \boldsymbol{r}_{t+1} \tag{2.10}$$

$$\boldsymbol{w}_t' \mathbf{1} = 1 \tag{2.11}$$

where $\mathscr{U}$ is the investors intertemporal utility function, $U(\cdot,\cdot)$ is the investors utility function applicable over a single period, C_t is the investor's consumption at time t, and W_t is the investor's wealth at time t .

The first constraint equation describes the evolution of wealth through time. At each period the investor has the choice between consuming his wealth, C_t, and investing his wealth into return-bearing assets. Whatever is not consumed, $W_t - C_t$, is invested and earns a return. The second constraint equation states that the return earned on the portfolio $r_{p,t+1}$ depends on the investor's portfolio choice and the return generated by each asset, where $\boldsymbol{w}_t$ is a vector of weights on each portfolio at time t and $\boldsymbol{r}_{t+1}$ is the corresponding return from time t to time $t+1$.

The utility function, $\mathscr{U}$, is assumed to be *time-separable* and *time-independent*. Time-separability means that the intertemporal utility can be decomposed into a linear combination of utility functions defined at each period. Time-independence means that the utility function U is assumed to be the same at each time period. Both of these assumptions are made for mathematical simplicity. The utility function is usually assumed to be concave and increasing. These imply, respectively, that there is a diminishing marginal utility of consumption $U''(C) < 0$, and that investors prefer more to less $U'(C) > 0$. Diminishing marginal utility can be expressed as

risk-aversion. The curvature of the utility function implies that the loss in utility for a unit decrease in consumption is greater than the gain in utility following a unit increase of consumption. A risk-averse investor would require a (risk) premium to enter a fair gamble. In finance, an investor would require a return in excess of the risk-free rate to be exposed to variability in return.

In addition, the utility function, $\mathscr{U}$, is assumed to be *state-separable* and *state-dependent*. State-separability, like time-seperability, is a mathematical simplification that allows for linear combinations of utility functions to be taken over states. This allows for us to form expected utilities without difficulty. The state-dependency of the utility function reflects the phenomena that investors value consumption differently depending on the prevailing economic conditions. For example, the news of an economic recession may cause an investor to delay his consumption in order to spread it over future periods. During the recession his capacity to consume may be otherwise limited by his available wealth, which is dependent on investment returns and labour income.

2.2.4 Intertemporal Portfolio Choice

The optimization problem (Eq.2.8) states how an investor can form portfolios of assets to transfer wealth between periods to maximise intertemporal expected utility of consumption. The properties of the the optimal consumption and portfolio policies are determined by the parameters of the problem.

Solving the portfolio choice problem requires that we define the investor's utility function, his intertemporal budget constraint, the probabilities for each future state, and the state-dependent returns for each asset. The model set-up in Eq. 2.8 is a rich formulation allowing for a multitude of variations, which has resulted in numerous articles deriving alternative solutions to the portfolio choice problem.

Analyses of the intertemporal problem started with Samuelson [26], and Merton [27]. In these initial studies, parameters were assumed to be constant and independent over time. Thus, the intertemporal problem could be simplified to the single-period problem, and the result that the mean-variance frontier contains optimal portfolios is recovered [27]. This provides no additional insight into normative behaviour. Rather, the formulation presented here follows Merton's subsequent article [28] and allows for temporal variation of parameters.

The information set captures all variables that affect an investor's decision at a point in time. This includes information relating to the *investment opportunity set* and information that influences an investor's preferences, such as the sequence of historical states. The investor's *investment opportunity set* is defined as the distribution of the set of returns available to investor, generated by the set of all possible portfolios in the market. The true investment opportunity set is exogenously determined and reflects the true frequencies of occurrence for possible sample path, together with the correct specification of future payoffs. Time-variation in the distribution of returns follows if the set of possible future outcomes at any period is state-dependent.

Predictive variables in the information set can be used to determine the conditional investment opportunity set at a point in time. Such variables proxy for the unobservable state variables and are examples of "news" that would cause investors to update expectations about asset returns. Aggregate revaluations will affect prices. Although asset prices are nearly a random walk, a number of variables have demonstrated, albeit low, predictive ability. Some of the variables that are well-accepted to be empirically robust under a large battery of tests and across different

datasets include the dividend yield [29, 30], the price-to-book [31, 32] ratio, and short-term price momentum [33].

In addition the investment opportunity set, an investor's decision is affected by his wealth at that point in time as well as the evolution of wealth in successive periods. The investor's wealth determines his risk aversion, affecting his portfolio choice and the optimal utility he can attain. However, if the investor has a utility with the property of *Constant Relative Risk Aversion* (CRRA), his decisions will be independent of his level of wealth. In general, the information set includes the investor's wealth as a variable. Variables that form the information set are known as *state-variables* in the context of dynamic programming. From the preceding description of the information set, it should be clear that although information and states are related, the concept of state-variables are different from the states themselves. Accordingly, in the dynamic programming context, the state-dependence of the utility function can be generalized to state-variable dependence.

Merton [28] approaches the intertemporal portfolio choice problem using dynamic programming. Dynamic programming reformulates the task of finding an optimal decision policy over multiple periods into dynamically making a series of optimal single-period decisions. This is a consequence of the *principle of optimality*. Formally, if the optimal decision policy for a discrete time decision problem over a time horizon of T periods is given by the sequence $d_1, d_2, ..., d_T$, the principle of optimality states that the last k decisions must be optimal. The principal of optimality can be succinctly defined as "the completion of an optimal sequence of decisions must be optimal" [34]. The formula specifying the recursion between the cost of successive optimal decisions is called the Bellman Equation.

To solve the intertemporal optimization via dynamic programming, it is required that the dynamics of the state-variables are described as Markov processes. For further details, see [28]. We follow Merton's [28] canonical analysis to derive expressions for the optimal portfolio. The intertemporal portfolio choice problem is presented in the continuous-time setting is chosen to simplify the ensuing mathematics. Suppose that an investor can trade and consume continuously over time. His optimization problem is stated as:

$$\begin{aligned} &\max_{C,w} \mathbb{E}_0 \left[\int_0^T U(C_t, t, z_t)\, dt + U(W_T, T) \right] \\ \text{where} \quad &dW = \left[\boldsymbol{w}'(\boldsymbol{\mu} - \mathbf{1} r_f) + r_f \right] W dt - C_t dt + W \boldsymbol{w}' \Sigma d\boldsymbol{z} \end{aligned} \tag{2.12}$$

where the average instanteous return is indicated by $\boldsymbol{\mu}$, the risk-free rate is indicated by r_f, the covariance matrix of returns is represented by Σ, and the instantaneous change in the state variable is given by dz. The remaining variables are as before.

The constraint describes the instantaneous change in wealth, dW. This is the continuous-time version of the budget constraint and is economically equivalent to the budget constraint 2.9. In order to derive the wealth process, it is assumed that the investor has an initial endowment, W_0, and has no exogneous income. The change in the investor's wealth at any point in time is determined by his consumption choices C_t and the investment returns $r_{p,t} = \boldsymbol{w}_t' \boldsymbol{r}_t$ on the invested portion of wealth $W_t - C_t$ over an infinitesimal period dt. The investor is able to rebalance his portfolio continuously.

The investment return for an asset is derived from the price process $r_{i,t} = \frac{dS_{i,t}}{S_{i,t}}$. Asset prices dynamics are modelled by stochastic processes with time-varying parameters governed by the

state-variable z. The price process for the asset is defined as:

$$\begin{aligned} \boldsymbol{dS}_t &= \boldsymbol{\mu}(z_t)\boldsymbol{S}_t dt + \Sigma(z_t)\boldsymbol{dx} \\ \text{where} \quad dz &= m(z_t)dt + s(z_t)dy \end{aligned} \tag{2.13}$$

where the $\boldsymbol{dx}$ is an n-dimensional vector representing infinitesmal increments of a multivariate Wiener processes. As before, the state-variable z is assumed to follow a Markov process, where the parameters of the increments are functions, m and s, of the variable at time t only. The investment opporunity set includes a risk-free asset that earns a fixed rate of interest over time. Therefore, the risk-free asset has deterministic dynamics described by $dS_f = r_f S_{f,t}$.

From the optimization problem (Eq. 2.12), the investor's value function in continuous-time is defined as:

$$J(z, W, t) = \max_{C_t, \boldsymbol{w}_t} \mathbb{E}_t \left[\int_t^T U(C_s, s, z_s) ds + U(W_T, T) \right] \tag{2.14}$$

At the terminal date T, the value function is equal to the terminal utility assuming that terminal wealth is consumed, $J(z, w, T) = U(W_T) = U(C_T)$. The corresponding continuous-time Bellman equation is:

$$0 = \max_{C_t, \boldsymbol{w}_t} \left\{ U(C_t, t, z_t)\, dt - \delta J_t dt + E_t[dJ] \right\} \tag{2.15}$$

where δ is the instantaneous rate to discount utility over time, and corresponds to β in the discrete-time problem 2.8. The instantaneous change in the value function is described by dJ. Directly applying Ito's formula, the instantaneous change in the value function, dJ, is equal to [28]:

$$dJ = J_t dt + J_w dW + J_z dz \frac{1}{2}\left(J_{zz}(dz)^2 + J_{ww}(dW)^2 \right) + J_{wz}(dW)(dZ) \tag{2.16}$$

Incorporating the budget constraint into the process dJ provides a Bellman equation that can be directly maximised by finding the derivatives in terms of the decision variables $c, \boldsymbol{w}$ [28]. This provides the first-order conditions [28], which are:

$$0 = U'(C) - J_w \tag{2.17}$$

$$0 = J_w W \boldsymbol{w}^\top (\boldsymbol{\mu} - \mathbf{1} r_f) + \frac{1}{2}\left(J_{ww} W^2 \boldsymbol{w}^\top \Sigma_x \boldsymbol{w} dt \right) + J_{wz} W \boldsymbol{w}^\top \boldsymbol{\sigma_z} \tag{2.18}$$

The first condition equates the marginal value of wealth with the marginal utility of consumption. This condition is known as the *Envelope condition* and expresses the idea that at the optimal value, an investor is indifferent between additional consumption and additional saving. The Envelope condition is used to determine the optimal consumption policy. The optimal portfolio can be derived from the second first-order condition, and it equal to:

$$\boldsymbol{w} = \frac{J_w}{W J_{ww}} \Sigma^{-1}(\boldsymbol{\mu} - r) - \frac{J_{wx}}{W J_w w} \Sigma^{-1} \boldsymbol{\sigma}_z \tag{2.19}$$

The expression for the optimal portfolio reiterates the discussion of the discrete-time Bellman equation (Equation **??**) more directly. Here, we see that the intertemporal optimal portfolio consists of two component portfolios. The first term in Equation 2.19 follows from the single-period case, and represents an mean-variance optimal portfolio on the conditional efficient frontier. The conditional mean-variance frontier is updated dynamically according to changes in the state-variable. This portfolio is called the *myopic* portfolio as it maximises the instantaneous ex-ante risk-adjusted return, or in the discrete-time case, it maximises the ex-ante return over the next period. Using the envelope condition, it can be seen that the coefficient to the myopic

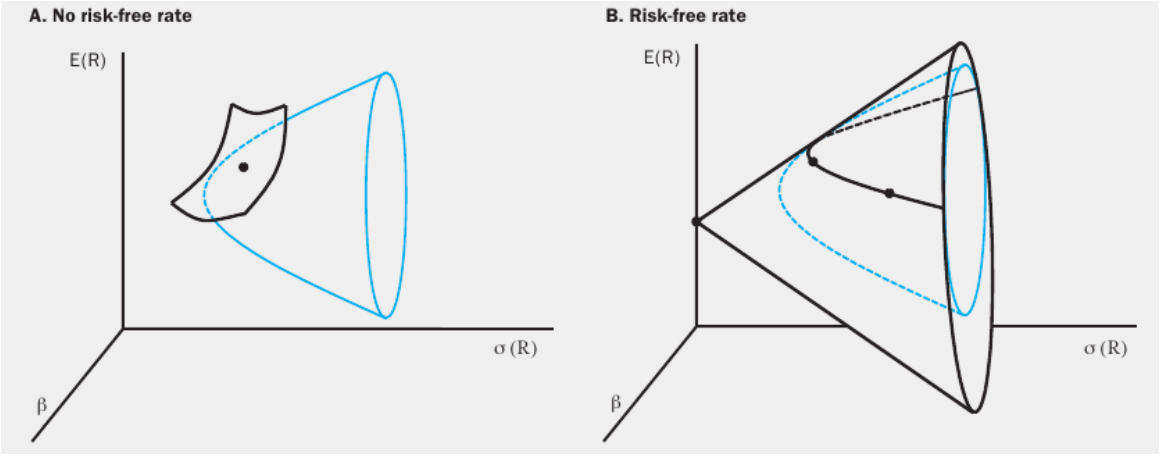


Figure 2.4: **Subfig. A:** The surface in blue represents the multifactor efficient frontier, whilst the surface in black represents the investors utility function. The dot depicts where the indifference surface is tangent to the efficient frontier, representing an optimal portfolio when there is no risk-free asset. **Subfig. B:** When an investor's preferences are a function of state-variable covariation, the efficient frontier becomes an efficient cone (depicted by the black surface). The surface in blue represents the efficient frontier without the risk-free assets. The set of tangency portfolios is represented by the black line with two dots. The fund seperation theorem states that every portfolio on the efficient cone can be attained as a combination of the two tangency portfolios (the black dots) and the risk-free asset. Figure reproduced from [35].

portoflio is the inverse of relative risk aversion. For a risk-averse investor, the relative risk aversion is always positive, which implies that an investor will always have a positive allocation to the myopic portfolio.

The second term in Equation 2.19 is a portfolio that maximises the correlation to the state variable. This portfolio is called the *hedging* portfolio since it used to hedge against adverse changes to the investment opportunity set. The degree to which the hedging portfolio plays a part in the investor's portoflio choice depends on how the value function changes with with wealth. Informally, this can be thought of as aversion to variation in the state-variable, and is analogous to relative risk aversion. The inclusion of this component means that the intertemporal investor's optimal portfolio is no longer on the mean-variance frontier. Conversely, the mean-variance frontier is sub-optimal. Cochrane [35] shows how the mean-variance efficient frontier can be extended to an efficient cone (Figure 2.4), when the state-variable covariation is included as a third axis.

The two-fund separation theorem states that the mean-variance frontier can be decomposed into a convex combination of any two different efficient portfolios. Since the risk-free asset has been included in this derivation, the mean-variance frontier becomes the capital market line. If a risk-free asset was not included, the first term would be the efficient bullet. Thus, by including of a single state-variable in the intertemporal problem, the two-fund separation theorem has become a three-fund separation theorem. This result can be extended to a more general $(2+m)$-fund separation theorem for the case where price processes are driven by m state variables.

The inter-portfolio allocation between on the myopic and hedging portfolio depends on the current state variables, the current level of wealth, and the value function. The value function depends on the choice and parameterisation of the investor's utility function. Determining the value requires solving the continuous-time Bellman equation above. The complexity of this problem depends on assumptions about state-space evolution and the specified preference function. In general, the Bellman equation is a nonlinear partial differential equation that is not

analytically tractable.

A number of articles [36–38] provide approximate solutions to the intertemporal asset-class allocation problem with a low-dimensional state-space. Extending these approaches to the problem of equity portfolio construction was, at the time the aritcles were published, computationally infeasible. The intertemporal portfolio choice problem has received little attention over the last 10 years, and has not extended into industrial application.

2.2.5 Towards Practical Portfolio Choice

In summary, this section has presented the canonical formulations of the portfolio choice problem. Markowitz's mean-variance optimizer derived the composition of an optimal single-period portfolio. A single-period optimal portfolio can be constructed using a minimum risk portfolio, and a risky portfolio [35]. Merton's analysis of the intertemporal portfolio choice problem demonstrated that an optimal portfolio is composed of $2 + m$ optimal portfolios, where the additional m portfolios are hedging portfolios. The hedging portfolio adjusts the composition of the optimal single-period portfolio to incorporate predictive information about the distribution of returns over subsequent periods as provided by the state-variables.

A mean-variance portfolio is ex-post optimal only if asset returns are normally distributed with known parameters. In practice, the distribution of asset returns must be estimated from historical data, and will contain estimation errors. For commercial implementation, the portfolio choice optimization must be modified to account for distributional uncertainty [39]. In the intertemporal problem, the optimal allocation between the myopic and hedging portfolios is yet to be solved. In practice, the portfolio construction process is simplified to a single-period problem, where a mean-variance optimal portfolio is dynamically constructed. In the next section, we discuss common modifications applied to the theoretical formulations to make the portfolio choice problem practical.

2.3 Practical Portfolio Choice

2.3.1 Tactical Asset Allocation

Given the computational intractability of the normative framework, sub-optimal portfolio construction solutions are implemented in practice [7]. Industrial portfolio selection consists of dynamically constructing "myopic" mean-variance portfolios, with updated return distribution parameters conditional on the latest available information. These parameters implicitly encode both long-term and short-term behaviour of the return processes. The process of dynamically tilting a long-term mean-variance portfolio to exploit short-term opportunities is known as Tactical Asset Allocation.

From section 2.2.2, a fully invested mean-variance portfolio has the solution:

$$\boldsymbol{w} = \left(1 - \frac{\mathbf{1}^\top \Sigma^{-1} \boldsymbol{\mu}}{\gamma}\right) \frac{\Sigma^{-1}\mathbf{1}}{\mathbf{1}^\top \Sigma^{-1}\mathbf{1}} + \left(\frac{\mathbf{1}^\top \Sigma^{-1} \boldsymbol{\mu}}{\gamma}\right) \frac{\Sigma^{-1}\boldsymbol{\mu}}{\mathbf{1}^\top \Sigma^{-1}\boldsymbol{\mu}} \tag{2.20}$$

The optimal portfolio solution can be rewritten as being composed of the global minimum variance benchmark portfolio, and a zero-cost portfolio that is used to earn additional returns by taking on additional risk [7]:

$$\boldsymbol{w} = \frac{\Sigma^{-1}\mathbf{1}}{\mathbf{1}^\top \Sigma^{-1}\mathbf{1}} + \frac{1}{\gamma} \frac{\Sigma^{-1}\left(\boldsymbol{\mu}\mathbf{1}^\top - \mathbf{1}\boldsymbol{\mu}^\top\right)\Sigma^{-1}}{\mathbf{1}^\top \Sigma^{-1}\mathbf{1}} \tag{2.21}$$

Lee [7] demonstrates that a zero-cost portfolio is necessarily composed of pairwise bets between assets. The size of the bets are proportional to the differences between the expected returns of the assets, scaled by the inverse covariance matrix and by the investors risk tolerance.

Lee [7] motivates tactical asset allocation by decomposing a mean-variance optimal portfolio into three component portfolios: a global minimum benchmark portfolio, a long-term strategic portfolio, and a short-term tactical portfolio. Suppose the long-term expected returns are given by $\boldsymbol{\pi}$. Then, the short-term expected returns in excess of the long-term returns are given by $\boldsymbol{\mu} - \boldsymbol{\pi}$. Substituting $\boldsymbol{\mu} = \boldsymbol{\mu} - \boldsymbol{\pi} + \boldsymbol{\pi}$, yields the following decomposition of the mean-variance portfolio:

$$\boldsymbol{w} = \frac{\Sigma^{-1}\mathbf{1}}{\mathbf{1}^\top \Sigma^{-1}\mathbf{1}} + \frac{1}{\gamma} \frac{\Sigma^{-1}\left(\boldsymbol{\pi}\mathbf{1}^\top - \mathbf{1}\boldsymbol{\pi}^\top\right)\Sigma^{-1}}{\mathbf{1}^\top \Sigma^{-1}\mathbf{1}} + \frac{1}{\gamma} \frac{\Sigma^{-1}\left(\left(\boldsymbol{\mu} - \boldsymbol{\pi}\right)\mathbf{1}^\top - \mathbf{1}\left(\boldsymbol{\mu} - \boldsymbol{\pi}\right)^\top\right)\Sigma^{-1}}{\mathbf{1}^\top \Sigma^{-1}\mathbf{1}} \tag{2.22}$$

In shorthand, the combined portfolio is written as, $\boldsymbol{w} = \boldsymbol{w}_g + \boldsymbol{w}_s + \boldsymbol{w}_{tc}$, where the subscripts tc refers to *tactical*. A full derivation is provided in appendix A.3.

The expression in (Eq. 2.22) decomposes the optimal frontier into two frontiers with the same benchmark. The first two component portfolios reflect the long-term investment, whilst the third component reflects deviations from the long-term strategy. Predictable long-term return behaviour has a theoretical justification using equilbrium theories of asset prices, and is empirically motivated by asset return mean-reversion. Long-term investment based on equilibrium arguments are synonymous with passive investment strategies, that invest in a market index. Investors claiming to add value beyond passive investment aim to justify their claims using the market index portfolio as a benchmark. Correspondingly, we call this long-term portfolio the strategic benchmark portfolio.

Short-term deviations from the benchmark can only be justified in the case of a sub-optimal benchmark. If the benchmark is the equilbrium portfolio, deviations from the benchmark

presume the current disequilbrium of the system. Investment strategies that deviate from the benchmark in search of superior risk-return behaviour are known as active management strategies.

In order to achieve mean-variance efficiency, the tactical portfolio must be constructed based on the excess return over the implied benchmark return. In the typical active management scenario, active managers use return estimates that are not benchmark-excess returns. This results in an active portfolio that is the same for any benchmark portfolio [40]. Active return estimates that are not benchmark-excess returns may incorporate the same information as the benchmark. In this case, positive tactical returns are not the result of superior predictions, but are rather due to additional exposure to the same risk premia in the benchmark portfolio.

2.3.2 Error-Maximisation and Sensitivity

The adoption of Markowitz theory as an industrial investment tool was initially met with resistance owing to the unwieldy outputs and poor out-of-sample performance [39]. The theory failed to adequately deal with errors in the inputs which ultimately dominated the optimizated controls. Resulting portfolios were said to be "error-optimized" as out-of-sample performance vastly diverged from what was expected [41]. The literature has shown Markowitz weights to be highly inefficient; portfolio composition was hyper-sensitive to the sampling variations in esimates [41, 42]. The use of Bayesian parameter estimates, implemented at various stages of the portfolio construction process, have shown success in regularizing the behaviour of the Markowitz optimizer.

Jobson and Korkie [41] study the distribution of the Markowitz estimators for the optimal portfolio weights, the expected returns, and the covariance matrix of returns . These estimator distributions were derived for the sample moment estimators of the means and the covariance matrix of the assets, assuming that the market returns are identically and independently distributed. Their analytical derivations showed that Markowitz estimators were biased in finite samples but asymptotically consistent. However, their simulation results showed slow convergence for estimators, requiring a number of observations that is practically infeasible.

Jobson and Korkie [41] demonstrate that Markowitz portfolios estimated in finite samples vastly underperform the true optimal portfolio. Additionally, the Markowitz portfolio underperforms an equally-weighted portfolio. This result is particularly problematic since the distrbution of returns has been empirically shown to vary over time. Key empirical studies in this regard are presented in the asset pricing section. In a time-varying system, historical data loses relevance with its age, and provides limited information to the current or future states of the system. Consequently, parameters will be estimated using sample sizes far smaller than that required for asymptotic convergence.

The results of Jobson and Korkie can be unpacked by addressing the complementary problems of estimation error and sensitivity. The error-maximisation property of the optimiser results in portfolios with large concentrated positions. Error-maximisation is potentially exacerbated if the covariance matrix estimate is ill-conditioned, which is another problem associated with finite sampling. Best and Grauer [42] demonstrate, analytically and through simulations, the extreme sensitivity of the Markowitz weights to changes in the input parameters. This extreme sensitivity causes the large variance in the sampling distribution for the Markowitz weights seen in Jobson and Korkie's study [41].

2.3.3 Bayesian Portfolio Choice

The abovementioned studies demonstrated that Markowitz portfolios are large and highly unstable. Barring concerns relating to trade-execution, these portfolios are justified if parameters are known. When parameters are uncertain, differences between estimated parameters may not be statistically significant, and large positions carry a large amount of out-of-sample risk. A Bayesian analysis of the portfolio choice problem is the natural starting point to derive a portfolio optimization procedure that accounts for uncertainty. In general, empirical evidence confirms that Bayesian approaches to portfolio construction provide superior out-of-sample performance over the traditional Markowitz portfolio [43, 44]. Local research confirms that the superior out-of-sample performance of Bayesian estimators [45, 46].
Brandt [47] surveys the literature and classifies Bayesian applications to portfolio choice problems into Plug-in estimation and Decision Theory.

Plug-in estimation methods aim to form superior point estimates through Stein-type shrinkage estimators for means [43] and covariances [44]. Decision theoretic approaches incorporate parameter uncertainty directly into the optimization objective function. Optimal weights are determined with respect to a distribution of returns, where the parameters have prior distributions. Depending on the specification of the prior distribution, the corresponding posterior hyper-parameters can have the form of a Shrinkage estimator, making the difference between these approaches an academic distinction.

As in the case with input parameters, shrinkage estimators can be derived for the weights themselves by optimally combining portfolio rules that have different statistical properties [48, 49]. Such portfolios may be optimal according to some Bayes decision rule, under which classical portfolio separation theorems are sub-optimal. Shrinkage portfolios resemble portfolio separation rules but comprise an additional mutual fund that accounts for estimation risk [48]. This additional portfolio provides error diversification, improving out-of-sample efficiency.

Another common approach to manage irregular weights is the use of constraints in the optimization. Imposing constraints in the optimization is a simple method to limit portfolio norms, but at the expense of reduced interpretability [42]. Jaganathan and Ma [50] explain how imposing a long-only constraint on the global minimum variance portfolio is equivalent to shrinking the covariance matrix, improving out-of-sample portfolio performance under circumstances where it would be otherwise expected. Similarly, imposing the long-only constraint on a mean-variance portfolio has the effect of shrinking the expected returns [51]. Lastly, constraints improve net returns by limiting turnover [51].

2.3.4 The Black-Litterman Model

In active management, the active return predictions are often expressed as the relative performance between two groups of assets. Beside the issue of sensitivity, the Markowitz optimizer requires a full set of absolute expected return estimates to be used. Black and Litterman's [52] solution was to use Theil and Goldberger's [53] mixed estimator to blend benchmark estimates and active views to form a Bayesian portfolio rule, which, benefiting from the implicit shrinkage properties, was able to simultaneously solve the problems of sensitivity and estimation error. Theil's mixed estimator is a Bayesian method that can combine multiple pieces of additional information about linear combinations of coefficients in a regression model. We present the original derivation of the Black-Litterman model below.

Suppose a sample of observations, $(\boldsymbol{y}_1, \mathbf{X}_1)$, satisfy the linear relationship:

$$\begin{aligned} \boldsymbol{y}_1 &= \mathbf{X}_1 \boldsymbol{b} + \boldsymbol{u}_1 \\ \mathbb{E}[\boldsymbol{u}] &= \mathbf{0} \\ \mathbb{E}[\boldsymbol{u}\boldsymbol{u}^\mathsf{T}] &= \Omega_1 \end{aligned} \tag{2.23}$$

where $\boldsymbol{y}_1$ is a vector of n_1 observations, $\mathbf{X}_1$ is a matrix of observations from the m dependent variables, $\boldsymbol{b}$ is a vector of parameters corresponding to each dependent variable.

Now suppose we have an alternate source of information about the parameters $\boldsymbol{b}$, such that observations from this second source satisfy the linear relationship:

$$\begin{aligned} \boldsymbol{y}_2 &= \mathbf{X}_2 \boldsymbol{b} + \boldsymbol{u}_2 \\ \mathbb{E}[\boldsymbol{u}] &= \mathbf{0} \\ \mathbb{E}[\boldsymbol{u}\boldsymbol{u}^\mathsf{T}] &= \Omega_2 \end{aligned} \tag{2.24}$$

where all variables are defined as before except that $\boldsymbol{y}_2$ and $\mathbf{X}_2$ consist of n_2 observations, where it is commonly the case that $n_1 \neq n_2$. Furthermore, this additional information source need not contain information about each parameter. The case of no additional information can be expressed by setting the column of observations in the matrix $\mathbf{X}$ to zero. Furthermore, the additional source may be consist of summarised information in the form of an alternative parameter estimate. For example, a single alternative parameter estimate can be incorporated through specifying $\boldsymbol{y}_2 = \boldsymbol{b}_2$ and $x_1 = 1$. The variance of the stochastic term , $var(u_2)$ would contain information about the certainty of the parameter.

Additional sources of information about the parameters $\boldsymbol{b}$ can be defined in the same way.

We can combine each source of information to write:

$$\begin{bmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix} \boldsymbol{b} + \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \end{bmatrix} \tag{2.25}$$

$$\mathbb{E} \begin{bmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{2.26}$$

$$\mathbb{E}\left(\begin{bmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{bmatrix} [\boldsymbol{u}_1^\mathsf{T} \quad \boldsymbol{u}_2^\mathsf{T}] \right) = \begin{bmatrix} \Omega_1 & 0 \\ 0 & \Omega_2 \end{bmatrix} \tag{2.27}$$

Note that the covariance is block diagonal, that is, there is no inter-model cross-correlation. However, the covariance matrix of each source of information may contain cross-correlations and heterskedastic errors.

The generalised least-squares estimator is for parameter $\boldsymbol{b}$ is:

$$\hat{\boldsymbol{b}} = \left([\mathbf{X}_1^\mathsf{T} \quad \mathbf{X}_2^\mathsf{T}] \begin{bmatrix} \Omega_1^{-1} & \mathbf{0} \\ \mathbf{0} & \Omega_2^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix} \right)^{-1} \left([\mathbf{X}_1^\mathsf{T} \quad \mathbf{X}_2^\mathsf{T}] \begin{bmatrix} \Omega_1^{-1} & \mathbf{0} \\ \mathbf{0} & \Omega_2^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{y_1} \\ \boldsymbol{y_2} \end{bmatrix} \right) \tag{2.28}$$

Multiplying the terms we get,

$$\hat{\boldsymbol{b}} = \left[\mathbf{X}_1^\mathsf{T} \Omega_1^{-1} \mathbf{X}_1 + \mathbf{X}_2^\mathsf{T} \Omega_2^{-1} \mathbf{X}_2 \right]^{-1} \left[\mathbf{X}_1^\mathsf{T} \Omega_1^{-1} \boldsymbol{y}_1 + \mathbf{X}_2^\mathsf{T} \Omega_2^{-1} \boldsymbol{y}_2 \right] \tag{2.29}$$

where the covariance matrix of the parameter estimate is given by:

$$Var(\hat{\boldsymbol{b}}) = \left[\mathbf{X}_1^\mathsf{T} \Omega_1^{-1} \mathbf{X}_1 + \mathbf{X}_2^\mathsf{T} \Omega_2^{-1} \mathbf{X}_2 \right] \tag{2.30}$$

Following the same approach we can derive the following mixed estimator applicable to N models that have no inter-model correlation:

$$\hat{\boldsymbol{b}} = \left[\sum_{i=1}^{N} \mathbf{X}_i^{\mathsf{T}} \Omega_i^{-1} \mathbf{X}_i^{\mathsf{T}}\right]^{-1} \left[\sum_{i=1}^{N} \mathbf{X}_i^{\mathsf{T}} \Omega_i^{-1} \boldsymbol{y}_i\right] \tag{2.31}$$

The equation 2.31 for the estimator has the intuitive interpretation that the optimal parameter estimate is a weighted average of the optimal estimates from each sample, where weights are determined in proportion to the variance matrices. This can be seen clearly by making the substitution:

$$\mathbf{K} = \left[\mathbf{X}_1^{\mathsf{T}} \Omega_1^{-1} \mathbf{X}_1 + \mathbf{X}_2^{\mathsf{T}} \Omega_2^{-1} \mathbf{X}_2\right] \left[\mathbf{X}_1^{\mathsf{T}} \Omega_1^{-1}\right] \tag{2.32}$$

Then,

$$\hat{\boldsymbol{b}} = \mathbf{K}\hat{\boldsymbol{b}_1} + (1 - \mathbf{K})\hat{\boldsymbol{b}_2} \tag{2.33}$$

The shrinkage interpretation follows immediately if $\mathbf{K}$ is viewed as the shrinkage intensity.

By defining the following quantities:

- $\boldsymbol{y}_1 = \pi$ is the vector of cross-sectional equilbrium expected returns,
- $\boldsymbol{y}_2 = Q$ is a vector of active return estimates, known as investor views
- $\mathbf{X}_1 = \mathbf{I}$ is the set of coefficients of the equilibrium estimates
- $\mathbf{X}_2 = \mathbf{P}$ is the matrix used to express absolute or relative active views
- $\Omega_1 = \tau\Sigma$ is covariance matrix that expresses uncertainty in equilibrium expected returns
- $\Omega_2 = \Omega$ is the uncertainty matrix associated with investor views
- $\hat{\boldsymbol{b}} = \hat{E[r]}$ is the vector of expected returns that we are trying to estimate

We can make the substitutions above to write the original Black-Litterman model:

$$\hat{E[R]} = \left[(\tau\Sigma)^{-1} + \mathbf{P}^{\mathsf{T}}\Omega^{-1}\mathbf{P}\right]^{-1} \left[(\tau\Sigma)^{-1}\pi + \mathbf{P}^{\mathsf{T}}\Omega^{-1}\mathbf{Q}\right] \tag{2.34}$$

2.4 Asset Pricing Models

2.4.1 Introduction

An asset pricing theory defines the prices for all assets under a set of idealised conditions. The mean-variance portfolio theory showed assets could be composed to enhance risk-adjusted returns. Since the development of the theory, investment decisions could no longer be done by considering assets individually. Similarly, an asset's ability to alter the risk of a portfolio, implied that idealised asset prices must be determined according to a portfolio. Thus, early asset pricing theories were concerned with the determination of an asset's covariation with a suitable portfolio. This is idea is reflected in the first-order conditions of the mean-variance optimization.

$$\boldsymbol{\mu} = \Sigma^{-1}\boldsymbol{w} \tag{2.35}$$

Expected returns are determined by their covariation to the tangency portfolio. The first asset pricing theory, the CAPM identified the tangency portfolio to determine expected returns in terms of the covariation with the Market portfolio. The CAPM, or rather the empirical failure of the CAPM, marked the beginning of the development asset pricing theories, and formed the cornerstone of the voluminous literature of subsequent refinements. The market is now recognised as demonstrating highly complicated dynamics, where the most authoritative asset pricing theories have been empirically shown to be only locally reasonable approximations of reality.

2.4.2 No-arbitrage and the APT

Rather than focusing on expected returns, contemporary asset pricing is discussed within the framework of the *stochastic discount factor*:

$$P = \mathbb{E}[MX] \tag{2.36}$$

where P is the price of the asset, M is the discount factor, and X is the asset's future payoffs. Price is the expected discounted future payoff, over a single period. In the case of an equity, the future payoff is composed of a dividend and the future price.

As in the previous section, for the purpose of this discussion, future uncertainty is captured by mutually-exclusive discrete states, $S = \{s_1, s_2, \ldots, s_n\}$. The discount factor is a vector of state-dependent discount rates. The role of the asset pricing theory is to specify how the discount rates are determined in the market to produce prices.

An assumption that underlies most pricing models, whether explictly or implicitly, is the no-arbitrage assumption. An arbitrage portfolio is defined as satisfying either of the following conditions [54]:

- provides a positive cashflow now with no possibility of a future loss, or
- requires no investment now (a zero-cost portfolio) but provides a postive future cashflow with no possibility of loss

In either case, an arbitrage portfolio provides the possibility of a net gain, without any possible future loss. Mathematically, this is expressed as [54]:

$$\boldsymbol{p\eta} \leq 0 \quad \text{and} \quad \mathbf{X}^{\mathsf{T}}\boldsymbol{\eta} > \mathbf{0} \tag{2.37}$$

where $\boldsymbol{p}$ is a (row) vector of asset prices, $\boldsymbol{\eta}$ is the number of shares held in each asset, $\mathbf{X}$ is a matrix of state-dependent payoffs for each asset. The assumption of No-arbitrage (NA) is to exclusion of arbitrage portfolios from the set of possible portfolios. Mathematically [55]:

$$\text{NA} \equiv \{\boldsymbol{\eta} | \boldsymbol{p}^\top \boldsymbol{\eta} \leq 0 \quad \text{and} \quad \mathbf{X}^\top \boldsymbol{\eta} > \mathbf{0}\} = \emptyset \tag{2.38}$$

Dybvig and Ross [56] were the first to demonstrate that the impossibility of forming an arbitrage portfolio led to a relationship between asset prices and asset payoffs. This is captured in the *Fundamental Theroem of Asset Pricing* (FTAP) which states that the following conditions are equivalent:

1. No-arbitrage
2. The existence of a positive linear pricing rule that prices all assets
3. The existence of a finite optimal demand for some agent that prefers more to less

The proof for the theorem can be found in [54].

We consider the equivalence of the first two statements. The assumption of no-arbitrage is equivalent to a unique linear mapping from asset payoffs to the price vector. In the discrete state-space, the existence of this mapping is represented by a vector, and can be represented mathematically as the vector $\boldsymbol{q}$ such that:

$$\boldsymbol{p} = \boldsymbol{q}^\top \mathbf{X} \tag{2.39}$$

The vector q is known as the state-price vector (or the state price density in a continuous state-space), since each component of the vector defines the price of an asset that has a unit payoff in that state. Thus, q is the price for the standard basis vectors of the payoff space.

Ross [55] demonstrates that the existence of the state-price vector has alternative equivalent representations. By rewriting the state prices as $q_i = \pi_i m_i$, where $m_i = \frac{q_i}{\pi_i}$, the assumption of no-arbitrage is equivalent to the existence of a positive stochastic discount factor. Note that the probabilities specified do not have to be the correct probabilities for the existence of a unique stochastic discount factor. It is important to note that no-arbitrage is a restriction excluding relative mispricings between assets. Absolute mispricings, however, are permissible under the FTAP. Stated differently, prices do not have to equal fundamental values.

Actual market prices may have arbitrage opportunities, but these are assumed to be discovered quickly and exploited such that the opportunity disappears. Nonetheless, by comparing market prices against those defined by a pre-specified pricing kernel, one could identify potential arbitrage opportunities. Specifying the pricing kernel requires state prices to be set for all states, of which an infinite number could exist. Ross provides a simple and practical solution to the problem of determining state prices in his development of the *Arbitrage Pricing Theory* (APT).

In the APT, returns are expressed as a linear combination of contemporaneous factors which represent undiversifiable sources of risk. Without loss of generality, we can consider the case where payoffs are returns $\mathbf{R}$, and variation is determined by a single factor, f. By no-arbitrage, the price of all payoffs is determined by the price of returns. In the APT, the returns matrix has an approximate low-rank representation:

$$\mathbf{R} = \mathbf{1}\big(\mathbb{E}[\boldsymbol{r}]\big)^\top + \boldsymbol{f}\boldsymbol{\beta}^\top + \epsilon \tag{2.40}$$

where ϵ is an orthogonal matrix representing the idiosyncratic variation of each asset.
Assuming no-arbitrage, the price of the returns must be the same as the price of the factors:

$$\boldsymbol{q}^\top \mathbf{R} = \boldsymbol{q}^\top \mathbf{1}\left(\mathbb{E}[\boldsymbol{r}]\right)^\top + \boldsymbol{q}^\top \boldsymbol{f}\boldsymbol{\beta}^\top + \boldsymbol{q}^\top \epsilon \tag{2.41}$$

$$\mathbf{1}^\top = \frac{1}{r_f}\mathbb{E}[\boldsymbol{r}]^\top - \frac{\psi}{r_f}\boldsymbol{\beta} \tag{2.42}$$

where $\boldsymbol{q}^\top \boldsymbol{f} = -\frac{\psi}{r_f}$, and ψ is defined as the scaled price of a factor. The price of the idiosyncratic risk is assumed to be zero. In other words, investors are not compensated for taking on additional idiosyncratic risk. This is justified if it is assumed that every payoff can be attained by a well-diversified portfolio. If this is the case, idiosyncratic variation will tend towards zero, while only factor risk remains. Applying the no-arbitrage assumption, if every payoff can be replicated by a combination of factors, only factors are priced.

The APT is usually expressed as a statement about the expected returns. This involves rearranging of the terms above:

$$\mathbb{E}[\boldsymbol{r}] - r_f \mathbf{1}^\top = \psi\boldsymbol{\beta} \tag{2.43}$$

The factor can be expressed as a return by projecting the factor onto the set of returns. The correponding return is called a factor-mimicking return and is determined by a factor-mimicking portfolio.If the factor is expressed as a return, then the price of the factor is the factor risk premium $\mathbb{E}[r_m] - r_m$. Finally, it can be shown that a factor model for payoffs is equivalent to a factor model for the pricing kernel. Thus, factors are used to explain covariation between assets, and through the assumption of no-arbitrage, are able to explain the cross-section of expected returns. APT allows for asset prices to be defined by variables that are exogenous to the set of assets.

2.4.3 Extending the APT

The principle of no-arbitrage has been extended to allow for an incomplete market [57], a continuous state-space [58],and continuous trading [59]. These generalisations of Ross's original formulation are not material for the interpretation of the empirical findings of the market discussed in section 2.4.5. As such, we refrain from further discussion concerning no-arbitrage in its generality.

The original APT was formulated as a single-period linear factor model where market prices are consistent with the principle of no-arbitrage. The linearity and time-horizon of the model can be discarded without contravening the no-arbitrage requirement. The multiperiod and nonlinear extensions are crucial for assessing the empirical evidence of a pricing model. These extensions are explored below.

The nonlinear APT

A factor model is emprically evaluated by testing if the specified factors are sufficient to explain the observed returns for each asset. An *alpha*, α, is defined as an average return that is not explained by the factor risk premia of the specified model:

$$\boldsymbol{\alpha} = \mathbb{E}[r] - R_f \mathbf{1}^\top - \lambda\boldsymbol{\beta} \tag{2.44}$$

An empirical discovery of a statistically significant alpha can be used to motivate the rejection of a factor model. An alpha can occur if errors are sufficiently large such that they are not diversifiable. The price of such errors needs to be incorporated into the pricing kernel by including additional

factors.

Alternatively, for a correctly specified factor model, an alpha can represent a statistical arbitrage opportunity. An investor can form a zero-risk (and hence zero-cost) portfolio that earns a positive payoff on average. If the zero-cost portfolio is well-diversified, such that idiosyncratic variation is negligible, the payoff is approximately constant and represents a standard arbitrage opporunity (as defined above). If a well-diversified portfolio can not be constructed, the idiosyncratic variation presents a risk to the investor. However, if the alpha persists, and the zero-cost portfolio can be repeatedly invested in, the variation of the average payoff will tend towards zero. This is known as a *statistical arbitrage*.

Bansal and Viswanathan [60] recognize that the linear description of the APT does not contain any economic justification, and explore the possibility of a nonlinear factor specification. A nonlinear pricing kernel provides an additional interpretation for the existence of an alpha. An alpha associated with a linear pricing kernel may indicate that payoffs are nonlinear functions of risk [19].

A nonlinear mapping from factors to returns is still consistent with no-arbitrage as it still specifies a unique state-price vector. A nonlinear specification would allow for derivative contract, which may be nonlinear in the underlying factors, to be priced. In contrast, using a linear factor model to price a derivative security can result in an arbitrage opportunity. While Bansal and Viswanathan's formulation allows for a general nonlinear function of the factor payoffs, nonlinearity is often attributed to time variation in the factor exposures, and the factor premia [61–63].

The conditional APT

Suppose, for simplicity, that the random payoffs for every asset are defined over a time horizon of T periods. At time $T-1$, for each node, the model set-up is the same as the single-period scenario from before. Thus, for each node it is possible we can define a conditional stochastic discount factor m_T, with discount rate that depend on the state-variable (ie. different discount rates for each node), that maps the payoffs to conditional no-arbitrage prices. Mathematically, this is expressed as:

$$P_{i,T-1} = \mathbb{E}[m_T X_{i,T} | \Phi_{T-1}] \tag{2.45}$$

where Φ_{T-1} represents the set of realized state variables[3] (z_{T-1}) at time $T-1$.

The conditional prices defined at each node at $T-1$ are then combined with dividend income ($D_{i,T-1}$) from the asset to form the payoffs at $T-1$, $X_{i,T-1} = P_{i,T-1} + D_{i,T-1}$. A conditional discount factor can be defined to map the $T-1$ payoffs to no-arbitrage prices for each node at time $T-2$. Continuing this way, conditional single period discount factors at each time period can be defined such that there no arbitrage opportunities exist over any single-period sub-tree. The absence of an arbitrage opportunity over every single-period sub-tree implies that no multiperiod arbitrage opporunties exist over the entire tree.

Thus, using the conditional single period discount factors, a conditional multiperiod stochastic discount factor at arbitrary time t can be defined by:

$$M_t = \prod_{i=1}^{i=t} m_i$$

[3] See section 2.2.4 for discussion regarding the impact of predictive information on portfolio choice

If $t = T$, then the multiperiod discount factor is the unconditional discount factor, and the information set is the empty set, $\Phi_0 = \emptyset$. In the single-period case, the unconditional and conditional discount factors are the same.

A conditional APT model can be derived by expressing the conditional pricing kernel as linear combination of factors, where the combination is state-dependent. Equivalently, the conditional expected return is determined by a conditional factor exposure, and a conditional factor premium. The factor exposures and the factor premia can vary across each node, according to the state-variable. Thus, the conditional APT model extends the single-period unconditional APT to incorporate predictive state-variables in the pricing kernel. As with the unconditional single-period case, evidence of predictability beyond what is incorporated in the pricing kernel does not indicate an arbitrage opportunity. A investor with superior information than is incorporated in the market pricing kernel can only stand to gain from exploiting the mispricing in absolute (as opposed to relative) terms.

Statisitical testing of a model requires that a sufficiently large sample can be drawn such that credible inferences can be made [14]. This is generally not possible when the distribution for the population are varies through time. Thus, conditional asset pricing models are difficult to test and unconditional formulations of the model are required. Single period discount factors can be composed to form the unconditional discount factor. Similarly, by composing conditional pricing models, an unconditional model can be derived. Hansen and Richard [61] pointed out that a single factor conditional model does not imply a single factor unconditional model. This is demonstrated below:
Suppose that we have a single factor conditional factor model expressed in terms of conditional expected returns:

$$\mathbb{E}_t[r_{t+1}] = r_{f,t+1} + \beta_t \lambda_t \tag{2.46}$$

Taking expectations over the informtion set to get the unconditional expectation:

$$\begin{aligned} &\mathbb{E}\big[\mathbb{E}_t[r]\big] = \mathbb{E}[r_f] + \mathbb{E}[\beta_t \lambda_t] \\ &\implies \mathbb{E}[r] = \mathbb{E}[r_f] + \mathbb{E}[\beta_t]\mathbb{E}[\lambda_t] + \mathrm{Cov}(\beta_t, \lambda_t) \end{aligned} \tag{2.47}$$

The unconditional model has an extra term to account for the covariation between the factor exposure and the factor premium. In the multifactor case, the covaration term needs to be defined by a covariance matrix to account for cross-correlations between different factors.

Having discussed the extensions to the APT, the main argument of this section can now be stated. A conditional factor model implies a nonlinear unconditional factor model. The nonlinearity is still compatible with the principle of no-arbitrage [60]. Although the unconditional model retains an affine structure, the expected return for an asset is explained by a time-invariant exposure to the factor as well as a component which accounts for time-varying exposure to the factor [61]. The additional component that captures the impact of time-variation related to the factor is ultimately due to the time-variation in the investment opportunity set [64].

While exploitable predictability is a contentious issue, there seems to be agreement in literature about the existence of predictive variables that covary with the investment opportunity set through time [31, 32]. As a result, nonlinearity in an asset pricing model is usually modelled through a parametric specification, where factor exposures and factor premia are determined by predictive information [63]. Since the predictive information varies over time, factor exposures are unconditionally nonlinear. Although there is no widely accepted theory specifying how factors relate to predictive information, this specification for nonlinearity is popular since it has strong

economic intuition. Furthermore, as Ghysels [65] points out, incorrect nonlinear specifications can result in worse models than the linear counterpart. Nonparametric methods can provide strong in-sample fits, but have poor performance out-of-sample.

2.4.4 Identifying Factors

The content of no-arbitrage pricing is that expected returns are related to undiversifiable variation. APT does not provide causal explanations relating specific factors to expected returns. Since APT does not identify factors, factor choice is determined by statistical fit and economic intuition, both of which can be misleading if not carried out with great care. Empirically determined factors are particularly susceptible to data snooping. *Data snooping* occurs when a dataset is reused multiple times to test alernative models. As the number of tested models increases, so does the probability of finding a false positive result [13]. The statistical significance of such factors deterioriate significantly out-of-sample.

In any given study, the number of factors that failed statistical testing is almost never reported. Data snooping accross research articles can occur as additional testing on successful factors is performed on the same dataset that motivated the hypothesis. In a recent paper, Harvey, Liu, and Zhu [13] review the factor literature and test a total of 316 factors proposed in the top financial and economic journals. Their survey accounts for multiple testing through increasing the significance threshold to reject the null hypothesis that the factor is unable to explain the cross-section of returns. Their reults show that *more than half of the discovered factors are likely to be false discoveries*. Data snooping is frequently used as a justification to disregard empirical results purporting a statistically significant alpha, defined with respect to a thoeretically motivated pricing model [13, 66].

The prevalence of false results and the inability to distinguish between alternative factor specifications suggests that only theoretically motivated factors should constitute an asset pricing theory [25]. In this regard, the dominant approach of economists has been to construct models of equilibrium prices, where risk premia are an endogenous outcome of agent preferences. In general, an economic equilibrium is an idealised state of a market where all market participants engage in trades such that each participant maximises their individual utility functions and markets clear [15].

The most celebrated equilbrium asset pricing models are the Capital Asset Pricing Model (CAPM) [4–6], the Intertemporal Capital Asset Pricing Model (ICAPM) [67], and the Consumption-Based Capital Asset Pricing Model (CCAPM) [68]. The CCAPM discovers the source of risk premia to be covariance of asset returns with aggregate consumption growth. The ICAPM is derived from the intertemporal portfolio choice problem. As such, the risk premia discovered by the ICAPM are the market wealth and state variables. Similarly, the CAPM can be derived from the single-period mean variance problem, and an assets risk premium depends on its covariance with market wealth. The CCAPM subsumes the ICAPM and CAPM since an optimal aggregate consumption policy depends on aggregate wealth and state variables.

The link between no-arbitrage pricing and equilbrium pricing is reflected in the third statement of the FTAP (see Section 2.4.2): the absence of arbitrage implies the existence of a finite optimal demand for some agent that prefers more to less. Thus, a statement about market prices is equivalent to making a statement about investor preferences. The APT is referred to as a preference-free approach since it makes no statements about investor preferences beyond non-satiation. No-arbitrage does not even require that the investor is risk-averse.

In contrast, equilibrium approaches impose a stricter set of assumptions on investor preferences, allowing for more specific restrictions on asset prices than no arbitrage alone. These additional restrictions are incorporated in the pricing kernel, which has an exact form expressed in terms of investor preferences. An investor's discount factor immediately emerges from solving the first order conditions for the investors maximisation problem. However, the discount factor is specific to the investor, and is determined endogenously by the market price and payoffs. Additional assumptions must be made to characterise an aggregate preference function to determine prices endogenously .

The derivation of an equilibrium model may be based on the assumption of a *representative agent* [69]. The assumption means that aggregate utility can be modelled as if the market consisted of a single investor. This is justified mathematically if individual investors are assumed to have independent and identical linearly aggregable utility functions. This allows the aggregate utility function to be specified as a linear combination of individual utilities, where weights are determined by investor endowments. Another assumption that is required is *rational expectations* [70]. By including these assumptions, a market pricing kernel can be specified in terms of investor preferences. This is demonstrated in the appendix for the single-period case to derive the CAPM.

By considering the investors preferences, the CAPM discovers that the identity of the factor model as the aggregate wealth portfolio. Since all investors wealth is held in assets, the aggregate wealth portfolio is the same as the market portfolio. The market portfolio consisting of all assets (incl. bonds, property, options, insurance, etc.) is not observable. When valuing equities, the market portoflio is most commonly proxied by a market index. The ICAPM includes additional factors for each state variable, leading to a multifactor model. Once again, we are faced with the problem of identifying factors since the ICAPM does not specify what the state-variables should be. However, the potential factors are restricted to variables that have some predictive ability.

2.4.5 Predictability

The issue of predictability in asset returns has been central debate in academic finance, and has been organised around the concept of *Market Efficiency*. Jensen [71] defines market efficiency as "A market is efficient with respect to information set Φ_t if it is impossible to make economic profits by trading on the basis of information set Φ_t". The identification of a profitable opportunity requires that an investor has an information advantage over the average investor, such that the investor's information has not yet been incorporated into the market price. The assumption of investor rationality is a sufficient to ensure that market prices always reflect available information precluding the possibility of earning risk-adjusted excess returns.

Predictability of returns is a necessary but not a sufficient condition for a profitable information advantage. As discussed in Fama's review article [72], the predictive ability of a variable can be justified by rational expectations. A predictive variable that is able to explain cross-sectional may proxy for an undiscovered risk factor. A predictive variable that is able to predict future returns may reflect time-variation of expected returns that is commensurate with risk. The evaluation of an "economic profit" requires the use of a pricing model to determine whether profitability is attributable to risk-premia. Thus, the discovery of an excess risk-adjusted return can not be seperated from an incorrect specification of an equilibrium pricing model [72].

While initial empirical testing validated the CAPM, starting in the late 70s, researchers started to

discover relationships between firm attributes and stock returns across the cross-section of the market, whereby it was possible to systematically earn returns higher than that predicted by CAPM [73–75]. The predictive power of firm attributes was evidence enough to reject the unconditional CAPM as a sufficient model to explain market prices. Studies investigating the CAPM anomalies in the JSE [76–80] corroborate the international literature. In a series of papers [31, 32, 81], Fama and French collated the existing research on CAPM-related anomalies to empirically derive the now famous three-factor model. In addition to the market premium, the exprected returns also contained a return premium related to value as reflected in the book-value-to-price ratio, as well as a premium associated with firm size:

$$\mathbb{E}[r_i] = r_f + \beta_{M,i}(\mathbb{E}[r_M - r_f]) + \beta_{V,i}(\mathbb{E}[r_{HML}]) + \beta_{S,i}(\mathbb{E}[r_{SMB}]) \tag{2.48}$$

where $\mathbb{E}[r_{HML}]$ is the value factor premium, and $\mathbb{E}[r_{SMB}]$ is the size factor premium.

The Fama-French three-factor model (FF3F) was empirically derived, and contains seemingly ad-hoc additions to the aggregate wealth factor of the CAPM. However, the FF3F model can still be economically motivated as an unconditional version of a conditional CAPM, or as an ICAPM. If the predictive variables are interpreted as state-variables, they can be consistent with an equilibrium model that allows for time-varying expected returns. The predictive ability of these firm-attributes has been repeatedly observed over various time periods and across various markets [82] including the South African market [19, 20], adding to the plausibility that these factors reflect investor preferences. However, in the local literature there is disagreement about the best characteristic to proxy the value factor [78, 80, 83]. Local researchers provide the surprising result that, in the presence of a size and value factor, the market beta is negative [78, 79].

Fama and French [32] propose that the book-to-market and size effects are a proxy for an economic "distress" factor. Investors would require a premium to compensate for the risk associated with firms that are unable to withstand a prolonged reduction in earnings associated with economic depression. This argument is consistent with investor rationality and the efficient market hypothesis.

Two other prominent competing hypotheses challenged Fama and French's risk-based explanation, namely data-snooping and market inefficiency. The first argument was proposed by Lo and Mckinlay [66], who argue that the return anomalies are spurious and the result of unintentional but widespread data-snooping throughout the finance literature. The second argument is related to irrational investor behaviour. Lakonishok, Shleifer and Vishny [84] suggest that the glamour of high price-to-book-value (PTBV) firms attract investors, and that investors extrapolate the recent historical earnings growth rates of successful firms, overestimating future earnings. The net result of irrational investor actions place upward pressure on the prices of high price-to-book value firms, and downward pressure on low price-to-book value firms, creating the "value" return premium. Under the hypothesis of irrational investor behaviour, markets exhibit cross-sectional pricing that is not commensurate with risk.

Additional concurrent studies reinforced the view of market inefficiency, but refrained from making behavioural hypothesis to explain their results. Daniel and Titman [85] investigate whether the firm characteristic variables in the Fama-French model are able to explain returns beyond their ability to form a risk model. The results demonstrate that firms with similar characteristics experience similar average returns, irrespective of their covariance structure, suggesting the possibility of earning excess risk-adjusted returns or that returns are non-linear in the factors.

Ferson and Harvey [63] test a conditional version of the three factor Fama-French model and find evidence of time-varying factor loadings. However, they also find evidence of a time-varying alpha suggesting that time-variation in the factor loadings alone is not sufficient to explain the cross-section of returns. Haugen and Baker [82] construct optimized portfolios based on predictions using firm characteristics to assess their ability to generate economic profits. Their results show that the optimized portfolios provide greater average returns at a lower volatility than market cap-weighted portfolios for five different countries, suggesting that markets are not efficient.

Gebbie [20] conducts the same testing procedure as the Daniel and Titman study in the South African market, and produces corroborating results. An important finding from Gebbie's study is that alpha is generally non-zero for both characteristic and risk-based models. Following Daniel and Titman, both the characteristic models and risk-based models in Gebbie's study do not account for time-variation. Wilcox and Gebbie [19] conduct a related study comparing the performance of portfolios formed using time-varying versions of characteristic and risk-based models. Their results are consistent with Haugen and Baker's study, demonstrating that characteristics can be used to form portfolios with higher returns and lower volatility than a risk-based counterpart. Wilcox and Gebbie conclude by arguing that the pricing kernel is non-linear due to time-variation and which includes a time-varying alpha that can be identified by the use of firm characteristics. Their conclusion forms the motivation of this research.

2.4.6 Beyond Equilibrium and Efficiency

The EMH influenced a large portion of the literature throughout the 70s to the 90s and inspired new directions of study, but still remains an unsettled issue. Grossman and Stiglitz [86] point out an internal inconsistency in the argument for strongly efficient markets. If markets are perfectly efficient, investors would not be willing to pay the cost for private information. Thus, these investors would not trade on the basis of private information, resulting in the paradoxical state that private information is no longer incorporated into prices. Tests of semi-strong form efficiency relate to the discovery of lagged public information. Supporters of the EMH suggest that predictive variables are spurious or have risk-based explanations [1]. Weak-form efficiency tests for the existence of autocorrelations in returns. The discovery of positive autocorrelation in returns [33], commonly referred to as momentum, is justified by EMH supporters as being consistent with time-variation in expected returns.

Amongst the predominant opposing theories to the EMH are arguments based on psychology criticizing the notion of a rational investor. An investor, subject to various cognitive biases and defects, is unable to form the correct mathematical expectation of discounted cashflows, as required under the Rational Expectations Hypothesis. These these biases include: over-confidence, herding, psychological accounting, and mis-calibration of probabilities [17]. Behaviourists often cite examples of speculative bubbles as evidence against the EMH; prices are driven upwards by systematic irrational expectations.

Black [87] offers an alternative theory whereby he hypothesizes that a market may be efficient, but investors may be irrational. In his theory, he separates traders into noise traders and information traders. Information traders initiate trades based on correctly identifying mispricings between price and value. As information traders exploit mispricings, prices incorporate increasing amounts of the available information. Conversely, noise traders are mistakenly identify mispricings where there are none. As a result of noise trading, noise is injected into prices causing prices to deviate from value in a manner similar to a random walk. As a result, it is possible that the information required to correctly determine value is publicly available (ie. markets are efficient in

some sense), yet prices are not equal to value. Over time, pressure from information traders cause prices to revert to value.

The Adaptive Market Hypothesis (AMH) is a recent attempt to reconcile the differences in opinion between the behaviourists and the supporters of the EMH [10]. The AMH is influenced by the field of ecology and evolutionary psychology, viewing investors as adapting, competing agents. In contrast to the EMH where a rational investor makes optimal decisions based on all available information, the AMH specifies that investors are incapable of making the calculations necessary for optimal decisions. As such, investors specialize in certain strategies and sectors, and make sub-optimal decisions. This behaviour is deemed "bounded rationality". Within the evolutionary paradigm, investors can learn to optimise their behaviour through positive and negative reinforcement of their previous actions, eventually leading to approximately optimal systematic behaviour. However, investors may display maladaptive behaviour in response to a changing economic climate. This presents opportunities that can be exploited.

2.4.7 Hierarchical Causality in Financial Economics

Wilcox and Gebbie [18] consider the causal processes that generate prices in a market from a complex systems perspective. A complex system consists of a large number of sub-units that interact in unique ways, in response to multiple sources of causality, to produce outputs that exhibit emergent dynamics. Emergence is a type of nonlinear relation between input and outputs, typically associated with collective behaviour of biological systems. Emergence in the market can be viewed as the antithesis to the assumption of representative agent, since nonlinearity implies that a reductionist approach can not be used to deconstruct system outputs to understand component interactions. In a complex system, all variables are determined endogenously and outputs feedback into the system as a source of *top-down* causation. The complex systems approach is more consistent with real-world market functioning than the equilibrium models of economics.

Wilcox and Gebbie define a causal model whereby the market can approximated by a hierarchy of interacting models, with each model representing the aggregate behaviour of market agents acting at different time-scales. Six levels of hierarchy are defined. The lowest level of the hierarchy is composed of the individual traders generating buy and sell decisions. The second level of the hierarchy represents collective behaviours of traders. The third level is characterised by aggregate market interpretations of bottom-up information generated from the lower levels as well as from top-down information stemming from macro-economic sources. Level four consists of macro-economic variables, whilst levels five and six to the structure of the market. They argue that higher levels can operate fairly independently from lower levels since information from lower levels (*bottom-up* causation) is lost as it sequentially rises up the hierarchy.

This research is focussed on the layer three, which is inhabited by medium to long-term investors. Price behaviour at this level is modelled using the conditional APT to allow for multiple sources of causation. Prices can be determined endogenously as a consequence of aggregate preferences, or exogenously from the other levels of the hierarchical model. Information and risk are considered as related but distinct variables. This allows for mispricings to exist if information and risks is not fully reflected in asset prices. The hierarchical model incorporates various philosophies about market efficiency as special cases.

The functional form of the model is based on Ferson and Harvey's conditional specification of the Fama-French model [63], that allows for time-varying alpha. Mispricings are allowed for by

considering risk and expected return separately. The risk-based models are given below.
The stochastic process for return residuals is given by:

$$r_{i,t+1} - \mathbb{E}_t[r_{i,t+1}] = \sum_{p=1}^{P} \beta_{i,p,t} \left[r_{p,t+1} - \mathbb{E}_t[r_{p,t+1}] \right] + \epsilon_{i,t+1} \tag{2.49}$$

where there are P risk factors.

Expected returns permit an alpha and are given by:

$$\mathbb{E}_t[r_{i,t+1}] - r_{f,t} = \alpha_{i,t} + \beta_{i,p,t} \mathbb{E}_t[r_{p,t+1}] \tag{2.50}$$

alpha is determined implicitly in the risk model as being unexplained by risks. The information based models is specified by Haugen and Baker and is used to capture patterns in returns which can be predicted by lagged (bottom-up) information variables. The information model is expressed cross-sectionally as:

$$\mathbb{E}_t[r_{i,t+1}] = \mathbb{E}_t[\alpha_{t+1}] = \sum_{m=1}^{M} \mathbb{E}_t[\delta_{m,t+1}] \theta_{i,m,t} \tag{2.51}$$

where $\theta_{i,m,t}$ is the mth lagged information variable at time t, and there are M information variables. Ferson and Harvey [63] specify a time-varying model that allows for deviations from the risk-based model, where time variation is expressed by a conditional specification of factor exposures using lagged information variables. In addition to bottom-up information variables, top-down macroeconomic information variables are included.

Starting with a conditional specification of the risk-based model:

$$\mathbb{E}_t[r_{i,t+1}] - r_{f,t} = \alpha_{i,t}(z_{k,t}) + \beta_{i,p,t}(\theta_{i,m,t}, z_{k,t}) \mathbb{E}_t[r_{p,t+1}] \tag{2.52}$$

where conditional factor exposures are modelled as:

$$\beta_{i,p,t}(\theta_{i,m,t}) = b^0_{i,p} + \sum_{m=1}^{M} b^1_{i,p} \theta_{i,m,t} + \sum_{k=1}^{K} b^2_{i,p} z_{k,t} \tag{2.53}$$

and alphas are vary depending on top-down information, and are expressed as:

$$\alpha_{i,t} = a^0_i + \sum_{k=1}^{K} a_{i,k} z_{k,t} \tag{2.54}$$

where $z_{k,t}$ is the k$^{\text{th}}$ information variable at time t. The dependence of the factor exposures and the alpha on the information variables is assumed to be fixed through time. Variation in these parameters is occurs only though changes in the information variables. Substituting the expected return model into the conditional risk model above, we obtain:

$$\begin{aligned} r_{i,t+1} = a^0_i + \sum_{k=1}^{K} a_{i,k} z_{k,t} \\ + \sum_{p=1}^{P} \left(b^0_{i,p} + \sum_{m=1}^{M} b^1_{i,p} \theta_{i,m,t} + \sum_{k=1}^{K} b^2_{i,p} z_{k,t} \right) r_{p,t+1} + \epsilon_{i,t+1} \end{aligned} \tag{2.55}$$

2.5 System Identification for Automatic Control

2.5.1 Adaptive Control

The stochastic nature of asset returns implies that portfolio weights can drift away from the desired strategy if portfolio weights are not constantly rebalanced [87]. Thus, even in the simplified scenario where the distribution of asset returns is constant over time, portfolio construction requires dynamic rebalancing. The analysis of the intertemporal portfolio choice problem demonstrates that an utility-maximising investor dynamically constructs his portfolio, accounting for information contained in stochastic state-variables. A large body of empirical research has demonstrated the predictive ability of a number of variables to explain both the cross-sectional and temporal variation of asset returns [88]. The changing market dynamics motivates the implementation of an online model to support portfolio decisions. Real-time updates to controls upon the arrival of new data is referred to as *online* decision-making[4].

The field of Automatic Control is concerned with the analysis and design of automated systems that are able to make adaptive decisions to maintain consistent performance with reference to some criterion, in response to environmental changes. The portfolio choice problem share similarities to the prototypical applications of Automatic Control, and it may prove useful to leverage design and theoretical insights from the engineering literature when developing an automated portfolio system. In general, practical portfolio construction is a domain-specific application of model-based control. From the perspective of adaptive control, tactical asset allocation bears strong resemblence to an *Indirect Adaptive Control* (IAC) scheme.

The design of an adaptive controller is determined by the specification of the control law, and the mechanism to adapt control parameters. In an IAC scheme, controls are determined using predictions based on a model of the environment, where the model parameters are estimated online. It is most common for the functional form of the model to be specified prior to the operation of the system, so that only the current parameters of the system are estimated. The parameter estimation task is known in the control literature as *system identification*. More sophisticated approaches exist where the environmental model is learned over time, but this is not explored in this research. The online estimation and prediction steps constitutes the “adaptation mechanism" of the IAC. The focus of this chapter is the derivation of a general algorithm for the adaptive mechanism in an *Indirect Adaptive Control* (IAC) scheme.

2.5.2 Recursive System Identification

The objective of system identification is to estimate the current state of the system, which is represented as the parameters of the environment model, using historical data. Models used for adaptive control are most often recursively estimated. Recursive estimation techniques are characterized by updating parameters sequentially, using only the most recent parameter estimates of the model and the incoming data. This reduces the primary memory requirement for computation, allowing for many such algorithms to run in parallel.

[4]The terms *online* and *offline* are used to describe different classes of control systems [89]. An online controller adapts to new data as it becomes available. Controls are determined whilst the system is operational. In contrast, an offline control rule is determined using a fixed batch of data, before the system is operational.

A general expression for a recursive parameter estimator is given by:

$$\begin{aligned} \hat{\boldsymbol{\theta}}_t &= \hat{\boldsymbol{\theta}}_{t-1} + \mathbf{G}_t e_t \\ \text{where} & \\ e_t &= y_t - \hat{y}_{t|t-1} \end{aligned} \tag{2.56}$$

where $\hat{\boldsymbol{\theta}}_t$ is the parameter estimate at time t, $\mathbf{G}_t$ is the *Gain* and is defined below, e_t is the a-priori estimation error, y_t is the realised output signal, and $\hat{y}_{t|t-1}$ is the prediction for time time t made at time $t-1$.

The parameter estimate update equation is expressed as the sum of the previous parameter estimate plus some correction term. The previous parameter estimate contains all historical information relevant to the determination of the parameter at time t. The correction term updates the parameter based on the additional "information" provided by new data. The additional information is reflected by the prediction error. If there is no error in the prediction, all information about the process is already reflected in the previous parameter estimate. Strictly speaking, a prediction error may not represent the inaccuracy of a parameter estimate, but may just be the result of system noise. The prediction error is rescaled by the operator $\mathbf{G}$, which is referred to as the *Gain*. The Gain factor reflects how the parameter should be modified based on the prediction error. The Gain is determined by various design choices in the problem specification, including the choice of system model and the choice of objective function

The parameter recursion expression can be directly derived from the corresponding offline situation. When the batch estimate can be analytically derived, it is possible to derive a recursive estimator which provides the exact same parameter estimates when provided with the same data. When the optimization problem has no closed-form solution, offline solutions are approximated using numerical algorithms which updates parameters by iterating through entire dataset numerous times. In contrast, the recursive estimator only passes through the entire dataset once, and iteratively updates parameters at each new data point. In the machine learning terminology, the offline solution is analagous to a *batch gradient descent* for finding the optimal parameters, whilst the online estimator is a second-order method analagous to *stochastic gradient descent*.

Ljung and Soderstrom [89] develop a general framework for recursive algorithms that have the objective of minimising a function of prediction errors. Well-developed recursive estimation technologies such as the recursive least squares algorithm, and the Kalman filter, can be identified as special cases within the framework. These cases are the result of differing design choices. The derivation of the general algorithm is provided below, albeit with some simplifications to condense the exposition.

We consider the general algorithm as a *prediction-error* method. A prediction-error identification method estimates system parameters by minimising prediction errors. Prediction-errors are calculated as the distance between model predictions and the true realisations. The structure of the model and the choice of distance metric are assumptions and are problem-specific. Guidance for setting these assumptions can be found in Ljung and Soderstrom [89]. The Prediction-error identification method can be formulated as [90]:

$$\hat{\theta}_t = \arg\min_{\boldsymbol{\theta}} \|e\| = \arg\min \sum_{t=1}^{T} \ell(e_t(\boldsymbol{\theta})) \tag{2.57}$$

$$\text{where} \quad e_t(\boldsymbol{\theta}) = \boldsymbol{y}_t - g_t(\boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{x}_{t-1}, \ldots, \boldsymbol{x}_1, \boldsymbol{y}_{t-1}, \ldots, \boldsymbol{y}_1) \tag{2.58}$$

where $\boldsymbol{\theta}_t$ is the parameter vector, $\boldsymbol{x}_t$ is the vector of system inputs, the norm is specified as a time-separable combination of single-period prediction-error loss functions $\ell_t(\boldsymbol{e}_t)$, and $g_t(\boldsymbol{\theta}, x_t, y_t)$ is the predefined system model.

Ljung and Soderstrom's general framework does not place restrictions directly on the admissible model set. However, the resulting objective function must satisfy various smoothness properties to ensure estimator convergence. The additional assumptions are detailed below. In addition, system identification requires that inputs are sufficiently exciting. *Persistent excitation* of inputs can be informally described as the condition that inputs and outputs have sufficient covariation. Lastly, recursive system identification procedures require that the input and output signals are specified prior to the operation of the controller.

The derivation of the recursive algorithm presented here is the simplified treatment provided in Soderstrom and Stoica [90]. This derivation retains the essential ideas of the original derivation found in [89], whilst easing the mathematical and notational burden.

The system identification problem for the determination of possibly time-varying system parameters, is formulated as an exponentially weighted least-squares estimation problem:

$$\mathscr{J}_t(\boldsymbol{\theta}_t) = \frac{1}{2}\sum_{t=1}^{T} \lambda^{T-t} e_t^{\top}(\boldsymbol{\theta}_t)\mathbf{Q}e_t(\boldsymbol{\theta}_t) \tag{2.59}$$

where J is the objective function, Q is a known symmetric positive definite matrix that is constant over time, and $e_t(\theta)$ is the prediction-error at time t for parameter vector θ. The objective function is defined as the sum of time-invariant quadratic loss functions weighted by an exponential memory factor λ. The memory factor is included as an adjustment on the loss-function to allow for time-variation of the parameters. A state-space formulation can be used to specify a more sophisticated model for parameter dynamics.

It should be noted that while the objective function is quadratic in the prediction-error, it may not be a quadratic function of the parameters. This will depend on the functional form of the model. Furthermore, the weight matrix Q is used to scale errors to account for cross-correlations. It is assumed that the inputs signals reflect all auto-correlation in the outputs, therefore cross-autocorrelations are not explicitly accounted for in the loss function.

The recursive framework is based on the (multivariate) Newton-Rhapson iterative method to numerically find the roots of a function. In the recursive formulation, the parameter update at each time-step is equivalent to an Newton-Rhapson iteration. To ensure convergence to the true solution, additional assumptions are required:

A 1. *Assume that the parameter $\boldsymbol{\theta}_t$ is located in a sufficiently small neighbourhood of the previous parameter $\boldsymbol{\theta}_{t-1}$.*

A 2. *Assume that the parameter estimate $\hat{\boldsymbol{\theta}}_{t-1}$ is optimal at time $t-1$*

The implication of the first assumption is that the optimum of the objective function can be approximated by a second-order Taylor series expansion centered at $\hat{\boldsymbol{\theta}}_{t-1}$:

$$\begin{aligned}\mathscr{J}_t(\boldsymbol{\theta}_t) \approx \mathscr{J}_t(\hat{\boldsymbol{\theta}}_{t-1}) + \nabla_{\boldsymbol{\theta}}^{\top}\mathscr{J}_t(\hat{\boldsymbol{\theta}}_{t-1})(\boldsymbol{\theta}_t - \hat{\boldsymbol{\theta}}_{t-1}) \\ + \frac{1}{2}(\boldsymbol{\theta}_t - \hat{\boldsymbol{\theta}}_{t-1})^{\top}\mathbf{H}(\mathscr{J}_t(\hat{\boldsymbol{\theta}}_{t-1}))(\boldsymbol{\theta}_t - \hat{\boldsymbol{\theta}}_{t-1})\end{aligned} \tag{2.60}$$

where $\nabla_{\boldsymbol{\theta}}$ respresents the grad operator, and $\mathbf{H}(\mathscr{J}_t(\boldsymbol{\theta}_t))$ is the Hessian matrix of the objective function at time t. For notational convenience, the following quantities are defined to represent

the Hessian matrix and the gradient vector respectively, evaluated using the optimal parameter estimate:

$$\hat{\mathbf{H}}_{t|t-1} \equiv \mathbf{H}(\mathscr{J}_t(\hat{\boldsymbol{\theta}}_{t-1})) \tag{2.61}$$

$$\hat{\boldsymbol{\psi}}_{t|t-1} \equiv \left[\nabla_{\boldsymbol{\theta}} \mathscr{J}_t\right]_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}_{t-1}} \tag{2.62}$$

The approximated objective function is a quadratic function that is expressed in the form $f(\boldsymbol{x}) = c + \boldsymbol{b}'\boldsymbol{x} + \boldsymbol{x}'\mathbf{A}\boldsymbol{x}$. The minimum of quadratic function f is attained at $x = \mathbf{A}^{-1}\boldsymbol{b}$. Thus, the optimal parameter estimate at time t is given by:

$$\hat{\boldsymbol{\theta}}_t = \hat{\boldsymbol{\theta}}_{t-1} - \hat{\mathbf{H}}^{-1}_{t|t-1}\hat{\boldsymbol{\psi}}_{t|t-1} \tag{2.63}$$

To complete the recursive formulation, recursive expressions for the gradient vector and the Hessian matrix of the objective function must be derived. From Equation 2.59, the objective function can be expressed recursively as:

$$\mathscr{J}_t(\boldsymbol{\theta}) = \lambda \mathscr{J}_{t-1}(\boldsymbol{\theta}) + \frac{1}{2} e_t^{\mathsf{T}}(\boldsymbol{\theta})\mathbf{Q}e_t(\boldsymbol{\theta}) \tag{2.64}$$

Thus, the gradient vector is given by:

$$\hat{\boldsymbol{\psi}}_{t|t} = \lambda\hat{\boldsymbol{\psi}}_{t-1|t} + \boldsymbol{e}^{\mathsf{T}}_{t|t}\mathbf{Q}\boldsymbol{\eta}_{t|t} \tag{2.65}$$

where $\boldsymbol{\eta}_{t|t}$ describes the gradient matrix of the prediction error evaluated using the time t parameter and is given by:

$$\boldsymbol{\eta}_{t|t} = \nabla_{\boldsymbol{\theta}}\boldsymbol{e}_{t|t} \tag{2.66}$$

Taking the grad of the gradient vector once more, provides an expression for the Hessian matrix

$$\hat{\mathbf{H}}_{t|t} = \lambda\hat{\mathbf{H}}_{t-1|t} + \boldsymbol{\eta}^{\mathsf{T}}_{t|t}\mathbf{Q}\boldsymbol{\eta}_{t|t} + \boldsymbol{e}_{t|t}\mathbf{Q}\left[\nabla_{\boldsymbol{\theta}} \otimes \boldsymbol{\eta}_{t|t}\right]^{\mathsf{T}} \tag{2.67}$$

where the operator $\otimes$ is the Kronecker product.

The expressions above fall short of being recursive since the previous estimates (at time $t-1$) of the gradient and Hessian are evaluated using the current parameters (at time t). Approximations derived from the assumptions enable recursive calculation. From Assumption 2 (A.2), it follows directly that the derivative of the objective function evaluated at the optimal point is equal to the zero vector:

$$\hat{\boldsymbol{\psi}}_{t-1|t-1} = \mathbf{0} \tag{2.68}$$

Thus, by incorporating A.2, and evaluating Equation 2.65 at the optimal value of the previous estimate, we have:

$$\hat{\boldsymbol{\psi}}_{t|t-1} = \hat{\boldsymbol{\psi}}_{t-1|t-1} + \boldsymbol{e}^{\mathsf{T}}_{t|t-1}\mathbf{Q}\boldsymbol{\eta}_{t|t-1} \tag{2.69}$$

$$\hat{\boldsymbol{\psi}}_{t|t-1} = \boldsymbol{e}^{\mathsf{T}}_{t|t-1}\mathbf{Q}\boldsymbol{\eta}_{t|t-1} \tag{2.70}$$

If we consider Assumption 1 (A.1), that the optimal estimate is close to the previous optimal estimate, in the scenario where the estimates are converging towards the true parameter value, it will imply that the prediction error sequence will approximate a white noise sequence. This will occur since at the true value the prediction error will only be composed of system noise. This means that after a sufficient number of observations, the prediction error at time t will have a mean of zero be independent of the data previous to t. Thus, the following approximation can be made for large t and large λ:

$$\boldsymbol{e}_{t|t-1}\mathbf{Q}\left[\nabla_{\boldsymbol{\theta}} \otimes \boldsymbol{\eta}_{t|t-1}\right]^{\mathsf{T}} \approx 0 \tag{2.71}$$

Finally, using assumption 1, we can approximate the Hessian at the current optimum, using the previous optimal parameter estimate:

$$\hat{\mathbf{H}}_{t-1|t-1} = \hat{\mathbf{H}}_{t-1|t-2} \tag{2.72}$$

Thus, substituting the approximation into the Hessian equation 2.67, and evaluating the expression at the optimal value of the previous estimate :

$$\hat{\mathbf{H}}_{t|t-1} = \lambda \hat{\mathbf{H}}_{t-1|t-2} + \boldsymbol{\eta}_{t|t-1}^{\top} \mathbf{Q} \boldsymbol{\eta}_{t|t-1} \tag{2.73}$$

In summary, using these approximations, the recursive parameter estimator is given by the following equations:

$$\hat{\boldsymbol{\theta}}_t = \hat{\boldsymbol{\theta}}_{t-1} - \hat{\mathbf{H}}_{t|t-1}^{-1} \boldsymbol{e}_{t|t-1}^{\top} \mathbf{Q} \boldsymbol{\eta}_{t|t-1} \tag{2.74}$$

where

$$\hat{\mathbf{H}}_{t|t-1} = \lambda \hat{\mathbf{H}}_{t-1|t-2} + \boldsymbol{\eta}_{t|t-1}^{\top} \mathbf{Q} \boldsymbol{\eta}_{t|t-1} \tag{2.75}$$

The recursive parameter equation (2.74) defines the terms in the general equation (2.56). The Gain factor is determined by $\mathbf{G}_t = \hat{\mathbf{H}}_{t|t-1} \boldsymbol{\eta}_{t|t-1} \mathbf{Q}$. Thus, the gain is equivalent to the Newton-Rhapson search direction. Accordingly, the size of the prediction error determines the step size in the parameter update, whilst the gain determines the direction of the update in the parameter space. The gain is determined by the cost function, through the Hessian matrix and the matrix $\mathbf{Q}$, and by the gradient vector of the prediction error. The functional form of the system model enters the Gain through the cost function and the gradient vector. For a more general loss function, $\ell(\boldsymbol{e}_t)$, and model structure, $\mathscr{M}(\boldsymbol{x_t}, \boldsymbol{\theta}, \boldsymbol{\epsilon_t})$, the recursive estimator equation can be generalized to:

$$\hat{\boldsymbol{\theta}}_t = \hat{\boldsymbol{\theta}}_{t-1} + \gamma_t \hat{\mathbf{H}}_{t|t-1}^{-1} \boldsymbol{\eta}_{t|t-1} \boldsymbol{\zeta}_t \tag{2.76}$$

where γ_t is referred to as the gain sequence, and $\boldsymbol{\zeta}_t = \nabla_e \ell(e)$ is a vector of the derivative of the loss function with the error. The gain sequence describes how the recursive algorithm tracks a time-varying parameter. In the recursive estimator, the gain seqeunce is defined solely by the memory factor. If the gain sequence tends towards zero as more data becomes available, the parameter will converge to a single number. Consequently, to track a time-varying system, the gain sequence must be specified such that it does not converge to zero.

In the instance that the system model is linear and the loss function is quadratic, the recursive least squares (RLS) algorithm is recovered. The algorithm becomes exactly true, since the approximations (2.71 and 2.72) are exact. If the objective function is the minimization of the quadratic error over a single time period, the least-mean square algorithm is recovered. A Bayesian estimator for a linear model maximises the posterior distribution of the parameter, for a specified prior. The Bayesian Recursive Filter can be derived by sequentially updating the prior to incorporate the new data, and although it is not a prediction-error model, it is another special case of the general recursive estimator. Lastly, the Kalman Filter can be derived by specifying a full state-space model that includes stochastic state innovations and system noise. For the interested reader, full derivations can be found in the text [89].

Computational Aspects

The form recursive estimator in Equation (2.74) is not implemented in practice since the estimator involves inverting a matrix which is computationally burdensome, especially if the model includes a large number of parameters to be estimated. A computationally efficient estimator can be derived by recursively calculating the inverse of the Hessian matrix directly, rather than updating

and then inverting the matrix. By applying Woodbury's matrix inversion identity (see appendix A.5), the recursive formula for updating the inverse Hessian is given by:

$$\hat{\mathbf{H}}_{t|t-1}^{-1} = \frac{1}{\lambda}\left\{\hat{\mathbf{H}}_{t-1|t-2}^{-1} + \hat{\mathbf{H}}_{t-1|t-2}^{-1}\boldsymbol{\eta}_{t|t-1}\left[\lambda\mathbf{Q}^{-1} + \boldsymbol{\eta}_{t|t-1}^{\mathsf{T}}\hat{\mathbf{H}}_{t-1|t-2}^{-1}\boldsymbol{\eta}_{t|t-1}\right]\right\} \tag{2.77}$$

Another important consideration is the choice of intial values for the estimator. In order to calculate estimates from the first time period, intial values for the parameter $\boldsymbol{\theta}_0$ and the (inverse) Hessian, $\hat{\mathbf{H}}_{1|0}$, need to be specfied. Initial values are required since the system of equations to be solved will be underdetermined until the number of time periods is equal to the number of parameters to be estimated. Specifying initial values for a prediction-error recursive algorithm is synonymous with specifying a prior distribution for a sequential Bayes Filter [89]. Correspondingly, the accuracy with which the intial values reflect the current state of system will strongly influence the rate of convergence, but not the limiting value in a stable algorithm.

In the case of the RLS estimator, specifying the intial values is computationally equivalent to specifying a prior distribution for the parameter, and can be interpreted as a Bayesian estimator. In this case, the initial inverse Hessian matrix is equivalent to the precision matrix of the prior estimate. This can be seen if we consider the offline estimator for the parameters of a single output signal and multiple input signals (which in the case of RLS provides exactly the same online estimates):

$$\hat{\boldsymbol{\theta}}_T = \left[\hat{\mathbf{H}}_{1|0} + \mathbf{X}^{\mathsf{T}}\mathbf{X}\right]^{-1}\left[\hat{\mathbf{H}}_{1|0}\boldsymbol{\theta}_0 + \mathbf{X}^{\mathsf{T}}\boldsymbol{y}\right] \tag{2.78}$$

where $\mathbf{X}$ is the matrix containing the historical values for the input signal, and $\boldsymbol{y}$ is vector containing historical values of the output signal, from $t = 1,\ldots,T$. Accordingly, as the norm of the Hessian increases, the initial value becomes increasily uninformative. As less weight is placed on the prior, the sequential estimator approaches the offline equivalent.

Having derived an expression for the recursive estimator, together with a recursive expression for the calculation of an inverse matrix, this section can be summarised by providing the complete algorithm of a general recursive prediction-error algorithm for a quadratic loss function and general model.

Algorithm 1: Online Parameter Estimation (OPE)

Function OPE($\boldsymbol{\theta}_0, \mathbf{H}_0^{-1}, \lambda, \mathbf{X}_t, \mathbf{y}_t$)

Input : initial parameter estimate, $\boldsymbol{\theta}_0$
inverse Hessian matrix prior, $\mathbf{H}_0$
forgetfullness factor, λ
input signal matrix until t, $\mathbf{X}_t$
output signal matrix until t, $\boldsymbol{y}_t$

Output: beta estimate at t, $\boldsymbol{\theta}_t$
predicted output at t+1, y_{t+1}

begin

Initialise $\mathbf{H}_0^{-1}$ and $\boldsymbol{\theta}_0$

calculate predicted value, $\hat{y}_t = g_t(\boldsymbol{\theta}, x_t, y_t)$

calculate a priori prediction error, $e_t = y_t - \hat{y}_t$

update inverse Hessian matrix,

$$\hat{\mathbf{H}}_{t|t-1}^{-1} = \frac{1}{\lambda}\{\hat{\mathbf{H}}_{t-1|t-2}^{-1} + \hat{\mathbf{H}}_{t-1|t-2}^{-1}\boldsymbol{\eta}_{t|t-1}\left[\lambda\mathbf{Q}^{-1} + \boldsymbol{\eta}_{t|t-1}^{\top}\hat{\mathbf{H}}_{t-1|t-2}^{-1}\boldsymbol{\eta}_{t|t-1}\right]\}$$

calculate gradient matrix, $\boldsymbol{\eta}_{t|t-1}$

update Gain matrix, $\mathbf{G}_t = \hat{\mathbf{H}}_{t|t-1}^{-1}\boldsymbol{\eta}_{t|t-1}\mathbf{Q}$

update parameter estimate, $\hat{\boldsymbol{\theta}}_t = \hat{\boldsymbol{\theta}}_{t-1} + \mathbf{G}_t e_t$

end

end

The OPE algorithm recursively estimates model parameters. The OPE algorithm can optimise quadratic loss function of prediction-errors from a nonlinear model using an iterative second-order numerical approximation. The algorithm is applicable to sequential data. A single iteration of the approximation is performed for each datapoint. Iterations are performed online as data becomes available.

Chapter 3

An Online Framework for Factor-based Portfolio Control

3.1 Introduction

The problem of *Online Portfolio Selection* relates to the design of intelligent algorithms that can learn optimal decision rules for wealth allocation. Online portfolio selection algorithms are buy-side algorithms that sequentially determine optimal portfolios, typically to maximise terminal wealth over a finite multi-period horizon. The algorithms have largely been developed as a machine learning problem, where portfolio construction is characterized by exploiting information from historical data without reference to normative assumptions. Li and Hoi [91] provide the most recent survey of the online portfolio selection literature. See [92] for a more recent application on the JSE.

In contrast, financial economists have investigated how to maximise an investor's multiperiod risk-adjusted return, using the mathematical framework of stochastic dynamic programming [28]. Investigation has been largely theoretical, since an analytically or numerically tractable solution that can be applied to a realistically-sized problem is not well-known. In practice, the solution of multi-period portfolio construction problem is approximated by sequentially constructing single-period mean-variance efficient portfolios [7]. This thesis constructs an algorithmic framework for online active quantitative portfolio management from first principles, where the design is based on central theoretical and empirical results in finance.

The main body work of this research is presented in this section. Concretely, the research develops an algorithmic framework that recursively approximates the ex-ante mean-variance efficient portfolio; where its dynamics are consistent with the complex hierarchical theory presented by Wilcox and Gebbie [18]. This framework describes an algorithmic strategy that earns rational risk premia, whilst aiming to exploit temporary mispricings resulting from the market's delayed assimilation of information. The online aspect is achieved through recursive exponentially weighted estimation. The framework is capable of extension to incorporate more advanced estimation techniques to improve prediction and portfolio construction.

The development of automated mean-variance strategy requires consideration about a number of related theoretical concerns that have arisen in the empirical literature. These relate to the difficulty of learning the system, which ultimately stems from a low signal to noise ratio. Epistemological uncertainty manifests as incorrect predictions, which are amplified and propagated to portfolio controls. The pertinent theoretical and practical concerns are briefly reviewed below:

3.1.1 Theoretical Limitations and Practical Challenges

- **Model Structural Uncertainty**: Asset pricing models are most commonly defined to express conditional expected returns as linear combination of prespecified factor premia. For models based on the Arbitrage Pricing Theory, the linear specification is often by convention or for simplicity, since nonlinear pricing models can still be consistent with no-arbitrage [60]. Observed nonlinearity in an unconditional model is often explained as time variation in risk factor premia and risk factor exposures [64], and modelled through correlations with predictive variables.

- **Model Variable Uncertainty**: Beyond the model structure and the determination of dynamics, knowledge of a full set of risk factors is not agreed upon. Harvey and Liu [13] survey a large number of proposed discoveries of factors and find that these discoveries are not statistically significant after adjusting the testing procedure for the effect of multiple testing. In addition, the identity of true predictive variables are not specified by theory. While there are some widely accepted predictive variables [31], discovery of predictive variables is often met with skepticism and, upon further inspection, are found to be the result of data mining [13, 66].

- **Spurious Correlation or Irrationality?**: Rational pricing theories posit that additional average returns are only possible from additional exposure to systematic risks. Temporary deviations from rational behaviour present the opportuntity to earn anomalous returns if they can be correctly identified. As additional return opportunities are exploited, expected returns will return to a rational relationship, and predictability will decrease. Maclean and Pontiff [11] show that out-of-sample predictability of a factor decreases substantially following the publication of its discovery. Potential correlations that are identified as spurious signals may be true temporary signals.

- **Backtest Overfitting**: The performance of a strategy over a historical sample may not be indicative of future performance, which is often inflated due to hidden risks, data-mining, and exhausted opportunities. Through an extensive search for profitable strategies over a historical dataset, an algorithmic investor is exposed to risk of false discoveries through extensive data-mining. [12, 93]. In-sample profitable strategies need to be adaptable to changing conditions.

A mean-variance portfolio requires a set of predicted expected returns and covariances. Asset pricing models can assist the investor in the task of making appropriate predictions. The inability to uniquely identify a complete set of factors, or the functional form of an asset pricing model, implies that parameter estimates will contain biases.The resulting predictions will have errors, which can still have practical value, but need to be carefully managed in the portfolio construction process.

Model uncertainty and parameter uncertainty is particularly problematic for a mean-variance optimizer which tends to amplify prediction errors [39], and is highly sensitive to changes in input parameters [42]. As a result, optimized portfolios are often highly concentrated, contradictory to a maximally diversified portfolio. Implementation issues stemming from parameter errors has traditionally been addressed through Bayesian methods which regularize estimates, improving estimation accuracy.

The customary criticism of Bayesian approaches for determining expected returns is the subjective nature of choosing a prior. Black and Litterman (BL) [52] were the first to use an economically justified prior for the purpose of active portfolio management. The BL approach

involves two steps: firstly, Black and Litterman introduce a “reverse-optimization" process to determine the implied return from the equilibrium portfolio (the market portfolio) weights. Subsequently, they apply Theil and Goldberger's (TG) mixed estimator [53] to blend active return and equilibrium return estimates, to construct a well-diversified active portfolio.

More recently, robust control methods have been investigated to account for model uncertainty in economic problems [94]. Robust control methods incorporate model uncertainty directly in the specification of the optimization, and avoid a characterizing uncertainty through subjective parameter distributions. The robust objective function determines the optimal controls for the worst case parameters, amongst a prespecified range of parmeters. Mathematically, this is captured by a max-min (or min-max) objective function. Robust control methods have also been applied portfolio choice problems: see references in [95].

This research follows the Bayesian approach to manage out-of-sample performance loss due to model and parameter uncertainty. Specifically, a slight adaptation to the BL approach is applied in this research, to blend active and systematic return estimates in proportion to the forecast accuracy of the respective models, where the forecast accuracy is measured online. Thus, the Theil mixed estimator serves as an intelligence mechanism that can adaptively allocate wealth to alternative strategies, in response to changing market efficiency and preventing persistent trading on spurious signals.

3.2 Problem Formulation

3.2.1 Objective

We develop an sequential investment process that aims to maximise the average risk-adjusted return over a finite multi-period horizon by dynamically solving a one-period mean-variance optimal portfolio. The risk-adjusted return earned by a strategy can be measured by the Sharpe Ratio, defined as:

$$SR = \frac{\mu}{\sigma} \tag{3.1}$$

where μ is the arithmetic average return in excess of the risk-free rate, and σ is volatility of the excess returns.

The mean-variance criterion is used to sequentially form an ex-ante single-period mean-variance efficient portfolio:

$$\max_{\boldsymbol{w}} \boldsymbol{w}^\top \boldsymbol{\mu} - \frac{\gamma}{2} \boldsymbol{w}^\top \Sigma \boldsymbol{w} \tag{3.2}$$

subject to the full investment constraint,

$$\boldsymbol{w}^\top \mathbf{1} = 1 \tag{3.3}$$

Since the true optimal frontier is not known in practice, active portfolio managers are assessed by their ability to produce excess returns over an observable benchmark portfolio. The benchmark strategies used in this research are the market-cap weighted portfolio (*the market (CAP) portfolio*) and the 1/n portfolio (*the naive diversification (ND) portfolio*). These benchmarks may appear too primitive to be compared with an automated process, but they are notoriously difficult to beat owing to poor predictions and error-maximisation [51].

The specified benchmarks are used for the purpose of performance measurement only: our goal is to approximate the unobservable mean-variance optimal portfolio, not to generate positive *active risk-adjusted returns*. Active returns are defined as returns in excess of a benchmark, whilst active risk is the variance of the active returns. The mean-variance portfolio construction process can be modified to target maximum risk-adjusted returns in excess of a benchmark. This is known as active risk and return optimisation. In practice, active risk-return asset allocation processes are common since investment managers are assessed and remunerated relative to a benchmark. It is worth noting that this is not ideal since an optimal portfolio in active risk-return space will be sub-optimal in total-risk return space if the benchmark is sub-optimal. While active portfolio optimisation is not implemented or discussed further, minor adjustments can be made to the presented algorithm in order to automate a benchmark-cognisant investment process.

3.2.2 Asset Price Dynamics

We consider asset prices to follow the model developed by Wilcox and Gebbie [18]. They specify a factor model for conditional returns, that includes a time-varying alpha. The time-varying alpha reflects changing market efficiency, while factors represent systematic risk sources consistent with arbitrage pricing theory.

Following Ferson and Harvey [63], Wilcox and Gebbie specify unanticipated returns as a time-varying linear combination of P systematic factors:

$$r_{i,t+1} - \mathbb{E}_t[r_{i,t+1}] = \sum_{p=1}^{P} \beta_{i,p,t}\left[r_{p,t+1} - \mathbb{E}_t[r_{p,t+1}]\right] + \epsilon_{i,t+1} \tag{3.4}$$

$$\mathbb{E}_t[\epsilon_{i,t+1}] = 0 \tag{3.5}$$

$$\mathbb{E}_t[\epsilon_{i,t+1} r_{p,t+1}] = 0 \tag{3.6}$$

where $r_{i,t+1}$ is the return of asset i in excess of the risk-free instrument, $r_{p,t+1}$ is the excess return on the p-th risk factor, $\beta_{i,p,t}$ is the i-th asset's exposure to the p-th risk factor, and $\epsilon_{i,t+1}$ is a stochastic disturbance term that is unrelated to the risk factors. The second and third equations represent standard moment restrictions for linear regression models. While our implementation utilises Fama and French [32] risk factors, the framework we develop is consistent with any (rational) factor model.

Expected returns are determined as a combination of systematic return premia, consistent with a conditional APT, but also includes a time-varying risk-free excess return, denoted by alpha $\alpha_{i,t}$:

$$\mathbb{E}_t[r_{i,t+1}] = \alpha_{i,t} + \sum_{p=1}^{P} \beta_{i,p,t}\mathbb{E}_t[r_{p,t+1}] \tag{3.7}$$

where,

$$\beta_{i,p,t} = b^0_{i,p} + \sum_{m=1}^{M} b^1_{i,p}\theta_{i,m,t} + \sum_{k=1}^{K} b^2_{i,p} z_{k,t} \tag{3.8}$$

$$\alpha_{i,t} = a^0_i + \sum_{m=1}^{M} a^1_p \theta_{i,m,t} + \sum_{k=1}^{K} a^2_{i,k} z_{k,t} \tag{3.9}$$

where $\theta_{i,m,t}$ represents the m-th information variable corresponding to the i-th stock, whilst $z_{k,t}$ represents macro-economic variables. These variables represent prespecified state-variables. Risk exposures are conditioned on the state-variables to model time-variation of these quantities. The dependence on the state-variables are determined by fixed parameters b^1, b^2, whilst a fixed

parameter b^0 denotes a time-invariant portion of the risk exposure, assuming that the state-variables are sufficient. The quantities a^0, a^1, a^2 can be similarly defined to model the temporal variation in alpha.

Parameters capturing the temporal variation need not be fixed. In our implementation, time-variation is incorporated through exponentially weighting observations in the estimation procedure. The exponential weighting also allows for recursive determination of all estimates. In addition, our implementation considers firm-specific characterisitics as state-variables. Firm characteristics have demonstrated predictive ability beyond their ability to proxy for unobserved risks [20, 85], indicating that predictive variables can be used to construct a strategy that earns excess returns [82]. However, the existence of an alpha may represent unmodelled risks.

3.2.3 Tactical Asset Allocation

Lee [7] motivates tactical asset allocation by decomposing a mean-variance optimal portfolio into three component portfolios: a global minimum variance (GMV) portfolio, a long-term strategic bets portfolio, and a short-term tactical bets portfolio. We use the same decomposition to seperate the ex-ante mean variance portfolio into a portfolio that earns systematic returns and a portfolio to earn active returns.

From section 2.3.1, a fully invested mean-variance portfolio has the solution:

$$\boldsymbol{w} = \left(1 - \frac{\mathbf{1}^\top \Sigma^{-1} \boldsymbol{\mu}}{\gamma}\right) \frac{\Sigma^{-1}\mathbf{1}}{\mathbf{1}^\top \Sigma^{-1}\mathbf{1}} + \left(\frac{\mathbf{1}^\top \Sigma^{-1} \boldsymbol{\mu}}{\gamma}\right) \frac{\Sigma^{-1}\boldsymbol{\mu}}{\mathbf{1}^\top \Sigma^{-1}\boldsymbol{\mu}} \tag{3.10}$$

This can be rewritten as the combination of the global minimum variance benchmark portfolio, and a zero-cost portfolio that is used to earn additional returns by taking on additional risk:

$$\boldsymbol{w} = \frac{\Sigma^{-1}\mathbf{1}}{\mathbf{1}^\top \Sigma^{-1}\mathbf{1}} + \frac{1}{\gamma} \frac{\Sigma^{-1}\left(\boldsymbol{\mu}\mathbf{1}^\top - \mathbf{1}\boldsymbol{\mu}^\top\right)\Sigma^{-1}}{\mathbf{1}^\top \Sigma^{-1}\mathbf{1}} \tag{3.11}$$

As Lee demonstrates, a zero-cost portfolio is necessarily composed of pairwise bets between assets. The size of the bets are proportional to the differences between the expected returns of the assets, scaled by the inverse covariance matrix and by the investor's risk tolerance.

By making the substitution, $\boldsymbol{\mu} = \boldsymbol{\pi} + \boldsymbol{\alpha}$, where $\boldsymbol{\pi}$ denotes the expected systematic return, and $\boldsymbol{\alpha}$ denotes the anomalous return, the expected return can be decomposed into systematic and non-systematic components:

$$\boldsymbol{w} = \frac{\Sigma^{-1}\mathbf{1}}{\mathbf{1}^\top \Sigma^{-1}\mathbf{1}} + \frac{1}{\gamma} \frac{\Sigma^{-1}(\boldsymbol{\pi}\mathbf{1}^\top - \mathbf{1}\boldsymbol{\pi}^\top)\Sigma^{-1}}{\mathbf{1}^\top \Sigma^{-1}\mathbf{1}} + \frac{1}{\gamma} \frac{\Sigma^{-1}(\boldsymbol{\alpha}\mathbf{1}^\top - \mathbf{1}\boldsymbol{\alpha}^\top)\Sigma^{-1}}{\mathbf{1}^\top \Sigma^{-1}\mathbf{1}} \tag{3.12}$$

resulting in a systematic bet portfolio, and an active tactical portfolio. The combined portfolio, $\boldsymbol{w} = \boldsymbol{w}_g + \boldsymbol{w}_s + \boldsymbol{w}_\alpha$, is ex-ante mean-variance efficient [1].

The GMV aims to achieve maximimum diversification, while the systematic bets portfolio is used to increase systematic exposure to increase risk-adjusted returns. The combination of the GMV and systematic bet portfolio is used to earn systematic risk premia in proportion to an investor's risk tolerance. The active bet portfolio is constructed to exploit short-term return anomalies,

[1] A full derivation is provided in appendix A.3.

where bet size is also proportional to risk tolerance.

Following the mean-variance derivation, the risk tolerance parameter for the active portfolio should be the same as the strategic portfolio. However, we can specify an active risk tolerance parameter, γ_A to determine the size of active bets. In the implementation, $\boldsymbol{\pi}$ is determined by Eqn. 3.7, where $\boldsymbol{\alpha} = 0$ to incorporate systematic returns only.

Active return estimates that are not benchmark-excess returns may incorporate the same information as the benchmark. In this case, positive tactical returns are not the result of superior predictions, but are rather due to additional exposure to the same risk premia in the benchmark portfolio. Using the TG mixed estimator, we shrink the active return estimates to the systematic returns to scale alpha estimates, and hence bet size, according to the realized predictive performance of the alpha model. Theil's mixed estimator is discussed further in the next section.

3.2.4 Regularization

Expected Return Estimation

Consistent with the BL approach [7, 52], we adopt a Bayesian view of uncertainty where expected returns have distributions that reflect our current knowledge. While a distributional assumption for the parameters are not explicitly required in Black and Litterman's derivation, their approach is consistent with the assumption that expected returns are normally distributed. Furthermore, we have two models of expected returns, as indicated by the systematic return model and the active return model. As a practical simplification, the covariance matrix of returns is assumed to known.

Following Black and Litterman's original derivation [52], we use the TG mixed estimator [53] to blend the expected return models to obtain the composite estimate $\boldsymbol{\mu}_{BL}$:

$$\boldsymbol{\mu}_{BL} = \left[\Omega_{\pi}^{-1} + \Omega_{\mu}^{-1}\right]^{-1} \left[\Omega_{\pi}^{-1}\boldsymbol{\pi} + \Omega_{\mu}^{-1}\boldsymbol{\mu}\right] \tag{3.13}$$

where Ω_{π} and Ω_{μ} represent the covariance hyper-parameter of the posterior distributions for the alternative models of the expected returns. The parameters $\boldsymbol{\pi}$ and $\boldsymbol{\mu}$ are defined as previously.

Since the estimator appears analagous to a convex combination of partial estimates, we can rewrite the mixed estimator as [96]:

$$\begin{aligned} \boldsymbol{\mu}_{BL} &= \boldsymbol{\pi} + \left(\left[\Omega_{\pi,t}^{-1} + \Omega_{\mu}^{-1}\right]\Omega_{\mu}^{-1}\right)(\boldsymbol{\mu} - \boldsymbol{\pi}) \\ &= \boldsymbol{\pi} + \Psi\boldsymbol{\alpha} \end{aligned} \tag{3.14}$$

where $\Psi = \left(\left[\Omega_{\pi}^{-1} + \Omega_{\mu}^{-1}\right]\Omega_{\mu}^{-1}\right)$.

The mixed estimate consists of the systematic return estimate and alphas that are scaled depending on the relative uncertainty between the systematic return estimate and the active return estimate. The greater the relative certainty in the systematic return estimate, the more the alpha diminishes. Thus, the mixed estimator has the effect of shrinking the active return estimates to the systematic return estimates.

We substitute these confidence-scaled alphas in place of the original alphas in the active tactical portfolio so that bet-size can be determined according to our confidence in the estimate. In our implementation, the uncertainty matrices corresponding to each return prediction is determined

recursively as the exponentially weighted covariance matrix of the realized prediction errors. This means that as signal strength decays, the tactical portfolio bets will become smaller.

Covariance Matrix Estimation

Using the unanticipated return model (3.4), the conditional[2] covariance matrix has the form:

$$\Sigma_{|\pi} = \mathbf{B}^{\mathsf{T}} \Sigma_f \mathbf{B} + \Sigma_\epsilon \tag{3.15}$$

where Σ is the covariance matrix for all assets, $\mathbf{B}$ is the matrix of factor exposures, Σ_f is the factor covariance matrix , and Σ_ϵ is a diagonal matrix of time-varying idiosyncratic variance.

Parameter uncertainty is an alternative source of variation that must be taken into account in the portfolio construction process. In the case that both the returns and the expectected returns are normally distributed, and the covariance is known, the resulting optimal portfolio rule is the same as the Markowitz rule, except that the covariance matrix is replaced with the marginal covariance matrix.

By applying the law of total variance formula [97],

$$Var(r) = \mathbb{E}\left(Var\left(r|\mu_{BL}\right)\right) + Var\left(\mathbb{E}[r|\mu_{BL}]\right) \tag{3.16}$$

the marginal variance can be determined, such that the marginal distribution of returns is given by:

$$r_t \sim \mathcal{N}(\boldsymbol{\mu}_{BL,t}, \Sigma_t + \Omega_{BL,t}) \tag{3.17}$$

where Ω_{BL} is the posterior uncertainty of the composite estimate of expected returns. Thus, the covariance matrix must change to $\Sigma + \Omega$ to account for uncertainty in expected returns [98]. In our implementation, the covariance matrix applied to strategic portfolio would be adjusted to account for strategic return estimate uncertainty, whilst the active tactical portfolio would use a covariance matrix adjusted by the uncertainty of the mixed active return estimate.

The derived distribution of marginal returns assumes that the covariance model is known, when in reality it is also estimated and subject to similar considerations as the expected returns estimate. The presented framework incorporates shrinkage of the covariance matrix prior to portfolio construction without an explicit reference to a probabilistic characteristation for parameter uncertainty. Shrinkage estimators can be used to improve the accuracy of sample covariance estimators, by shrinking the sample estimate to a more structured covariance estimator [44]. In our implementation, the shrinkage intensity is determined offline.

3.2.5 An Online Algorithm for Factor-Based Portfolio Control

This research presents the design of an online algorithmic framework to implement a factor-based strategy which attempts to earn excess risk-adjusted returns. Risk is defined in terms of the portfolio's exposure to prespecified risk factors. The algorithm is called the *Factor-Based Portfolio Control (FBPC) Algorithm*. The psuedo-code for the algorithm is presented in Algorithm 2. Control flow charts for the framework are shown in Figure 3.1 and Figure 3.2.

[2] Note that "conditional" refers to both conditional on the time t information and assuming that the expected returns are known

This system is an automated tool that dynamically constructs an ex-ante mean-variance optimal portfolio according to an information set that is updated online. The system is appropriate for use with daily, weekly or monthly sampled data for weekly to monthly portfolio rebalancing decisions. Exponentially weighted updates allow for estimation of the time-varying system. The primary uses for the framework are online algorithmic portfolio management and offline strategy backtesting.

At each timestep, the ex-ante MV frontier can be decomposed into the systematic risk-return frontier and active risk-return frontier. The sources systematic risk premia are prespecified offline. Active returns are temporary excess risk-adjusted returns that are defined to be anomalous according to the set of specified factors. Correspondingly, the selected optimal portfolio is decomposed into a benchmark portfolio that earns risk premia, and a tactical portfolio to earn excess risk-adjusted returns. The tactical portfolio accounts for changing signal strength by mixing model estimates in a Bayesian manner where posterior distributions are calibrated using time-varying performance criteria.

The framework is separated into clearly defined components that represent solutions to the predominant problems encountered in quantitative investing (see Section 3.1.1). The component-wise design allows for isolated improvements to be made without disrupting the working of the algorithm. The solutions presented are not necessarily optimal, and the components are not uniquely defined; alternative superior frameworks are indeed probable. Rather, the presented design represents pragmatic trade-offs between complexity, empirical performance, and theoretical purity required for a functioning automation.

While the workflow may appear complicated, it should be borne in mind that, ultimately, we are sequentially updating the predictive distribution for returns to solve for a mean-variance optimal portfolio. Over each time period, expected returns are determined by composing models in a quasi-Bayesian framework. Parameters are sequentially updated using exponential smoothing, but the framework does allow for a Bayesian updating scheme. Furthermore, expected return inputs are updated adaptively based on the relevant prediction error. Initial values can be specified from historical knowledge, or can be determined empirically during the initialisation period.

At time t, after the current estimates have been updated, predictions for the $t+1$ need to be made. Here, the framework does not restrict how the prediction is formed. For example, holt-winters exponential smoothing methods can be used to make more sophisticated predictions for systematic returns that are compatible with the arbitrage pricing thoery. General prediction functions are indicated by ϕ. Predictions are regularized, either through model averaging or explicit shrinkage. These “cleaned" predictions are used to construct portfolios.

System Operation

The control flow chart for the algorithm is shown in Figure 3.2 and Figure 3.1. The psuedocode corresponding to the entire portfolio construction algorithm is provided in algorithm 2.

Components in the framework are divided into offline and online components, depicted by grey and blue coloured blocks respectively. The online tasks are the focus of this research, and represent which tasks in the investment process are automated by this framework. Offline tasks are necessary to implement an quantitative investment process and typically involve statistical testing, which is outside the scope of the current automation. The offline tasks that are performed prior to the operation of the system include: data extraction and cleaning, signal and factor selection and transformation. Lastly, performance evaluation and testing is implemented as an

Algorithm 2: Factor-Based Portfolio Control (FBPC) Algorithm

Data: realised characteristics, $\tilde{z}_t$
realised factor returns, $\tilde{\boldsymbol{f}}_t$
realised returns, $\tilde{\boldsymbol{r}}_t$
prediction errors, $\tilde{e}_t$

Input
  memory factors, $\lambda_1, \lambda_2, \ldots, \lambda_l$
  risk tolerances, γ_s, γ_a
  portfolio contstraints $\{C_g(\boldsymbol{w}_g), C_s(\boldsymbol{w}_s), C_a(\boldsymbol{w}_a)\}$
end

Result: portfolio weights, $\{\boldsymbol{w}_{g,t}, \boldsymbol{w}_{s,t}, \boldsymbol{w}_{a,t}\}$

begin
  Initialise
    Factor exposure, $\hat{\mathbf{B}}_0$
    Factor premium, $\mathbb{E}[\hat{f}_0]$
    Systematic return model uncertainty, $\Omega_{\pi,0}$
    Characteristics payoffs, $\hat{\boldsymbol{\delta}}_0$
    Active return model uncertainty, $\Omega_{\mu,0}$
    Factor covariance matrix, $\hat{\Sigma}_{f,0}$
    Idiosyncratic risk matrix, $\hat{\Sigma}_{\epsilon,0}$
  end
  Update Parameter Estimates at t
    Factor exposure, $\hat{\mathbf{B}}_t = \hat{\mathbf{B}}_t + \mathbf{G}\tilde{e}_{\pi,t}$
    Factor premium, $\mathbb{E}[\hat{f}_t] = \lambda\mathbb{E}[\hat{f}_{t-1}] + (1-\lambda)\tilde{\boldsymbol{f}}_t$
    Systematic return model uncertainty, $\Omega_{\pi,t} = \lambda\Omega_{\pi,t-1} + \lambda(\tilde{\boldsymbol{e}}_t - \bar{e}_t)(\tilde{\boldsymbol{e}}_t - \bar{e}_t)^\mathsf{T}$
    Characteristics payoffs, $\hat{\boldsymbol{\delta}}_t = \varphi(\hat{\boldsymbol{\delta}}_{t-1}, \boldsymbol{z}_{t-1}, \boldsymbol{r}_t)$
    Active return model uncertainty, $\Omega_{\mu,t} = \lambda\Omega_{\mu,t-1} + \lambda(\tilde{\boldsymbol{e}}_t - \bar{e}_t)(\tilde{\boldsymbol{e}}_t - \bar{e}_t)^\mathsf{T}$
    Factor covariance matrix, $\hat{\Sigma}_{f,t} = \lambda\hat{\Sigma}_{f,t-1} + \lambda(\tilde{\boldsymbol{f}}_t - \bar{\boldsymbol{f}}_t)(\tilde{\boldsymbol{f}}_t - \bar{\boldsymbol{f}}_t)^\mathsf{T}$
    Idiosyncratic risk matrix, $\hat{\Sigma}_{\epsilon,t} = \lambda\hat{\Sigma}_{\epsilon,t-1} + (1-\lambda)\hat{\epsilon}\hat{\epsilon}^\mathsf{T}$
  end
  Predict Portfolio Inputs at $t+1$
    Systematic return, $\hat{\pi}_{t+1} = \phi_\pi(\hat{\mathbf{B}}_t, \mathbb{E}[\hat{f}_t])$
    Conditional covariance matrix, $\hat{\Sigma}_{t+1|\pi} = \phi_\Sigma(\hat{\mathbf{B}}_t, \hat{\Sigma}_{f,t}, \hat{\Sigma}_{\epsilon,t})$
    Shrinkage covariance matrix (GMV), $\hat{\Sigma}_{g,t+1} = \kappa_g\Sigma^* + (1-\kappa_g)\hat{\Sigma}_{t+1|\pi}$
    Unconditional systematic covariance matrix, $\hat{\Sigma}_{u,\pi,t+1} = \Sigma_{t+1|\pi} + \Omega_{\pi,t}$
    Shrinkage covariance matrix (systematic bets), $\hat{\Sigma}_{\pi,t+1} = \kappa_s\Sigma^* + (1-\kappa_s)\hat{\Sigma}_{u,\pi,t+1}$
    Active return estimates, $\hat{\mu}_{t+1} = \phi_\mu(\hat{\boldsymbol{\delta}}_t, \boldsymbol{Z}_t)$
    Mixed active and systematic return estimates, $\hat{\mu}_{bl,t+1} = \hat{\pi}_{t+1} + \Psi_t\hat{\alpha}_{t+1}$
    Mixed alpha estimate uncertainty, $\Omega_{bl,t} = \left[\Omega_{\pi,t}^{-1} + \Omega_{\mu,t}^{-1}\right]^{-1}$
    Unconditional active return covariance matrix, $\hat{\Sigma}_{u,\alpha,t+1} = \hat{\Sigma}_{t+1|\pi} + \Omega_{bl,t}$
    Shrinkage covariance matrix (active bets), $\hat{\Sigma}_{\alpha,t+1} = \kappa_a\Sigma^* + (1-\kappa_a)\hat{\Sigma}_{u,\alpha,t+1}$
  end
  Construct Portfolios at $t+1$
    Decide investible set
    Global minimum variance portfolio, $\boldsymbol{w}_{g,t+1} = \texttt{GMV}(\hat{\Sigma}_{g,t+1}, C_g(\boldsymbol{w}_g))$
    Systematic Bets portfolio, $\boldsymbol{w}_{s,t+1} = \texttt{SBM}(\hat{\Sigma}_{\pi,t+1}, \hat{\boldsymbol{\pi}}_{t+1}, C_s(\boldsymbol{w}_s), \gamma_s)$
    Active return tactical portfolio $\boldsymbol{w}_{a,t+1} = \texttt{ATP}(\hat{\Sigma}_{\alpha,t+1}, \hat{\boldsymbol{\alpha}}_{t+1}, C_a(\boldsymbol{w}_a), \gamma_a)$
  end
end

The FBPC algorithm corresponds to control flow charts in Figures 3.1 and 3.2. The Nomenclature is provided in Table 1. The FBPC algoirthm uses low-frequency financial data and consists of four major tasks: *Initialisation*, *Update Parameter Estimates*, *Prediction*, and *Portfolio Construction*. The *Initialisation* and *Update Parameter Estimate* steps of the Algorithm correspond to the *System Identification* component of the control flow. With the exception of Prediction, all subtasks for each of the major tasks can occur in paralell. Within the Prediction task, raw predictions are formed followed by regularisation.

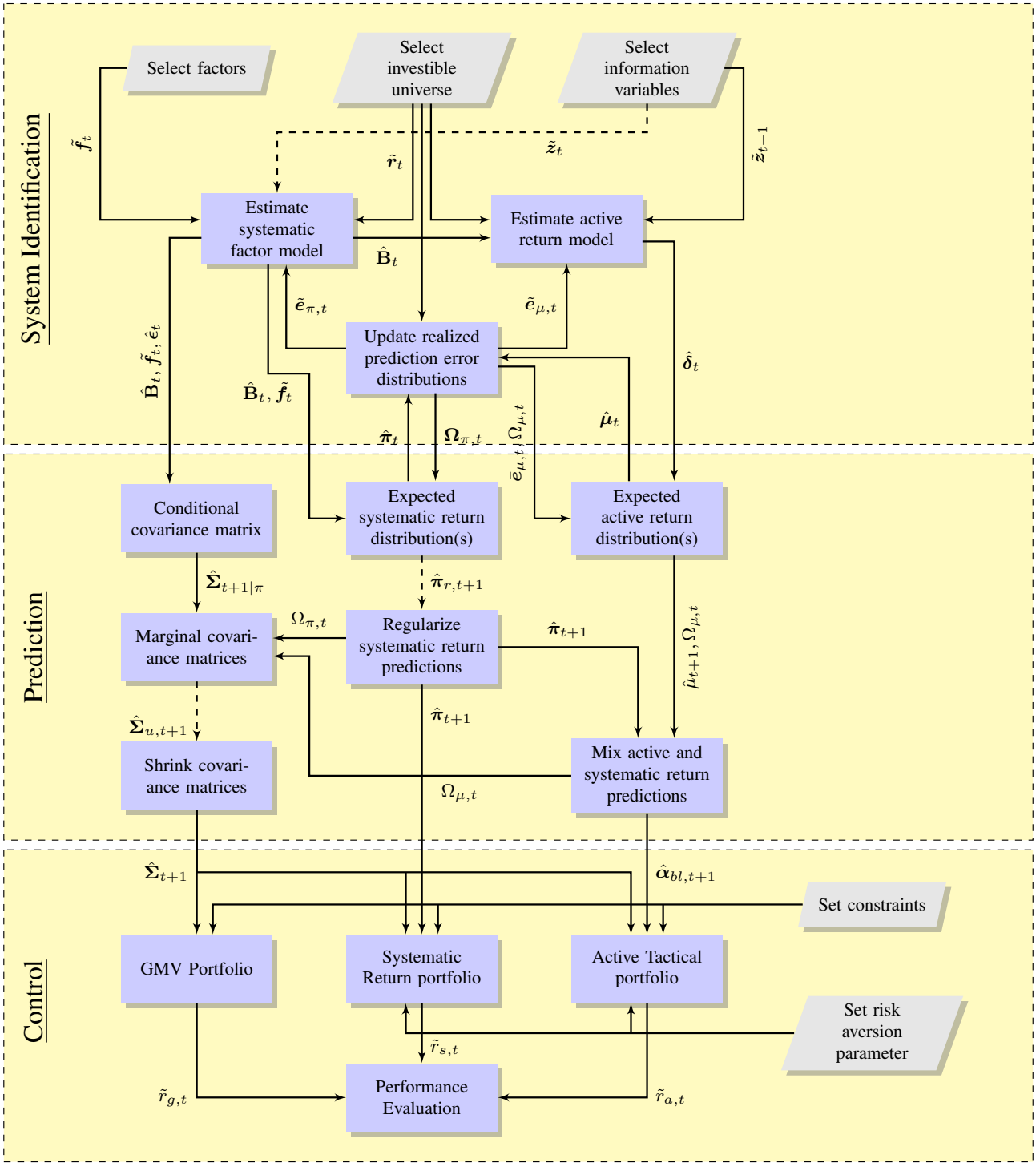


Figure 3.1: Annotated Control Flow Chart for Factor-Based Portfolio Control Algorithm: The control flow chart represents one iteration of the automated portfolio construction process. The chart reads from top to bottom. Each block in the algorithm represents a basic component which is labelled by the function it performs. Arrows represents the flow of data and variables beetween components. Dotted arrows represent connections that were not implemented in this research. Sub-algorithms corresponding to basic components, or collections of basic components, are dependent on user choices (end of Section 3.2.5). Grey blocks represent offline tasks or decisions, while blue blocks represent online tasks. The same chart is shown in Figure 3.2 except with formulae in place of descriptive labels. The Psuedocode is provided in Algorithm 2. Nomenclature is provided in Table 1.

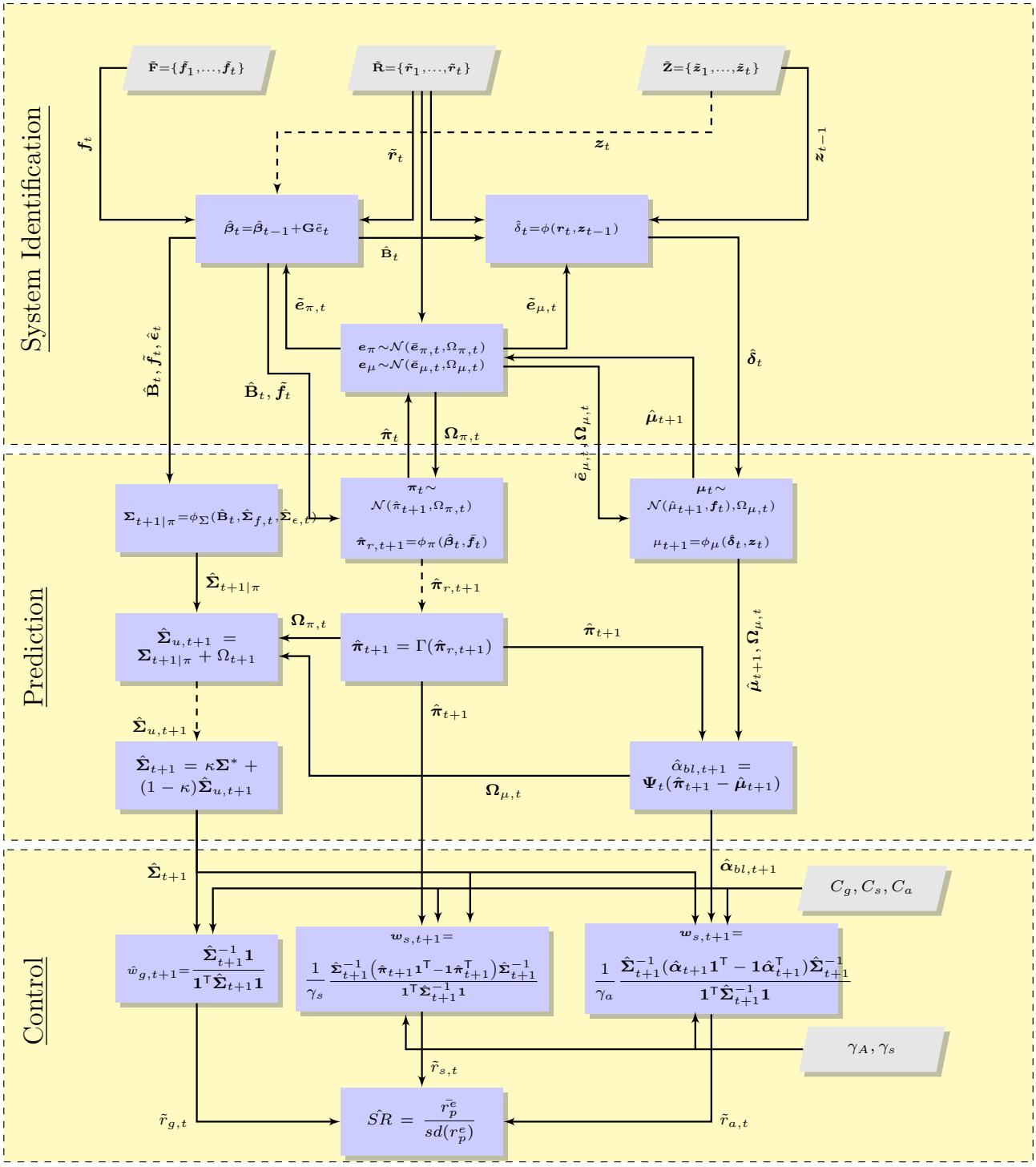


Figure 3.2: Control Flow Chart for Factor-Based Portfolio Control Algorithm: The control flow chart represents one iteration of the automated portfolio construction process. The chart reads from top to bottom. Each block in the algorithm represents a basic component which is labelled by the formulae of the task it performs. Arrows represents the flow of data and variables beetween components. Dotted arrows represent connections that were not implemented in this research. The outputs from the top grey blocks represent incoming data. As more data arrives at each timestep, the process is repeated. Grey blocks represent offline tasks or decisions, while blue blocks represent online tasks. The same chart is shown in Figure 3.1 except with descriptive labels in place of formulae. The Psuedocode is provided in Algorithm 2. Nomenclature is provided in Table 1.

offline process following the operation of the system on the available dataset.

The algorithm is divided into three phases which carry out the general compound tasks of system identification, prediction, and control. In the system identification phase, the current parameters (at time t) of the system are estimated online using the most recent market data ($\boldsymbol{f}_t, \boldsymbol{R}_t, \boldsymbol{z}_t$), and the model forecast errors ($\boldsymbol{e}_{\pi,t}, \boldsymbol{e}_{\mu,t}$). Parameters include the current factor exposures ($\hat{\boldsymbol{\beta}}_t$) and the firm-characteristic payoffs ($\hat{\boldsymbol{\delta}}_t$). A general algorithm to update models is provided in section 43. The current forecast errors are also used to update the forecast error distribution.

In the prediction phase, the mean and variance predictions used to construct each component portfolio are determined. The current parameter estimates are used to forecast future parameters, which are then used to construct the mean and covariance inputs. The conditional covariance matrix ($\hat{\Sigma}$) is constructed using the modelled risk factors and an estimate of the idiosyncratic risk ($\hat{\epsilon}$). The systematic risks will typically have the form: $\hat{\mathbf{B}}^{\top}\hat{\Sigma}\hat{\mathbf{B}}$. The marginal covariance matrices for each portfolio are constructed by adding the relevant uncertainty matrices to the conditional covariance matrix. The marginal risk matrix is shrunk towards a target, where the shrinkage intensity is determined offline.

During the prediction phase, expected systematic returns are constructed using an estimate of systematic risk premia ($\bar{\boldsymbol{f}}$) and the current factor exposures. Expected systematic returns can be regularised before portfolio construction (this is not implemented). Active returns are determined using the active return model. Active expected returns and systematic expected returns are blended using Theil's mixed estimator prior to constructing the active portfolio. The predictions are blended in proportion to their respective uncertainties.

Regularization primarily occurs at the prediction phase, but can be implemented at other stages in the process. For example, our implementation uses robust least-m adaptive filters [99] to ensure robustness against outliers. Regularization can also be implemented directly on the weights, as [48, 49] does in the offline situation, but this is not included in the current framework and is a consideration for future work.

The control phase is the final phase, and consists of constructing optimal component portfolios from the relevant modified mean-variance estimates. In implementation, the portfolio controls were determined analytically without the use of an optimiser. This is unlikely to work for larger scale implementations, with a large number constraints. The portfolios require risk-aversion parameters to be specified to determine the systematic and active portfolio bet size. Performance evaluation can occur online or offline.

Components may require user-defined inputs which are provided offline. User inputs can be classified into investor-specific and algorithmic configurations. Investor-specific configurations include subjective investor choices such as risk tolerance and portfolio constraints. Algorithmic configurations refer to tuning parameters of the component algorithms. In the current implementation, such configurations include the memory factors corresponding to each online estimation scheme, the robust filter tuning constant, the shrinkage targets and intensities, the frequency of parameter updates, and the frequency of portfolio rebalancing. These quantities can be optimized adaptively in a more sophisticated treatment.

Online Estimation

To make the system online, the estimation and prediction models are recursively updated. Controls are recalculated at each time step [3]. Our approach to online adaption is simple: we use exponential weights on observations to recursively update parameters. The exponential weighting places greater weight on more recent data, allowing the estimator to track time-varying parameters.

Section 2.5.2 provides a general form for a recursive estimator

$$\hat{\boldsymbol{\theta}}_t = \hat{\boldsymbol{\theta}}_{t-1} + \mathbf{G}_t \tilde{e}_t \tag{3.18}$$

where

$$\tilde{e}_t = \tilde{y}_t - \hat{y}_{t|t-1} \tag{3.19}$$

where $\hat{\boldsymbol{\theta}}_t$ represents estimate at time t for any parameter, $\mathbf{G}_t$ is known as the gain, $\tilde{e}_t$ is the a-priori estimation error, and $\hat{y}_{t|t-1}$ respresents a the prediction of y made at time $t-1$. The memory factor is included in the gain term for an exponentially weighted recursive estimator. The Online Parameter Estimation algorithm [90] (Alg. 3) can be used to recursively estimate a time series model.

Algorithm 3: Online Parameter Estimation

Function OPE($\boldsymbol{\theta}_0, \mathbf{H}_0^{-1}, \lambda, \mathbf{X}_t, \mathbf{y}_t$)
- **Input :** initial parameter estimate, $\boldsymbol{\theta}_0$
  inverse Hessian matrix prior, $\mathbf{H}_0$
  forgetfullness factor, λ
  input signal matrix until t, $\mathbf{X}_t$
  output signal matrix until t, $\boldsymbol{y}_t$
- **Output:** beta estimate at t, $\boldsymbol{\theta}_t$
  predicted output at t+1, y_{t+1}
- **begin**
  - Initialise $\mathbf{H}_0^{-1}$ and $\boldsymbol{\theta}_0$
  - calculate predicted value, $\hat{y}_t = g_t(\boldsymbol{\theta}, x_t, y_t)$
  - calculate a priori prediction error, $e_t = y_t - \hat{y}_t$
  - update inverse Hessian matrix, $\hat{\mathbf{H}}_{t|t-1}^{-1} = \frac{1}{\lambda}\{\hat{\mathbf{H}}_{t-1|t-2}^{-1} + \hat{\mathbf{H}}_{t-1|t-2}^{-1}\boldsymbol{\eta}_t \left[\lambda \mathbf{Q}^{-1} + \boldsymbol{\eta}_t' \hat{\mathbf{H}}_{t-1|t-2}^{-1}\boldsymbol{\eta}_t\right]\}$
  - calculate gradient vector, $\boldsymbol{\eta}_t$
  - update Gain matrix, $\mathbf{G}_t = \hat{\mathbf{H}}_{t|t-1}^{-1}\boldsymbol{\eta}_t \mathbf{Q}$
  - update parameter estimate, $\hat{\boldsymbol{\theta}}_t = \hat{\boldsymbol{\theta}}_{t-1} e_t$
- **end**

end

The algorithm is a second-order method that sequentially optimises a generalised least-squares objective function, constructed from time series data. It is common for parameters to be estimated by minimising prediction errors. If the prediction model is nonlinear, the objective function will not be quadratic in the parameters, and the algorithm will be approximately optimal. For an adaption algorithm that can handle a more general objective function, see Ljung [89].

[3] As a future consideration, this framework can be expanded to include recursive estimation of the controls themselves. Estimating and regularizing controls directly may be useful when considering trading costs explicitly. For example, see [100] for a Q-learning implementation for online trading

In the implementation, all parameters apart from the systematic factor exposures are updated using exponentially weighted sample estimators. Exponentially weighted moving averages are for all mean estimates, whilst exponentially weighted covariance estimates are used for all covariance matrices.

The various matrices need to be inverted at each time step. We can estimate the inverse matrix recursively. An exponentially weighted covariance matrix has the update scheme [**?**]:

$$\hat{\Sigma}_t = \lambda \hat{\Sigma}_{t-1} + (1-\lambda)\boldsymbol{x}_t \boldsymbol{x}_t \tag{3.20}$$

where $\boldsymbol{x}_t$ represents the centered observation at time t. By applying Woodbury's matrix inversion identity (see Appendix A.5), the inverse of the matrix can be recursively estimated as:

$$\begin{aligned} \hat{\Sigma}_t^{-1} &= \left[\lambda \hat{\Sigma}_{t-1} + (1-\lambda) x_t x_t^\top\right]^{-1} && (3.21) \\ &= \frac{1}{\lambda}\hat{\Sigma}_{t-1}^{-1} - \frac{1}{\lambda}\hat{\Sigma}_{t-1}^{-1} x_t \left[x_t^\top \hat{\Sigma}_{t-1}^{-1} x_t + \frac{\lambda}{1-\lambda}\right]^{-1} x_t^\top \hat{\Sigma}_{t-1}^{-1} && (3.22) \end{aligned}$$

The inverse matrix update equation reduces the computational burden from inverting a matrix to inverting a scalar. This scheme is used to update the uncertainty matrices used to blend expected returns, as well as the factor covariance matrix.

The covariance matrix of marginal returns that is used to construct portfolios is a more complicated estimator, composed of multiple matrices:

$$\begin{aligned} \hat{\Sigma}_{R,t} &= \kappa \Sigma^* + (1-\kappa)\hat{\Sigma}_{u,t} && (3.23) \\ \hat{\Sigma}_{u,t} &= \hat{\mathbf{B}}_t^\top \hat{\Sigma}_{f,t} \hat{\mathbf{B}}_t + \hat{\Sigma}_{\epsilon,t} && (3.24) \end{aligned}$$

Eqn 3.23 is a shrinkage covariance estimator, where Σ^* is the shrinkage target, and $\hat{\Sigma}_{u,t}$ is the unconditional covariance matrix estimated using a factor model. The factor model covariance matrix is constructed as the sum of the covariance due to systemic factor and factors due to idiosyncratic exposure, where the idiosyncratic exposure can include expected return uncertainty[4].

A recursive algorithm for the shrinkage estimator is not developed. However, if we shrink the inverse covariance matrix directly, we can focus on deriving a recursive estimator for the factor covariance matrix. Using of Woodbury's matrix inversion identity:

$$\hat{\Sigma}_{u,t}^{-1} = \hat{\Sigma}_{\epsilon,t}^{-1} - \hat{\Sigma}_{\epsilon,t}^{-1}\hat{\mathbf{B}}_t^\top \left[\hat{\mathbf{B}}_t \hat{\Sigma}_{\epsilon,t}^{-1}\hat{\mathbf{B}}^\top + \hat{\Sigma}_{f,t}^{-1}\right]^{-1} \hat{\Sigma}_{\epsilon,t}^{-1} \tag{3.25}$$

The computational burden of inverting the large dimensional covariance matrix has been reduced to inverting a matrix with a dimension equal to the number of factors.

User Choices

The implementation of the workflow requires that each component is given a specific description, which the user must choose depending on his requirements. A generic set of choices is listed below, together with a brief description rgarding general considerations.

1. **Investible Set**: The user must decide which stocks will be used to estimate models and which stocks are investible. The choice is often made with regard to a liquidity threshold. When

[4] The idiosyncratic risk and the uncertainty are typically modelled as diagonal covariance matrices.

stocks are not traded, no corresponding return signal is generated. When stocks are traded, there may be sizeable deviations in price given the absence of competition. These properties result in large errors in estimates. In addition, low liquidity stocks have much larger costs of trading due to price impact. Inventory in these stocks is limited, making it difficult to enter and exit positions.

2. **Factors and Information Variable Selection**: The chosen risk factors define a model for the covariance structure between assets and determine the risk premium bets earned by the GMV and systematic bets portfolio. A secondary consideration is how factor mimicking portfolios should be constructed, or if factors variables should be used. The information variables determine the tactical bets and should be chosen so as to provide an informational advantage over the risk factors. The problems associated with multicollinearity become increasingly prevalent as the number of variables in the model increases.

3. **Sampling, Estimation, Forecast, and Rebalancing Frequency**: Analysis of the intertemporal problem (Section 2.2.4) suggests that (without considering trading costs) it is theoretically optimal for sampling, estimation, and portfolio rebalancing to occur continuously. In practice, the optimal sampling, estimation, and rebalancing frequency is empirically motivated by considering the costs and benefits associated with different frequencies. Determining estimation and forecast frequency involves considering the prediction accuracy, whilst determining rebalancing frequency depends on trading costs.

4. **Expected Return Estimation, Prediction, and Uncertainty**: Our algorithm determines the risk and return models as a two-step procedure: current state estimation, followed by prediction. The filters used to estimate the model parameters should allow for various features of the data which could cause large errors in estimates. This includes model endogeneity, high multicolinearity of the varibles, and outliers. In addition, the estimation method may need to accomodate trend, seasonal, and cyclic components that may be present.
   For short forecast horizons, setting the forecast equal to the current estimate may be sufficient. The prediction method becomes more important as the forecasting horizon increases. The expected return uncertainty is used to control the size of the active bets. Confidence in the estimate should reflect the forecast accuracy.

5. **Covariance Model Estimation and Prediction**: The covariance model represents the total uncertainty that must be accounted for in the portfolio construction process. The components of the total uncertainty include systematic risk, idiosyncratic risk, and expected return uncertainty. A variety of multivariate volatility forecasting techniques exist for estimating and forecasting the various components of the total risk. A review of standard techniques can be found in [101].

6. **Portfolio Optimisation**: The equations in Figure 3.2 used to construct portfolios are derived from the standard mean-variance objective function (Equation 3.2). The mean-variance objective can be modified to incorporate trading costs. Alternatively, trading cost considerations can be incorporated as a turnover constraint. Constraints can also be used to restrict short positions. The risk tolerance parameters are used to determine the bet size. The risk tolerance parameter can be selected to maximise the ex-ante single period Sharpe Ratio.

Chapter 4

Implementation

4.1 Data

This research makes use of two datasets. The first dataset is a *development* dataset consisting of monthly data of traded equities on the Johannesburg Stock Exchange (JSE) over the period January 1994 to June 2007. The development dataset was provided by Thompson-Datastream. The development dataset was used in the motivating studies [19, 20], and is used in this research for initial prototyping, and to verify the data of the second dataset. No results presented in this document pertain to the development dataset. Further details relating to this dataset can be found in the above-mentioned studies.

The second dataset used in this research is a *production* dataset. It is used for further development of the algorithm, to illustrate operation of the algorithm, and to measure its performance. The production dataset consists of daily-sampled data for equities traded on the JSE over the period 14 January 1994 to 21 April 2017. The development dataset was provided by iNet-BFA, and was downloaded on 23 April 2017 prior to iNet-BFA's merger with IRESS. The production dataset suffers from two key limitations. The first limitation was that key variables had to be handcrafted. This is discussed further below. The secondary limitation was that sufficiently detailed industry classifications were not available at the time of downloading. As a result, REITs are included in the dataset.

Table 4.1: Summary of Datasets

Dataset	Provider	Frequency	Start Date	End Date	Observations	No. of Tickers
Development	Thompson-Datastream	Monthly	30-04-1994	31-05-2007	158	471
Production	iNet-BFA	Weekly	14-01-1994	21-04-2017	982	666

This research makes use of two datasets. The *development* dataset was used in the studies [19, 20] and was used in this research for initial prototyping. The *production* dataset is used for further development of the algorithmic strategy and for performance measurement. Results (Section 4.3) were prepared using the production dataset. The datasets are available at `https://github.com/apaskara/Algo_Invest`.

The datasets contain data which can be categorised into, what we will refer to as, *market data* and

company data. Market data are a set of basic standard metrics reflecting the trading activities of a market. For low-frequency data, the most important of these are the stock prices and trading volumes. Company data are business metrics generated from the (annual) published financial accounts fo each company. Common financial ratios, such as book value to price (BVTP), price-to-earnings (PE), dividend yield (DY) combine market and company data. We will also refer to these variables as market data. Macroeconomic data are not used in this implementation.

The published financial accounts of a listed stock are prepared in accordance to JSE listing requirements and the prevailing accounting standards, and are audited[1]. This means that key financial variables are calculated according to standardised definitions. Accounts are published twice a year: interim results are released at mid-year and full-year results are released following the completion of the financial year. Interim results are generally condensed, whilst full-year results are always comprehensive. The JSE listing rules [102] require that the audited full-year results must be published within three months of the financial year end. It is common for the financial year-end date not to coincide with the end of a calendar year, and for financial year end dates to vary between companies.

Failure to meet reporting deadlines may result in suspended trading in the company's financial instruments. Missing a reporting deadline is a rare occurence and frequently precedes severly detrimental financial results. In the original (unprocessed) datasets, data belonging to the published accounts is stored at the financial year-end date, rather than the release date. The dates that the accounts were made public are not available from the data-providers, and it is common for researchers to arbitrarily, but prudently, lag these data to avoid look-ahead bias. In this implementation, the data relating to published accounts were lagged by 88 working days in the production dataset. This roughly equates to four months, where an additional month over the three month deadline is added for prudence.

The production dataset contained various descrepencies scattered accross each variable. These were identified through comparing the data against external sources, including the development dataset, the actual published accounts and the data from non-proprietary data vendors, for various test cases. Of particular concern were the inaccuracies in the determination of the BVTP Ratio. Investigation into the test cases revealed the reasons for the discrepencies: an inconsistent definition of book-value accross companies, a lack of clarity regarding determination of share price, and inconsistencies between the currency of the book-value and the share price. The issue of inconsistent currencies in the database is specific to dual-listed companies. As a result, the BVTP used for modelling was re-calculated using data that was consistent with external sources. The methodology for calculation is mentioned below. Although the new methodology is not free from errors, it is preferable to use verifiable data, so that the accuracy of results can be compared against a set of data that is understood.

Amongst the difficulties encountered with financial datasets is missing data, resulting in variation in the lengths of historical data for each ticker. Differences exist with regard to the quantity of company and market data available at each point in time. The missing data is a property of the system rather than the data collection process. Without trading activity in a stock, prices are not generated. Reasons for low trading volumes include availability and desirability. Asynchoronous trading complicates the estimation of the covariance matrix, which may no longer be postive definite. In this research, it is assumed that if price data is missing, the stock is not tradeable

[1] JSE listing requirements can be found at:
https://www.jse.co.za/content/JSERulesPoliciesandRegulationItems/JSE%20Listings%20Requirements.pdf.

over that period.

4.1.1 Market and Company Variables

A list of variables used in the production dataset together with the construction methodology is presented below. The definitions are reproduced from [103]. The variable choice follows the studies [19, 20]. Prior to processing, all market data is sampled daily whilst company data is sampled annualy. Market data is indicated by a **M** and company data is indicated by **C**.

TR (M): (log) Total Returns consist of capital gains and all distributions to shareholders. Distributions are most commonly in the form of regular or special dividends, but can also include non-cash ("in-specie") distributions. Non-cash distributions can be in the form of scrip dividends or share-repurchases. The total return index must also account for possible variations in the number of outstanding shares, which can occur from a share split or share buy-back. Failure to account for this may erronously inflate or deflate the return of the share, without an increase in the market value of the company.

Unfortunately, a Total Return Index (TRI) was not available from iNet at the time of downloading the data. The price data from iNet-BFA has been adjusted for share splits and buybacks. To calculate the total return, only cash dividends are considered. The methodology to calculate total returns is as follows: cash dividends are added to the share price at the ex-dividend date, and are divided by the previous stock price $r_t = ln(\frac{P_{t+1}+D_{t+1}}{P_t})$. Dividends are paid three days after the ex-dividend date to the shareholder who holds the stock at the last-day-to-trade (LDT) date. The following day, the stock becomes ex-dividend. Consequently, on the ex-dividend date, stock prices reduce by the declared dividend amount to account for the loss of future cashflow, in accordance with the dividend discount model [2].

MV (M): free-float Market Value is the daily-quoted market capitalisation, and is equal to MC_t = No. shares in Issue × Share Price$_t$. The logarithm of the Market Value is used in implementation.

BVTP (C): We were unable to reconcile the BVTP figures reported in the database with external information, and so reverted to calculating the ratio from first principles. Here, we have that $BVTP = \frac{\text{avg. BV}_t}{\text{Share Price}_t}$. We have defined the book-value to be equal to the total shareholder equity, which is simply calculated as the Total Assets – Total Liabilities from the balance sheet. The recorded amounts for these variables did reconcile with the publicly-available published accounts.

Dual-listed companies may publish accounts in a foreign currency, where market data is always in the local currency. To avoid the problem of an inconsistent currency between market and company data, the BVTP ratio was calculated using the product of the return on equity (RoE) and earnings yield (EY) ratios,

$$BVTP_{k+t} = \frac{1}{RoE} \times EY = \frac{\text{avg BV}_k}{\text{Net Earnings}_k} \times \frac{\text{Earnings per Share}_{k+t}}{\text{Price per Share}_{k+t}}$$

where k is the release date of the most recent published data, and t is any time period after that. As mentioned, in the absence of knowing the true release data, k is set to 88 days

[2] The dividend discount model is given by: $P_0 = \sum_{t=1}^{\infty} \frac{D^t}{(1+r)^t}$

following the the financial year end. Fortunately, the reported earnings yield from the iNet database was calculated using consistent currencies. Thus, by standardising each variable with the corresponding currency, we were able to remove the effect of the currency discrepancy. However, additional inaccuracies were introduced through potentially inconsistent definitions between the Net Earnings and Earnings per Share.

Long-Term Momentum (M): The price momentum variables are calculated from the adjusted prices. It is equal to difference between year-to-year prices: $MOML = P_t - P_{t-12}$, where $t-12$ refers to the price at the end of the previous year.

Short-Term Momentum (M): The short term momentum is the quarterly difference between prices: $MOMS = P_t - P_{t-3}$, where $t-3$ refers to the price at the end of the previous quarter.

Dividend-Yield (M): The dividend yield was amongst the first company variables used to predict future returns. It is calculated as $DY_t = \frac{DPS_k}{P_t}$, where DPS_k is the dividend per share and the dividend relates to the previous ex-dividend date, indicated by k.

Risk-Free Rate (M): 91-day local treasury bills are used as the risk-free instrument. The quoted discounts are released by the Reserve Bank and are available the iNet database. These instruments are quoted at a discount, and the price of the instrument is calculated as $P = (100 - d(\frac{91}{365}))$, where d is the annual discount rate. The effective log risk-free return, at the relevant frequency, is calculated by the formula

$$P(1+r)^{\frac{91}{f}} = 100 \implies \ln(1+r) = -\left(\frac{f}{91}\right)\ln\left(1-d\left(\frac{91}{365}\right)\right)$$

where f is the number of days in a period at the desired frequency.

4.2 Methodology

In section 4.3, the out-of-sample results for a particular calibration of the FBPC algorithm are presented. This section details the steps required to replicate the results. The steps for the implementation are presented in the format of algorithmic psuedocode. Offline user choices, and the process for hyper-parameter tuning are discussed separately. The corresponding MATLAB code used to prepare the results can be found in Appendix B, and is available for download at the github site `https://github.com/apaskara/Algo_Invest`.

Initial development of the FBPC algorithm made use of the code patterns used in the studies [19, 20]. The code patterns re-used from these studies relate to data preparation, and characteristics models estimation. The code patterns also served as a guide to set out the MATLAB code to prepare results of the FBPC algorithm. Code patterns for the remaining components of the FBPC algorithm (see Figure 3.1) and for performance evaluation were developed from scratch.

In addition, this research benefitted from a MATLAB toolbox written predominantly by T. Gebbie and D. Wilcox. The toolbox is made available at the above-mentioned github repository with permission of the authors. Of particular importance is a series of functions used for constructing exponentially-weighted moving average (EWMA) estimates. As part of this research, the EWMA functions were extended to allow for missing data.

4.2.1 Data Preparation

The development dataset is used to motivate corrections and refinements to the algorithm, whereas the production dataset is used to replicate real-world implementation and to assess performance. The results presented in Section 4.3 are prepared using the production dataset. The processing of the production dataset and descriptions of the variables are detailed in Section 4.1. As discussed, key limitations of the production dataset are that the total return index had to be created from scratch, and that industry classifications for tickers were not available at the time of downloading. As a result, the list of tickers includes REITs.

The production dataset is split into an in-sample training period and an out-of-sample testing period. The in-sample period is further divided into an initialisation period, and online update period. The initialisation period is set from 1994 to 2000, the training period is set to be from 2000 to 2007, while the validation period is set to be from 2008 to 2017. To avoid data snooping, the validation period excludes the data in the Wilcox and Gebbie [19] study, which gave rise to the hypothesis about the time-variation in alpha. The characteristics data relating to the published financials are poorly populated in the years 1994 to 1998. This is partly due to changes in reporting requirements, major changes in the South African market following the first democratic elections, as well as deficiencies in the proprietor's database. The fairly lengthy initialisation period is chosen to partially alleviate biases in the filters originating from the non-representative sample.

Following initialisation, parameters are sequentially updated and portfolios are constructed in the online update period. The choice of hyperparameters are determined by comparing corresponding portfolio performance, and alternative component-specific metrics, over this period. Interactions between hyperparameters can result in a set of optimal hyperparameters that are unconventional and are not interpretable. Thus, our hyper-parameter choice is not purely empirical, but guided by expert opinion.

Following initialisation and hyper-parameter tuning, the optimal model is selected amongst alternatives according to the Deflated Sharpe Ratio (DSR) peformance criterion. The Deflated Sharpe Ratio is defined and discussed in further detail in the results section. The optimal model is then tested in the out-of-sample dataset and assessed using the Probabilistic Sharpe Ratio (PSR), also defined in Results section.

4.2.2 Offline User Choices

Our online portfolio control algorithm dynamically constructs mean-variance efficient portfolios. The expected returns are seperated into two component models: one for systematic returns and one for active returns. The parameters for this algorithm are learned online through exponential smoothing. Hyper-parameters are fixed and determined offline. As discussed in section 11, the user has various design choices which must be specified prior to operation of the algorithm. These design choices made for this implementation are discussed below.

Stock Universe

We consider three investible universes, consisting of the 40 largest stocks (Top40), the 100 largest stocks (Top100), and the 160 largest stocks (Top160). Size is determined by the daily reported free-float market capitalisation. These universes correspond to the JSE Top40; the combined JSE Top40 and Mid Cap indexes; and the JSE All Share index, respectively. The JSE is highly skewed in terms of market cap with each universe accounting for approximately 80, 90, and 99% of the total JSE market cap respectively. Constituents of each stock universe change over time. Models for all stocks are estimated regardless of their inclusion in the investible set. However, only investible stocks are included in the factor mimicking portfolio construction. The risk-free asset is excluded from the investible set.

Systematic Risk Factors and Information Variable Selection

The choice of systematic risk factors and information variables is suggested by Wilcox and Gebbie's motivating study [19]. Systematic returns are modelled by the Fama-French 3 Factor model, while the following information variables are used to determine active return models: BVTP, MV, CAPM beta, MOML, MOMS, and DY, FF3F alpha.

Factor Mimicking Portfolios Construction

We follow the procedure in the original Fama and French study [32] to construct the factor mimicking portfolios, making appropriate adjustments to account for the unique features of the JSE. At each time period t, stocks within the investible universe are ranked according to market capitalisation (size) and BVTP (value). Stocks are split into bins based on these rankings. First, stocks are split into two bins based on size, where the split is determined by the median. The bins are named “big" (B) and “small" (S) respectively. Then stocks are split into three bins according to value, where the splits are the 33^{rd} and 66^{th} percentile respectively. The value bins are labelled “low" (L), “medium" (M), and “high" (H) respectively [3]. As Fama and French [32] note, the splits within the data are arbitrary. Stocks without data at the time of portfolio sorting are excluded.

Following the double-sort procedure, the following intersections bins are formed

1. Low and Small (hereafter, LS)

[3] In this regard, “high" value corresponds to high book-value-to-price which is usually associated with distressed firms. “High" value corresponds to “cheap" firms.

2. Medium and small (MS)
3. High and small (HS)
4. Low and big (LB)
5. Medium and big (MB)
6. High and big (HB)

Constituents in each intersection bin are equally weighted to form intersection portfolios at the start of time $t+1$. Factor mimicking portfolios (FMPs) are formed using these intersections portfolios, and are created as follows:

- size: $\frac{1}{3}(LS+MS+HS)-\frac{1}{3}(LB+MB+HB)$
- value: $\frac{1}{2}(HS+HB)-\frac{1}{2}(LS+LB)$

Each FMP is a zero-cost portfolio that captures the return differential between big and small stocks, and the return differential between high and low value stocks. FMPs should ideally be constructed from orthogonal characteristics variables in order to isolate each variable's effect on realised returns. The double-sort FMP construction procedure is a crude method to correct for correlations between characteristics. The FMPs earn a return over the period $t+1$, and are reconstructed at the end of the period. FMP returns at time $t+1$ are used to estimate contemporaneous factor loadings.

Barring the first year (1994) of observations in the dataset, the time-series of factor realisations is well populated. The first year contains no BVTP data since BVTPs were calculated using a differenced series of Return on Equity. The ability of the FMP to accurately reflect the underlying risk factor is affected by number of assets in each intersection portfolio. The returns of intersection portfolios with a low number of assets contain idiosyncratic error. Measurement errors in the FMP return series will result in errors in estimated factor loadings.

The market portfolio is constructed at each time step by combining all stocks in the investible universe in proportion to each stocks market capitalisation. The market factor premium is calculated as the the return of the market portfolio minus the risk-free rate.

Sampling, Estimation, Prediction, and Rebalancing Frequency

The market data has a weekly sampling frequency. The data is reported at the end of each Friday or, in the case of a holiday, the last business day of the week. Company variables are sampled annually using published full-year financials. These variables are lagged by 18 weeks (approximately 4 months) from the financial year-end to approximate the actual publishing date. The variables from the company financials are constant between annual publishing dates. For derived variables, such as BVTP, weekly market data is used with company data.

Estimates are updated weekly upon the arrival of new data. The forecast horizon is fixed at one week. Weekly and monthly rebalancing frequencies were compared on the strategic benchmark portfolio to examine the impact of rebalancing frequency on net performance.

Systematic Return Estimation and Prediction

Our algorithm uses a two-step procedure to form portfolio inputs. First, the current parameters are estimated. Then, predictions are made based on the current parameters. Our implementation sets the prediction equal to the current estimate. All mean and covariance estimates are determined through some use of exponential smoothing. Separate memory factors are used for factor prediction, factor loading estimation, characteristic model prediction, and uncertainty matrix estimation. Setting the values of the memory factors is discussed in the hyper-parameter tuning subsection.

The systematic expected returns are used to reflect the average difference between the factor risk premia. The forecast equation for a general factor model is given is given by [54]:

$$\begin{aligned} \pi_{i,t+1} = \mathbb{E}_t[r_{i,t+1}] &= \mathbb{E}_t[\boldsymbol{f}_{t+1}^{\mathsf{T}} \boldsymbol{\beta}_{i,t+1}] \\ \mathbb{E}_t[r_{i,t+1}] &= \mathbb{E}_t[\boldsymbol{f}_{t+1}]^{\mathsf{T}} \mathbb{E}_t[\boldsymbol{\beta}_{i,t+1}] + Cov(\boldsymbol{f}_t, \boldsymbol{\beta}_{i,t}) \end{aligned} \tag{4.1}$$

where the vector of factor loadings ($\boldsymbol{\beta}$) does not include an intercept term. Our forecast is equal to the current estimate of the systematic expected return:

$$\hat{\pi}_{t+1|t} = \hat{\pi}_{t|t} = \bar{\boldsymbol{f}}_t^{\mathsf{T}} \hat{\boldsymbol{\beta}}_t \tag{4.2}$$

where $\hat{\pi}_{t+1|t}$ represents the prediction for time $t+1$ conditional on the information at time t. Similarly, $\hat{\pi}_{t|t}$ represents the current estimate of the systematic expected return, conditional on the current information set. The predictions in our implementation do not consider the covariance component (that is, we set $\text{Cov}(\boldsymbol{\beta}_{t+1}, \boldsymbol{f}_{t+1}) = 0$).

A short-coming of our implementation is that we do not estimate the FF3F model as a conditional model. We use exponential weightings on the unconditional formulation of the FF3F to capture any time-variation in the factor loadings. Estimation error in the factor loadings arising from model misspecification is not accounted for. The simplicity of the forecast method motivates future investigation into more complicated recursive instrumental variable forecasting methods that include trend, and seasonal components.

Active Return Estimation and Prediction

We follow the same method as Haugen and Baker [82] to predict expected returns using characteristics. We consider nine different characteristic models. The variables used in each model are specified in table 4.2. The predicted time-varying alphas used in the tactical portfolio are calculated by mixing predictions from the characteristics model and the systematic return model. The characteristics model is a cross-sectional regression model of excess returns (above the risk-free rate) on lagged characteristics. Our implementation estimates the model using a robust least squares regression at each time step. Characteristics are lagged by 4 weeks.

The current estimate for the expected return is given by:

$$\hat{\mu}_t = \hat{\boldsymbol{\delta}}_t^{\mathsf{T}} \boldsymbol{z}_{t-4} \tag{4.3}$$

where $\hat{\mu}_t$ denotes the expected return of at time t. Characteristic model return predictions are calculated as the product of smoothed characteristic payoffs, $\bar{\boldsymbol{\delta}}_t$, and the characteristic information at time $t-3$:

$$\hat{\mu}_{t+1} = \bar{\boldsymbol{\delta}}_t^{\mathsf{T}} \boldsymbol{z}_{t-3} \tag{4.4}$$

Characteristic model predictions and systematic return predictions are blended using Theil's mixed estimator. The alpha prediction used in the tactical portfolio is given by the difference of the blended expected return prediction and the systematic return prediction. Our implementation calibrates the uncertainty matrix for each model using a diagonalised exponentially weighted sample covariance estimate of ex-post forecast errors. The diagonalisation is the simplest method to account for ill-conditioning due to asynchronous trading.

Table 4.2: Characteristics Models

Model	Variables
1	BVTP, MV
2	BVTP, MV, CAPM β
3	BVTP, MV, MOML
4	BVTP, MV, MOML, CAPM β
5	BVTP, MV, MOML, MOMS
6	BVTP, MB, CAPM β, MOML, MOMS
7	BVTP, MB, CAPM β, MOML, MOMS, FF3F α
8	BVTP, MB, CAPM β, MOML, MOMS, DY
9	BVTP, MB, CAPM β, MOML, MOMS, FF3F α, DY

Characteristics models are cross-sectional regression models. The variables are defined as follows: BVTP (book-value-to-price), MV (market value), CAPM β (estimated CAPM beta), MOML (long-term momentum), MOMS (short-term momentum), FF3F α (estimated intercept from FF3F regression), DY (dividend yield)

Covariance Model Estimation and Prediction

Each component portfolio uses a different covariance matrix reflecting the different relevant sources of uncertainty. The covariance matrix models the unanticipated return and, if appropriate, also captures the uncertainty associated with the expected returns. For each portfolio, the covariance matrix is shrunk towards the identity matrix.

In our implementation, the conditional risk model for each portfolio is given by the factor risk only:

$$\hat{\Sigma}_{t+1} = \hat{\Sigma}_{F,t+1} \tag{4.5}$$

where $\hat{\Sigma}_{F,t+1}$ is the prediction of the systematic factor risk matrix, constructed using the Fama-French risk factors. The factor risk matrix is a low-rank matrix that is equal to:

$$\hat{\Sigma}_{F,t+1} = \hat{\mathbf{B}}_{t+1|t}^{\mathsf{T}} \hat{\Sigma}_{f,t+1|t} \hat{\mathbf{B}}_{t+1|t} \tag{4.6}$$

where the predicted covariance matrix of the factor realizations is determined using a flat forecast of the current covariance estimate:

$$\hat{\Sigma}_{f,t+1|t} = \hat{\Sigma}_{f,t|t} = \lambda \hat{\Sigma}_{f,t-1|t-1} + (\tilde{\boldsymbol{f}}_t - \bar{\boldsymbol{f}}_t)(\tilde{\boldsymbol{f}}_t - \bar{\boldsymbol{f}}_t)^{\mathsf{T}} \tag{4.7}$$

and predicted factor loadings are equal to the current estimates:

$$\hat{\mathbf{B}}_{t+1|t} = \hat{\mathbf{B}}_{t|t} \tag{4.8}$$

Constructing the GMV

The risk matrix for the GMV portfolio is constructed by shrinking the conditional risk matrix to the identity matrix.

$$\hat{\Sigma}_{t+1} = \kappa \mathbf{I} + (1-\kappa)\,\hat{\Sigma}_{F,t+1} \tag{4.9}$$

where $\mathbf{I}$ is the identity matrix, and κ is the shrinkage intensity. Idiosyncratic risk is not explicitly modelled to avoid the issues associated with error-maximisation[4]. Without shrinkage, the conditional risk model is not invertible [44]. The shrinkage intensity is chosen based on the in-sample volatility of the GMV portfolio return. Setting the shrinkage intensity is discussed further in the hyper-tuning section.

Constructing the Systematic Bet Portfolio

Systematic bets are determined using the systematic model predictions for returns, derived from the Fama-French three factor model. The systematic portfolio risk matrix is constructed by shrinking the marginal predictive covariance matrix to the identity matrix. The marginal predictive covariance matrix is given by:

$$\hat{\Sigma}_{t+1} = \hat{\Sigma}_{F,t+1} + \hat{\Omega}_{\mu,t+1} \tag{4.10}$$

where $\hat{\Sigma}_{F,t+1}$ is the covariance due to factor risk, and $\hat{\Omega}_{\mu,t+1}$ is the (diagonal) uncertainty covariance matrix of the expected returns. Incorporating uncertainty of the parameters makes the system invertible, but not necessarily well-conditioned. Shrinkage can assist with ill-conditioning. The shrinkage covariance matrix is given by:

$$\hat{\Sigma}_{t+1} = \kappa \mathbf{I} + (1-\kappa)\left(\hat{\Sigma}_{F,t+1} + \hat{\Omega}_{\mu,t+1}\right) \tag{4.11}$$

Shrinkage intensity is chosen based on in-sample performance.

Risk tolerance is treated as a free parameter, which in our implementation is used to subsume the leverage constraint. The risk tolerance is chosen to fix the sum of the absolute values of the weights, $\sum_{i=1}^{N} |w_i| = 1$. This implies that the maximum weight in a single stock is 0.5. The leverage constraint limits turnover and protects the portfolio against large error-maximised positions. Alternative specifications of the risk tolerance parameter were considered during the hyper-tuning phase.

Constructing the Tactical Bet Portfolio

The implemented tactical portfolio represents the characteristic model choice that yielded the highest in-sample realised performance. Tactical bets are constructed using the blended alpha predictions. The tactical portfolio marginal predictive covariance matrix is given by the sum of the factor risk and the diagonalised uncertainty matrix of the blended estimate. The active risk tolerance parameter is treated similarly to the strategic portfolio construction case. Incorporating the leverage on the tactical portfolio implies that (if the GMV is long-only) there are no short positions in the combined portfolio.

Hyper-Parameter Choice

In this implementation, the hyper-parameters consist of different memory factors used to estimate the different models, the initial values for each filter, and the different shrinkage intensities for the

[4] Estimation of idiosyncratic noise is particularly susceptible to error-amplification. The relative error is largest for the subspaces corresponding to the small eigenvalues in the covariance matrix [104].

covariance matrix in each component portfolio. In particular, memory factors corresponding to the following tasks are specified: systematic return prediction λ_S, factor premium and covariance prediction λ_F, characteristic model return prediction λ_C, uncertainty matrix estimation λ_v. It should be noted that using the same memory factor for the factor premiums and covariance matrix implicitly assumes that these variables experience the same temporal variation. This is done for convenience, in order to avoid the difficulties associated with measuring the quality of the covariance prediction.

Initial values for the RLM filter are the factor loadings, β_0, and the search direction $(\mathbf{F}'\mathbf{F})_0$. Factor returns initial values are indicated by f_0 and $\Sigma_{f,0}$. An initial value is required for the characteristic model payoff, $\hat{\delta}_0$. Initial parameter uncertainty matrices are specified for the characteristic models, $\Omega_{\mu,0}$, and the systematic return models, $\Omega_{\pi,0}$, respectively.

4.2.3 Hyper-Parameter Tuning

In this subsection, we discuss the process of selecting hyper-parameters. The hyper-parameter tuning process involves running the algorithm to construct portfolios. The same algorithm is used in the same manner to prepare out-of-sample results. Thus, this subsection also serves to detail the steps to reproduce the results presented in the next section. The process described below is repeated for each investible universe.

The hyper-parameters for each component should ideally be determined by an optimization. Ultimately, hyper-parameters should be tuned to provide the best out-of-sample realised portfolio performance. The realised in-sample portfolio performance may not generalise out-of-sample owing to the large amount of noise in the system, the non-stationarity of the system signals, and the small quantity of data. The interpretability of the hyper-parameters is important to make qualitative judgements about out-of-sample performance. The tuning experiments revealed that it is possible that minor changes in some hyper-parameters can result in unexpected and nonsensical portfolio performance. As such, the objective function for the hyper-parameter optimization should also be appropriate for the intended use of the particular component in the workflow. For example, prediction accuracy should also be considered when determining optimal hyper-parameters for prediction. This helps to ensure that each component is working as expected.

Our implementation adheres to the guideline of component-specific metrics to determine hyper-parameters but avoids the use of an optimization. A numerical optimization procedure is required to determine the smoothing parameter and the initial value even in the the simple case of single variable exponential smoothing. For more complicated exponential smoothing predictions, the criterion function has multiple local optima, requiring more sophisticated and robust optimization algorithms, which are outside of the scope of this research. Our implementation determines hyper-parameters manually through a greedy approach. A pre-specified subset of hyper-parameters is considered in order to reduce the search space. The pre-specified set contains a small number of alternatives and is guided by expert opinion. [5]

The full dataset is split into the in-sample (training set) and out-of-sample (validation) datasets. The hyper-parameters are tuned using the in-sample data. The in-sample period consists of 729 time periods, whilst the out-of-sample period consists of 486 time periods. A third of the in-sample

[5] For example, memory parameters in the range of 0.97, 0.98 have been used by T. Gebbie in practice using monthly data

data (243 periods) is used for initialisation , and is denoted by T_1. The remaining 486 observations are used for the update period, and is denoted by T_2. The end of the validation set is denoted by T_3. The initialisation period is used to determine the initial values for the different component models. The update period is used to calibrate hyper-parameters that depend on the realised returns. The algorithm is run online and portfolios are constructed during the update period. The validation dataset is used for out-of-sample testing. The optimal in-sample and corresponding out-of-sample results are presented in the Results section.

The prediction models are implemented as online filters, and require initialisation during the initialisation period as well. In the initialisation period, initial values are determined in an offline manner using all data over the period. Memory factors are determined separately for the initialisation period and the update period, and are selected to minimise forecast errors. In addition to expert opinion, the subset of memory factors is informed by the decay of weights on historical observations. For simplicity, covariance models use the optimal memory parameter corresponding to the expected return model. This avoids the difficulties associated with quantifying optimal variance forecasts [101]. Each hyper-parameter is determined in isolation, corresponding to the implicit assumption that there are no interdependencies between hyper-parameters.

The initial values applicable to the different prediction models are determined before the other hyper-parameters. The initialisation procedure differs for the systematic model predictions and the characteristic model predictions. The procedure for the systematic return model is somewhat more involved and is discussed first.

Systematic Return Predictions

The steps to prepare systematic predictions are presented below as algorithmic psuedo-code. The derivation for the RLM adaptive filter is presented in appendix A.6.

Algorithm 4: Implementation - Systematic Return Prediction

Data: FMP portfolio returns, $\tilde{\mathbf{F}} = [\tilde{\boldsymbol{f}}_{mkt}, \tilde{\boldsymbol{f}}_{smb}, \tilde{\boldsymbol{f}}_{hml}]$
stock returns, $\tilde{\mathbf{R}}$
Result: Factor loadings, $\hat{\mathbf{B}}_{t+1}$
Systematic return predictions, $\hat{\boldsymbol{\pi}}_{t+1}$
Uncertainty matrix, $\Omega_{\pi,t+1}$

begin
  initialise factor model
    estimate mean FMP return, $\bar{\boldsymbol{f}}_{T_1} = \frac{1}{T_1}\sum_{t=1}^{T_1} \tilde{f}_t$
    estimate factor covariance matrix, $\left[\Sigma_{f,T_1}\right]_{i,j} = \frac{1}{T_1-1}\sum_{t=1}^{T_1}(\tilde{e}_{i,t} - \bar{e}_i)(\tilde{e}_{j,t} - \bar{e}_j)$
  end
  initialise parameter filter
    for *each company* **do**
      if *company has return data* **then**
        Offline equivalent initialasation, $\hat{\beta}_0 = 0$ and $(\mathbf{F}^\mathsf{T}\mathbf{F})_0^{-1} \to 0$
        estimate for factor loadings, $\boldsymbol{\beta}_{T_1} = \texttt{RLM}(\boldsymbol{\beta}_0, (\mathbf{F}^\mathsf{T}\mathbf{F})_{T_1}^{-1}, \lambda_s, \mathbf{F}_{T_1}, \mathbf{R}_{T_1}, w(r))$
        predict returns, $\hat{\pi}_{t+1} = \hat{\boldsymbol{f}}_{t+1}^\mathsf{T}\hat{\boldsymbol{\beta}}_{t+1} = \bar{\boldsymbol{f}}_t^\mathsf{T}\hat{\boldsymbol{\beta}}_t$
      end
      else
        set factor loadings equal to cross-sectional median, $\boldsymbol{\beta}_{T_1} = med(\mathbf{B})$
    end
  end
  initialise uncertainty matrix
    calculate mean forecast error, $\bar{e}_{T_1} = \sum_{t=1}^{T_1} \tilde{e}_t$
    calculate uncertainty matrix, $[\Omega_{\pi,T_1}]_{ij} = \sum_{t=1}^{T_1}(\tilde{e}_{i,t} - \bar{e}_{i,T_1})(\tilde{e}_{j,t} - \bar{e}_{j,T_1})$
  end
  update
    for $t \leftarrow T_1 + 1$ **to** T_3 **do**
      for *each company* **do**
        forecast errors, $\tilde{e}_t = \tilde{\boldsymbol{r}}_t - \hat{\pi}_t$
        Uncertainty matrix, $\Omega_{\pi,t} = \lambda_s\Omega_{\pi,t-1} + (1-\lambda_s)(\tilde{e}_{i,t} - \bar{e}_{i,t})(\tilde{e}_{j,t} - \bar{e}_{j,t})^\mathsf{T}$
        factor loading estimates, $\hat{\boldsymbol{\beta}}_t = \texttt{RLM}(\beta_{t-1}, (\mathbf{F}^\mathsf{T}\mathbf{F})_{t-1}^{-1}, \lambda_s, \tilde{\boldsymbol{f}}_t, \tilde{\boldsymbol{r}}_t, w(r))$
        factor mean, $\bar{\boldsymbol{f}}_t = \lambda_f\bar{\boldsymbol{f}}_{t-1} + (1-\lambda_f)\tilde{\boldsymbol{f}}_t$
        return prediction, $\hat{\pi}_{t+1} = \bar{\boldsymbol{f}}_t{}^\mathsf{T}\hat{\boldsymbol{\beta}}_t$
      end
    end
  end
end

The psuedocode details the *Initialisation*, *Update*, and *Prediction* steps of Algorithm 2 used to produce expected systematic returns. The FMP realisations are dependent variables, whilst the stock returns are response variables. Factor premiums, factor loadings, and systematic return uncertainty are initialised using offline equivalent methods. Factor premiums, and systematic return uncertainty are updated using exponential smoothing, while factor loadings are updated using the RLM filter. Predictions are equal to current parameter estimates.

Systematic return predictions ($\hat{\pi}_{t+1}$) are defined as the product of the current factor premium estimate ($\bar{\boldsymbol{f}}_t$) and the current factor loading estimate ($\hat{\boldsymbol{\beta}}_t$), excluding the estimated intercept term. Factor loadings are estimatated using an univariate RLM filter. If there is no return during a period, the updated factor loadings are set equal to the previous estimate.

The initial factor premium ($\bar{\boldsymbol{f}}_{T_1}$) and factor covariance are taken to be the sample mean and sample covariance estimates respectively of the factor realisations over the initialisation period. Periods

with missing observations are discarded when estimating the sample covariance.

During the initialisation period, the initial values for factor loadings are specified as $\beta_0 = 0$ and the initial search direction is given by $(\mathbf{F}^\top\mathbf{F})_0 = 99999\mathbf{I}$ [6]. This specification corresponds to an offline exponentially weighted robust least squares estimation. Stocks may enter the market after the initialisation period. The factor loadings for these stocks are initialised as the cross-sectional median of factor loadings at the last date of the initialisation period. For example, suppose that stock j enters the market at date $T_1 + l$, where T_1 is the last period in the initialisation phase. Then, for stock j, the initial factor loadings are given by $\boldsymbol{\beta}_{j,T_1+l} = median(\mathbf{B}_{T_1})$. The initial search direction is given by the most recent factor realisations [7].

The memory factor for the RLM filter is determined by comparing the one-step ahead forecast using the current FMP realisation and current factor loading. The subset of memory factors considered for the RLM filter is the set $\Lambda_s = \{0.9, 0.91, \ldots, 1\}$. The chosen memory factor has the minimum median of the cross-section of mean-absolute scaled (forecast) errors (MASE) over the update period. The MASE for each stock is weighted by the number of observations. Mathematically,

$$\underset{\lambda}{\arg\min}\, med\{w_i(MASE_i)\forall i = 1\ldots n\} = \{\lambda|\lambda \in \Lambda\} \tag{4.12}$$

where Λ is the prespecified set of hyperparameters, w_i is the number of historical observations in the initialisation period for the i$^{\text{th}}$ stock. The median is used to prevent outliers from influencing the memory factor choice. Outlying forecast errors are typically associated with stocks with a few number of data points.

The uncertainty matrix for systematic expected returns (Ω_π) is initialised from the forecast errors over the initialisation period, using the sample covariance estimator. During the update period, the uncertainty matrix is updated online by exponentially smoothing the deviation of the forecast errors. The elements of the uncertainty matrix corresponding to stocks that newly enter the universe during the update period do not have initial values. In such cases, the initial mean forecast error is given by the first forecast error, whilst the initial variance of forecast errors is determined using the first two forecast errors. This process for initialisation can be improved, perhaps by initialising values using some cross-sectional average.

Active Return Predictions

The algorithmic psuedo-code used to prepare active returns predictions is presented below in Algorithm 5. The function *TME* refers to the Theil and Goldberger mixed estimator. The active return predictions ($\hat{\alpha}_{BL,t+1}$) are calculated from taking the difference between the blended characteristics model predictions ($\hat{\mu}_{bl,t+1}$) and the systematic model predictions ($\hat{\pi}_{t+1}$). The characteristic model predictions are constructed as the product of realised characteristics and smoothed characteristic payoffs ($\bar{\delta}_t$). The smoothed payoffs require initialisation.

[6] If $(\mathbf{F}^\top\mathbf{F})_0 = C\mathbf{I}$ for large C, this implies $(\mathbf{F}^\top\mathbf{F})_0^{-1} \to 0$

[7] The initialisation of the factor loadings can be improved by updating the median at each time step. In a future iteration, we will consider a multiple variable extension of the RLM filter which can more conveniently incorporate this feature.

Algorithm 5: Implementation - Active Return Prediction

Data: Information variables, $\tilde{\mathbf{Z}}$
Stock returns, $\tilde{\boldsymbol{R}}$
Systematic returns predictions, $\hat{\pi}_{t+1}$
Systematic uncertainty matrices, Ω_{π}
Result: Active return predictions, $\hat{\boldsymbol{\alpha}}_{bl,t+1}$
Uncertainty matrix, $\Omega_{bl,t+1}$

begin
  initialise model Predictions
    for $t \leftarrow 2$ **to** T_1 **do**
      winsorize each predictor variable, $\boldsymbol{z}_t$
      estimate raw payoffs using robust regression, $\hat{\bar{\boldsymbol{\delta}}}_t = \texttt{RobustLS}(\boldsymbol{z}_{t-1}, \boldsymbol{r}_t)$
    end
    initialse smoothed payoffs, $\bar{\delta}_1 = \frac{1}{T_1-1}\sum_{t=2}^{T_1}\hat{\bar{\boldsymbol{\delta}}}_t$
    for $t \leftarrow 2$ **to** T_1 **do**
      calculate forecast error, $\tilde{\boldsymbol{e}}_t = \tilde{\boldsymbol{r}}_t - \hat{\boldsymbol{\mu}}_{t+1}$
      update smoothed payoff, $\bar{\boldsymbol{\delta}}_t = \lambda_a \bar{\boldsymbol{\delta}}_{t-1} + (1-\lambda_a)\hat{\bar{\boldsymbol{\delta}}}_t$
      predict characteristics model returns, $\hat{\mu}_{t+1} = \bar{\boldsymbol{\delta}}_t^{\mathsf{T}} \tilde{\boldsymbol{z}}_{t-3}$
    end
    initialise uncertainty matrix, $[\Omega_{T_1}]_{ij} = \frac{1}{T_1-2}\sum_{t=2}^{T_1}(\tilde{e}_{i,t} - \bar{e}_{i,T_1})(\tilde{e}_{j,t} - \bar{e}_{j,T_1})^{\mathsf{T}}$
    initialise smoothed payoff for update period, $\dot{\delta}_{T_1}$
  update
    for $t \leftarrow T_1 + 1$ **to** T_3 **do**
      forecast error, $\tilde{\boldsymbol{e}}_t = \tilde{\boldsymbol{r}}_t - \hat{\mu}_t$
      uncertainty matrix, $\Omega_{\mu,t} = \lambda_a \Omega_{\mu,t-1} + (1-\lambda_a)(\tilde{\boldsymbol{e}}_{i,t} - \bar{\boldsymbol{e}}_{i,t})(\tilde{\boldsymbol{e}}_{j,t} - \bar{\boldsymbol{e}}_{j,t})^{\mathsf{T}}$
      winsorize each predictor variable, $\tilde{z}_{t-1}$
      raw payoffs estimate, $\hat{\bar{\boldsymbol{\delta}}}_t = \texttt{RobustLS}(\tilde{\boldsymbol{z}}_{t-1}, \tilde{\boldsymbol{r}}_t)$
      smoothed payoff, $\bar{\boldsymbol{\delta}}_t = \lambda_a \bar{\boldsymbol{\delta}}_{t-1} + (1-\lambda_a)\hat{\bar{\boldsymbol{\delta}}}_t$
      characteristics model returns prediction, $\hat{\boldsymbol{\mu}}_{t+1} = \bar{\boldsymbol{\delta}}_t^{\mathsf{T}} \tilde{\boldsymbol{z}}_{t-3}$
      blend characteristics and sysematic predictions,
      $\boldsymbol{\mu}_{bl,t+1} = \texttt{TME}(\hat{\mu}_{t+1}, \hat{\pi}_{t+1}, \Omega_{\mu,t+1}, \Omega_{\pi,t+1})$
      blended alpha prediction, $\hat{\boldsymbol{\alpha}}_{bl,t+1} = \hat{\boldsymbol{\mu}}_{bl,t+1} - \hat{\boldsymbol{\pi}}_{t+1}$
      blended uncertainty matrix prediction, $\hat{\Omega}_{bl,t+1} = \left[\hat{\Omega}_{\mu,t+1}^{-1} + \hat{\Omega}_{\pi,t+1}^{-1}\right]^{-1}$
    end
end

The psuedocode details the *Initialisation*, *Update*, and *Prediction* steps of Algorithm 2 used to produce expected active returns. The active returns use the characteristics data and the returns data. Characteristic model payoffs and active return uncertainty are intialised offline. Characteristic models are estimated using cross-sectional robust regressions. Characteristic payoffs used for prediction are updated online using exponential smoothing. Characteristic model expected returns are blended with systematic returns. Blended active returns are used for portfolio construction.

During the initialisation period, raw estimates ($\hat{\bar{\delta}}_t$) of payoffs are estimated using a robust regression. The raw estimates are smoothed, where the sample mean is used as the initial value. The final value of the smoothed payoffs during the initialisation period is used to initialise payoffs in the update period. The optimal memory factor is determined by comparing the forecast accuracy of the characteristic predictions across the set of pre-specified memory factors. This procedure is repeated for each characteristics model.

The set of memory factors used in the hypertuning experiments is reduced to the set Λ_c =

$\{0.95, 0.96, \ldots, 1\}$. The smallest memory factor considered for the characteristics models is larger than the systematic models. The exponential weights used for smoothing the characteristics models is given by λ^n, where n is the number of observations before the current observation. The characteristic model exponential weight relies on an asymptotic approximation. In contrast, the weight of an observation in the RLM filter depends on the number of datapoints already filtered. Weights in the RLM are given by $\frac{\lambda^n}{\sum_1^N \lambda}$, where N is the number of previous datapoints already filtered.

The uncertainty matrix for the characteristic predictions ($\Omega_{\mu,t}$) are updated online. The process for initialising the characteristic uncertainty matrices is the same as for the systematic uncertainty matrix. The initial uncertainty is given by the sample covariance of forecast errors during the initialisation period. The memory factor used to update the uncertainty matrix is is the same as the memory factor used to construct the corresponding predictions. At each time step, uncertainty matrices are determined for each model.

At each time step, the characteristics model predictions are blended with the systematic return predictions using Theil's mixed estimator. The estimates from the different models are blended using the respective uncertainty matrices. The uncertainty matrices are diagonalised to avoid the issues originating from estimating a covariance matrix using asynchronous data. The active return prediction is obtained from subtracting the systematic returns from the blended estimate. The covariance matrix of the Theil estimator is used as the uncertainty matrix for the active estimate.

4.2.4 Covariance Prediction and Portfolio Construction

The psuedo-code for the portfolio construction is presented below in Algorithm 6. Thus far, we have discussed the preparation of expected return estimates. The psuedo-code below includes the preparation of the risk model for each portfolio.

The risk matrix used in each portfolio is regularised by shrinking the covariance matrix towards the identity matrix. The optimal shrinkage intensity (κ) for each covariance matrix is determined by the comparing in-sample performance of the corresponding portfolio. The set of shrinkage intensities considered is the set $\{\kappa\} = \{0, 0.1, \ldots, 1\}$. Optimal shrinkage intensities are determined such that the size of combined portfolio is approximately 130/30. More specifically, the global minimum variance portfolio is approximately long-only, whilst the combination of strategic and tactical weights should not result in excessively large short positions. The optimal shrinkage intensities are chosen from the set of admissible shrinkage parameters by ranking the average Probabilistic Sharpe Ratios (PSR) of the corresponding portfolios, over in the in-sample period. The PSR is defined in the next section.

At the portfolio construction stage, the strategic risk tolerance parameter (γ_s) needs to be specified for the systematic bets portfolio, and the active risk tolerance parameter (γ_a) for the active bets portfolio. Each risk tolerance parameter depends on the predictions, and changes over time as predictions are updated. We compare three alternative specifications for each of the risk tolerance parameters, which have the following functions:

- to fix the size of the bet portfolio: constraining $\sum |w_i| = 0.4 \implies \gamma = \frac{1}{\sum |w|}$
- to provide ex-ante single period tangency portfolio: $\gamma = \mathbf{1}^\top \Sigma^{-1} \left(\boldsymbol{\mu} - \boldsymbol{r}\right)$
- to provide ex-ante single period Maximum Absolute Risk-Adjusted (MARA) return: $\gamma = \mathbf{1}^\top \Sigma^{-1} \boldsymbol{\mu}$.

Algorithm 6: Implementation - Portfolio Construction

begin
  for $t \leftarrow 1$ **to** *end* **do**
    Construct GMV
      calculate conditional risk matrix, $\hat{\Sigma}_{t+1|\pi} = \hat{\mathbf{B}}_t \hat{\Sigma}_{f,t} \hat{\mathbf{B}}_t^\top$
      set shrinkage intensity, κ_g
      calculate shrinkage covariance estimator, $\hat{\Sigma}_{g,t+1} = \kappa \mathbf{I} + (1-\kappa)\hat{\Sigma}_{t+1}$
      calculate GMV, $\boldsymbol{w}_{g,t+1} = \frac{\hat{\Sigma}_{g,t+1}^{-1}\mathbf{1}}{\mathbf{1}^\top \hat{\Sigma}_{g,t+1}^{-1}\mathbf{1}}$
    end
    Construct SBM
      calculate unconditional covariance matrix, $\Sigma_{u,\pi,t+1} = \hat{\mathbf{B}}_t \hat{\Sigma}_{f,t} \hat{\mathbf{B}}_t^\top + diag(\Omega_{\pi,t})$
      set shrinkage intensity, κ_s
      calculate shrinkage covariance estimator, $\Sigma_{s,t+1} = \kappa_s \mathbf{I} + (1-\kappa_s)\Sigma_{u,\pi,t+1}$
      calculate unscaled systematic bets, $w^*_{s,t+1} = \frac{\hat{\Sigma}_{s,t+1}^{-1}\left(\hat{\boldsymbol{\pi}}_{t+1}\mathbf{1}^\top - \mathbf{1}\hat{\boldsymbol{\pi}}_{t+1}^\top\right)\hat{\Sigma}_{s,t+1}^{-1}}{\mathbf{1}^\top \hat{\Sigma}_{s,t+1}^{-1}\mathbf{1}}$
      set risk aversion parameter to enforce leverage constraint,
       $\sum |w| = 1 \implies \gamma_{s,t+1} = \frac{1}{\sum |w^*_{s,t+1}|}$
      calculate systematic bets portfolio, $w_{s,t+1} = \gamma_{s,t+1} w^*_{s,t+1}$
    end
    Construct ATP
      Calculate unconditional covariance matrix, $\hat{\Sigma}_{u,\mu,t+1} = \hat{\mathbf{B}}_t \hat{\Sigma}_{f,t} \hat{\mathbf{B}}_t^\top + diag(\Omega_{BL,t})$
      Set shrinkage intensity, κ_a
      Calculate shrinkage covariance estimator, $\hat{\Sigma}_{a,t+1} = \kappa_a \mathbf{I} + (1-\kappa_a)\hat{\Sigma}_{u,\mu,t+1}$
      Calculate unscaled active bets, $w^*_{a,t+1} = \frac{\hat{\Sigma}_{s,t+1}^{-1}\left(\hat{\boldsymbol{\alpha}}_{BL,t+1}\mathbf{1}^\top - \mathbf{1}\hat{\boldsymbol{\alpha}}_{BL,t+1}^\top\right)\hat{\Sigma}_{s,t+1}^{-1}}{\mathbf{1}^\top \hat{\Sigma}_{a,t+1}^{-1}\mathbf{1}}$
      Set active risk aversion parameter to enforce leverage constraint,
       $\sum |w| = 1 \implies \gamma_{a,t+1} = \frac{1}{\sum |w^*_{a,t+1}|}$
      calculate active bets portfolio, $w_{a,t+1} = \gamma_{s,t+1} w^*_{a,t+1}$
    end
  end
end

The psuedocode details the *Portfolio Construction* step of Algorithm 2. The portfolios use the conditional risk matrix and the expected returns as inputs. For each component portfolio, the risk matrix is shrunk towards the identity matrix using different shrinkage intensities. Shrinkage intensities are determined using a grid search, where the Sharpe Ratio was used as the performance metric. Leverage constraints are used to limit bet size, and are incorporated through the risk aversion parameters.

The specifications are informed by our objective of constructing portfolios that maximise the ex-post Sharpe-Ratio. The first specification restricts the largest weight of the bet portfolio in a single stock to 0.2. This specification controls ex-post realised volatility by limiting portfolio concentration. The second and third specifications are theoretically motivated. The tangecy portfolio is the optimal ex-ante portfolio for an intertemporal portfolio choice problem, when there is no predictability. The MARA portfolio provides the active portfolio with the largest information ratio, which is defined as the ratio of excess returns to excess risk.

Ex-ante, the tangency portfolio carries greater risk than the MARA portfolio. A theoretically pure specification of the tangency and MARA portfolios would require that the mean and variance estimates from each model be recombined, to specify a single risk-tolerance parameter.

Our hypertuning experiments resulted in the leverage constrained risk tolerance specification

out-performing the tangency and the MARA specification. Deeper investigation revealed that the MARA and tangency specifications sometimes resulted in infeasibly large positions. Our out-of-sample results are prepared using the leverage constrained risk tolerance specification.

The implemented calibration uses the risk tolerance to fix, rather than limit, the size of the bet portfolios. In retrospect, we recognise that fixing the bet size of the active portfolios goes against its intended purpose. The intention behind the Bayesian specification of the active returns was to control the size of positions to reflect confidence in the active return estimate. By fixing the bet size, our portfolios are effectively making bets on any predicted difference of between characteristic and systematic models, which may be caused by statistically insignificant differences. Statistically insignificant bets uneccesarily incur trading costs, especially if positions are concentrated. However, failure to constrain the bet size results in sporadically large positions.

Our implementation avoided the use of a numerical optimizer for the portfolio construction in order to have better understanding on the behaviour of the resulting weights. The simplest solution would be to change the risk tolerance to incorporate an inequality constraint, to allow small bets to be made. A more complicated constraint can be used to limit positions in individual stocks, enforcing diversification of bets, further reducing error-maximisation. This option would require a numerical optimizer. A larger consideration would be to ammend the design such that confidence in bets is reflected in the risk tolerance. Currently, the confidence in the individual predictions determines the relative, rather than absolute, size of bets.

4.3 Results

4.3.1 Performance Measures

The time series of gross cumulative performance for the algorithmic strategy and the benchmark strategies are plotted over the entire period in Figure 4.1, and over the out-of-sample period in Figure 4.3. Similarly, the net performance for the entire period is shown in Figure 4.2, while Figure 4.4 contains the net performance over the out-of-sample period. The out-of-sample performance curves are calculated by re-basing the cumulative performances indices at the start of the out-of-sample period.

The cumulative return series represent the average realised return. Risk should also be included when assessing portfolio performance. The historical risk-adjusted performance of each strategy is measured by calculating the annualised Sharpe Ratio, on both a gross return and net return basis. The historical Sharpe Ratio for a strategy is defined as:

$$\hat{SR} = \frac{\hat{\mu}_p - r_f}{\hat{\sigma}_p} \tag{4.13}$$

where the mean ($\hat{\mu_p}$) and standard deviation σ_p of returns are calculated using method of moments estimators. R_{rf} denotes the risk free-rate.

The net return over the period t (nR_t) incorporates transaction costs, and is calculated as:

$$1 + nr_t = (1 + r_t)\left(1 - c \times \sum_{i=1}^{N} |w_{i,t} - w_{i,t^-}|\right) \tag{4.14}$$

where c represents the transaction costs as a proportion of the turnover $\sum_{i=1}^{N} |w_{i,t} - w_{i,t^-}|$. The turnover is the difference between the portfolio weights at the start of following period ($w_{i,t}$) and the weights at the end of the current period w_{i,t^-}. Each strategy's net returns are calculated assuming transactional costs are 50 basis points of turnover, $c = 0.005$. For the risk-free investment, we assume that the full capital amount is turned-over every three months, and transactional costs are reduced to $c = 0.0025$. The annualised gross and net Sharpe ratios are presented in table 4.4 (in-sample) and table 4.6 (out-of-sample).

The historical Sharpe-Ratio can be viewed as a point estimate of a strategy's true Sharpe ratio. The Sharpe-ratio estimates may be specific to the testing period, and may not generalise to a different out-of-sample period. Statistical comparisons of measured performance amongst different strategies can be made to account for sample variation, and prevent backtest overfitting.

The distribution of the Sharpe-Ratio estimator is required to perform statistical tests on measured performance. If the strategy's returns are IID and Normally distributed, then the asymptotic distribution of the Sharpe ratio estimator ($\hat{SR}$) is [105]:

$$\left(\hat{SR} - SR\right) \rightarrow N\left(0, \frac{1 + \frac{1}{2}SR^2}{n}\right) \tag{4.15}$$

where SR is the strategy's true sharpe ratio, and n is the number of periods in the historical sample, expressed in the same frequency as the Sharpe Ratio.

If a strategy does not produce Normally distributed returns, then Equation 4.15 understates the standard deviation of the Sharpe-Ratio estimator. In the case of non-normal realised portfolio

returns, the Sharpe-Ratio estimator is still asymptotically normally distributed [106], but the higher moments affect the standard deviation of the returns:

$$\left(\hat{SR} - SR\right) \rightarrow N\left(0, \frac{1 + \frac{1}{2}SR^2 + \left(\frac{\mu_4 - 3}{4}SR^2 - \mu_3 SR\right)}{n}\right) \tag{4.16}$$

where μ_3 is the skewness of the stategy's returns, and μ_4 is the kurtosis of the strategy's returns. The standard deviation estimator for non-normal returns can be used to construct confidence intervals and to perform hypothesis tests in large samples.

It may be difficult to rank strategies based on point estimates whilst incorporating the associated confidence intervals. Bailey and Lopez de Prado [107] create a measure called the *Probabilistic Sharpe Ratio* which combines the Sharpe-Ratio point estimate with its standard devation:

$$PSR(SR_b) = P[\hat{SR} > SR_b] = 1 - \int_{-\infty}^{SR_b} p(\hat{SR}_b) d\hat{SR}_b \tag{4.17}$$

such that

$$PSR(SR_b) = \Phi\left[\frac{(\hat{SR} - SR_b)\sqrt{n-1}}{\sqrt{1 - \hat{\mu_3}\hat{SR} + \frac{\hat{\mu_4} - 1}{4}\hat{SR}^2}}\right] \tag{4.18}$$

where Φ is the cumulative normal distribution function, SR_b is the *benchmark* Sharpe Ratio, and the moments $\hat{\mu}_3$,$\hat{\mu}_4$, represent the sample moment estimates using the historical sample.

The PSR determines the probability that an estimate of a strategy's Sharpe ratio is greater than the benchmark Sharpe Ratio, where the estimators distribution is parameterised using plug-in estimates. In the context of decision making, calculating the PSR is analgous to calculating the power of a statistical test. In the context of backtesting a trading strategy, statistical power is reflects the ability to correctly detect an outperforming trading strategy. An investor who is more concerned about false negatives than false positives is more concerned about rejecting a profitable strategy than adopting a false strategy [8].

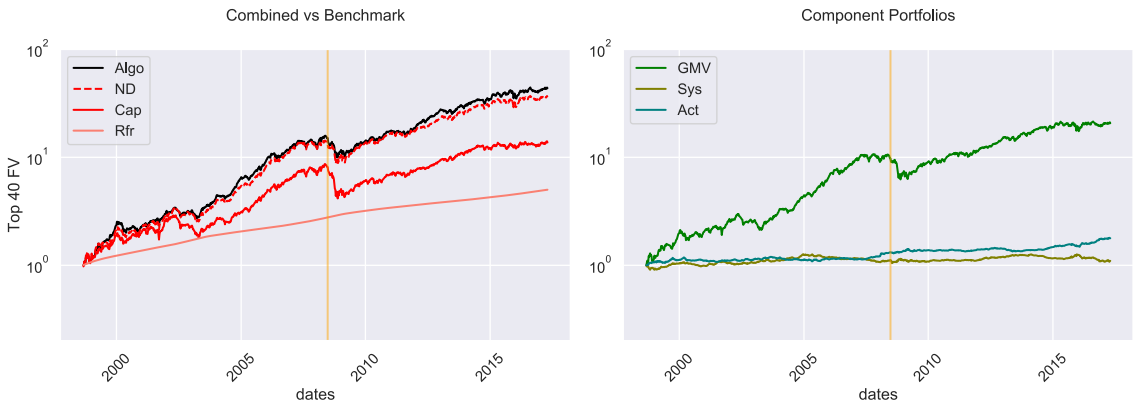
The PSR is calculated for the algorithmic strategy, against the naive diversificaiton (ND) and market-cap weighted benchmark (CAP) on a gross and net basis, for the in-sample period (Table 4.5) and the out-of-sample period (Table 4.7). The out-of-sample PSR calculations are used to carry out the second objective of this work, which is to test:

> "Does the Sharpe-Ratio of the algorithmic strategy consistently exceed the benchmark?"

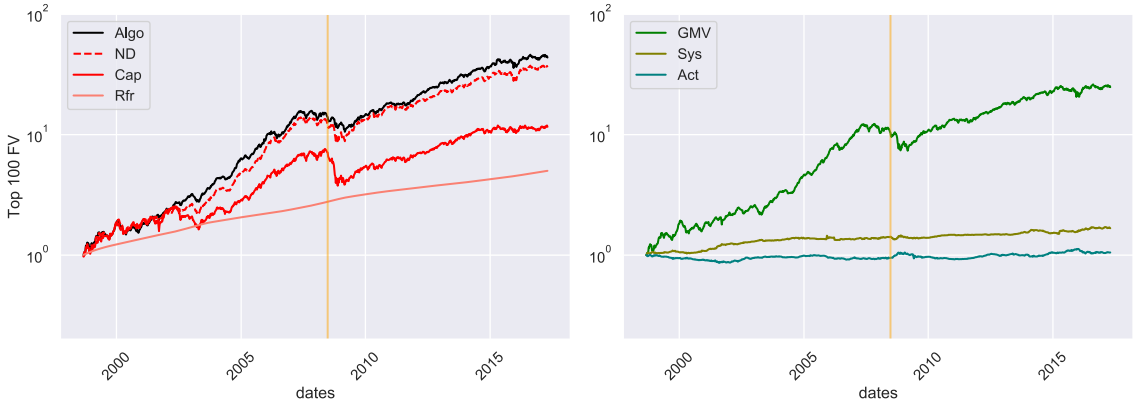
using the null hypothesis that the sample moments of the realised returns represent the true distribution of the strategy's performance.

It should be noted that the in-sample PSR results are not used to evaluate the strategy, but are only used to compare against the out-of-sample results. The in-sample PSR results are inflated by the multiple comparisons bias, which occurs as a result of the hyper-parameter tuning. Recently, Bailey et al. [14] have introduced a backtest performance measure that uses the PSR and accounts for multiple comparisons. This measure is not used in this research since our testing methodology explicitly uses an out-of-sample period which tests a single configuration of the algorithm.

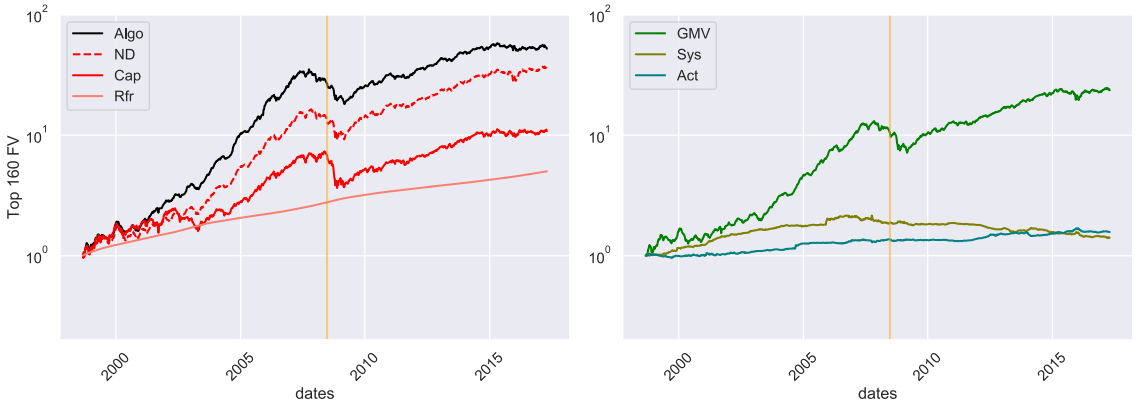
[8] A false strategy is defined as a stategy that has a true Sharpe ratio equal to or less than the benchmark, but has a historical Sharpe Ratio that exceeds the benchmark over the backtest period



(a) Top40 Universe: The algorithmic portfolio outperforms the benchmark portfolios. The cumulative returns for each component portfolio are positive over the full period. The algorithmic portfolio's returns consist mostly of returns from the GMV.



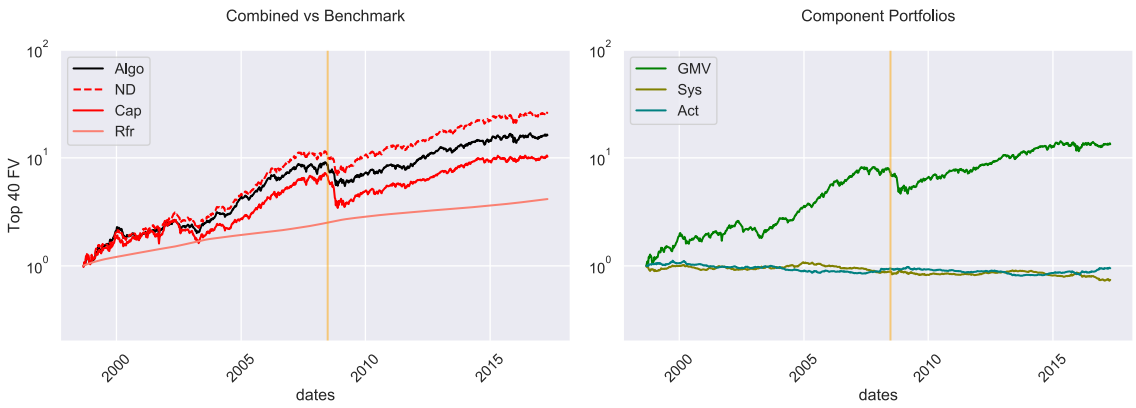
(b) Top100 Universe: The algorithmic portfolio outperforms the benchmark portfolios. The GMV and the systematic bets portfolio have positive cumulative returns over the full period. The cumulative returns for the active bets portfolio is negative in-sample and slightly positive at the end of the full period.



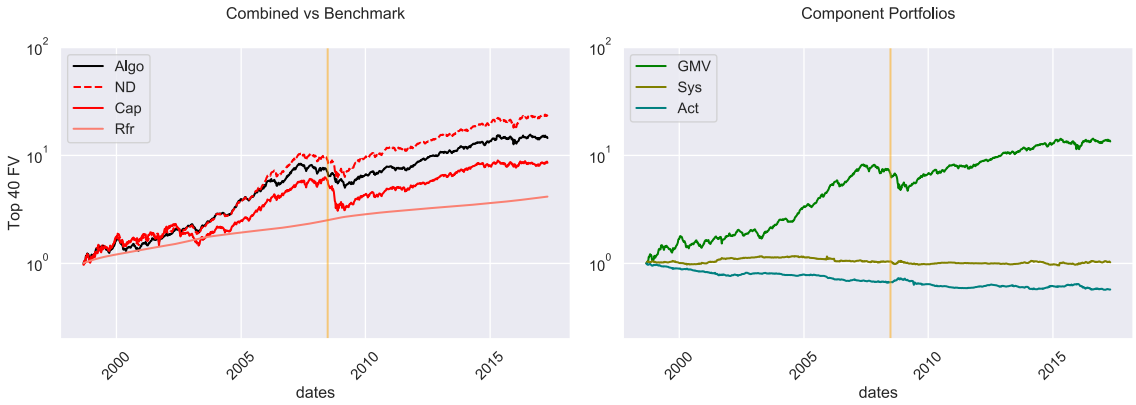
(c) Top160 Universe: The algorithmic portfolio outperforms the benchmark portfolios by a larger margin than in the Top 40 and Top 100 universes. All component portfolios have positive cumulative returns over the full period. The cumulative returns for the systematic portfolio deteriorates out-of-sample.

Figure 4.1: Comparison of Gross Cumulative Returns Over Full Historical Sample

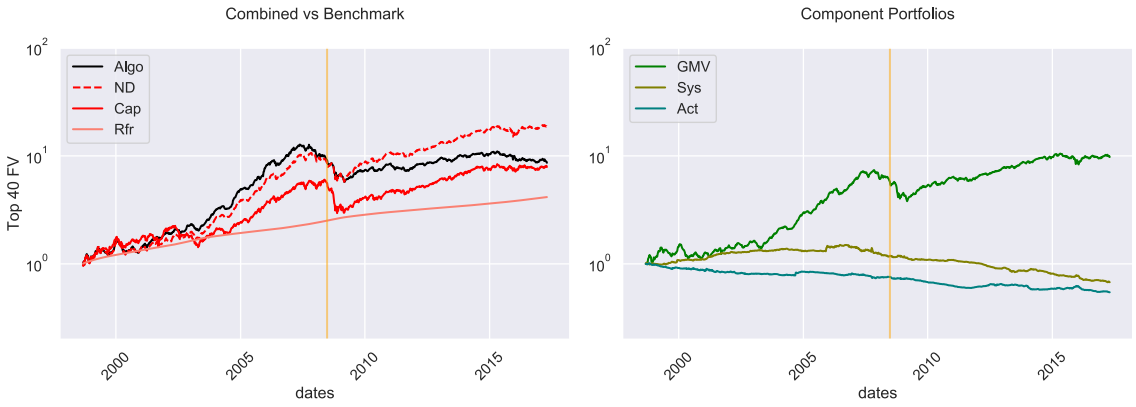
The gross cumulative returns during full historical period are compared for each of the investible universes: Top 40, Top 100, and Top 160. On the left, the algorithmic portfolio (*ALGO*) is compared against the benchmark portfolios (*ND* and *CAP*) and the risk-free instrument (*Rfr*). On the right, the component portfolios which comprise the algorithmic portfolio are plotted. *GMV* is the global minimum variance portfolio, *Sys* is the systematic bets portfolio, and *Act* is the active bets portfolio. The orange vertical line separates the in-sample period from the out-of-sample period.



(a) Top40 Universe: The algorithmic portfolio outperforms the CAP benchmark but underperforms the ND benchmark. The cumulative returns for the active bets portfolio and the systematic bets portfolio are mostly negative over the entire period.



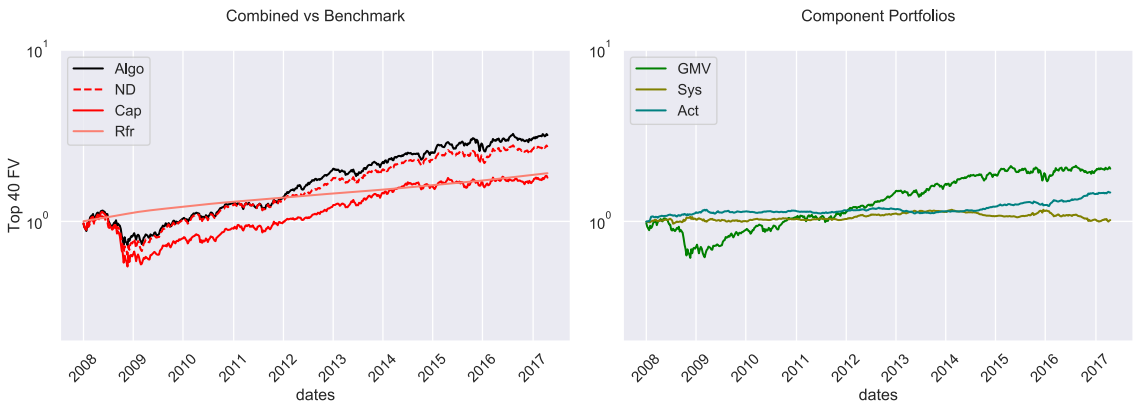
(b) Top100 Universe: The algorithmic portfolio outperforms the CAP benchmark but underperforms the ND benchmark. The in-sample cumulative returns of the systematic bets portfolio is slightly positive. The cumulative returns for the active bets portfolio and the systematic bets portfolio are negative at the end of the entire historical sample.



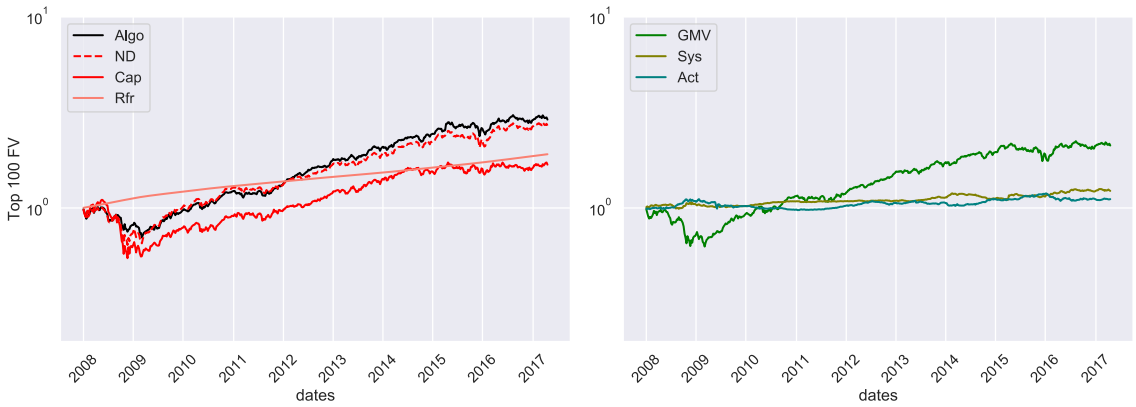
(c) Top160 Universe: The algorithmic portfolio outperforms the CAP benchmark but underperforms the ND benchmark. The cumulative returns of the systematic bets portfolio is positive in-sample but deteriorates rapidly out-of-sample. The cumulative active returns are consistently negative.

Figure 4.2: Comparison of Net Cumulative Returns Over Full Historical Sample

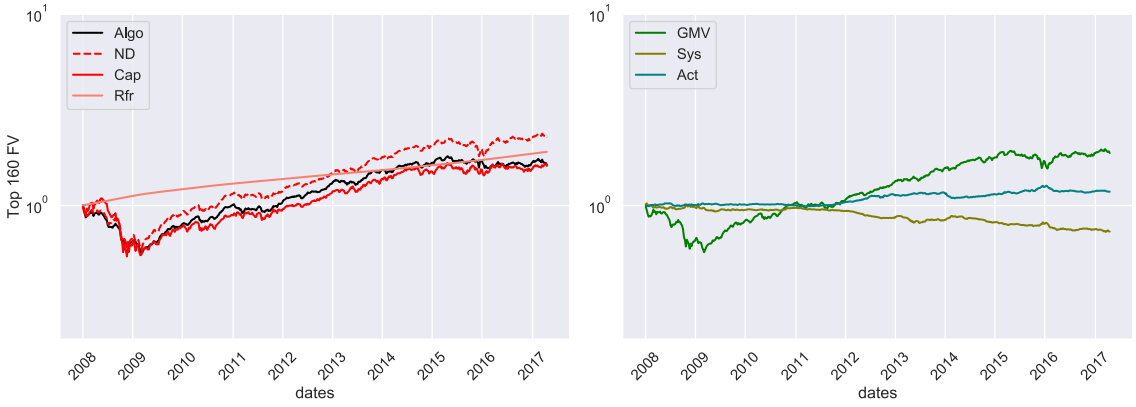
The net cumulative returns during full historical period are compared for each of the investible universes: Top 40, Top 100, and Top 160. On the left, the algorithmic portfolio (*ALGO*) is compared against the benchmark portfolios (*ND* and *CAP*) and the risk-free instrument (*Rfr*). On the right, the component portfolios which comprise the algorithmic portfolio are plotted. *GMV* is the global minimum variance portfolio, *Sys* is the systematic bets portfolio, and *Act* is the active bets portfolio. The orange vertical line separates the in-sample period from the out-of-sample period.



(a) Top40 Universe: The algorithmic portfolio has a greater cumulative return than the benchmark portfolios. The GMV has the largest contribution to the algorithmic portfolio's returns. The active returns are strongly positive at the end of the full period, whilst cumulative returns of the systematic portfolio are minimal.



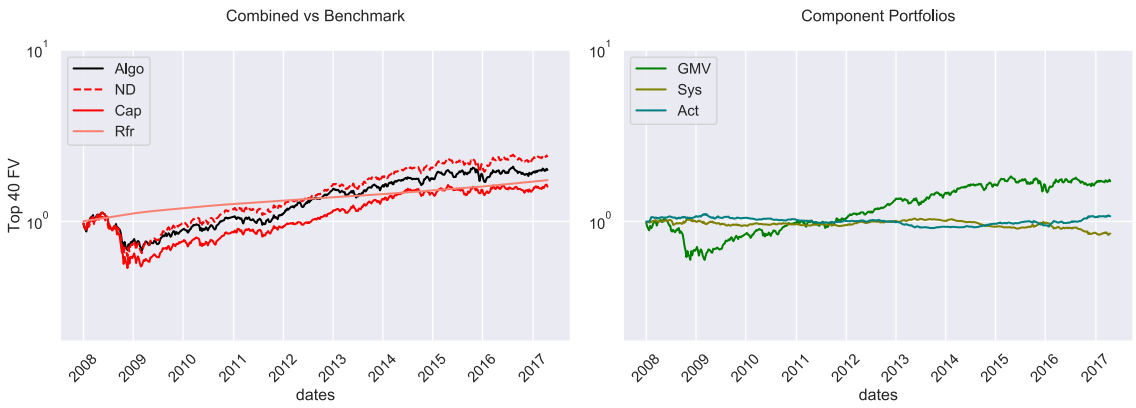
(b) Top100 Universe: The algorithmic portfolio has a greater cumulative return than the benchmark portfolios. The GMV has the largest contribution to the algorithmic portfolio's returns. The systematic bets portfolio and active bets portfolio both have positive cumulative returns.



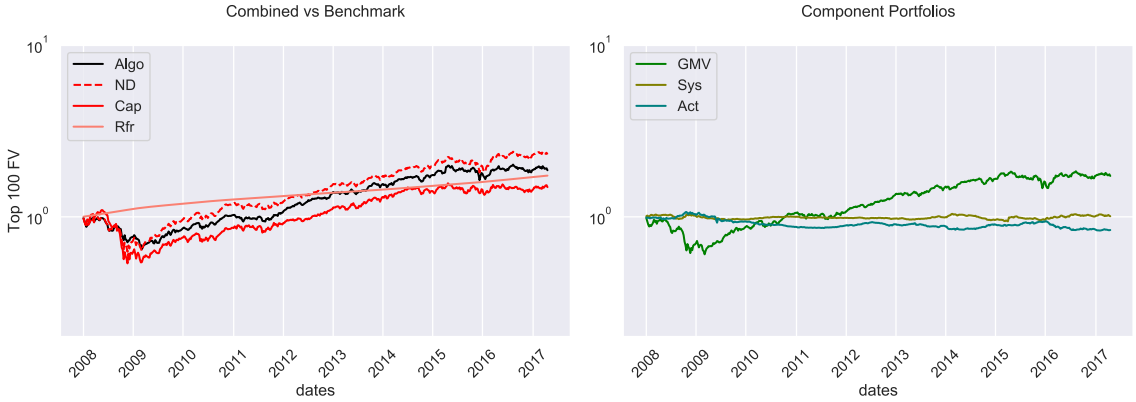
(c) Top160 Universe: The algorithmic portfolio has a greater cumulative return than the benchmark portfolios. The GMV has the largest contribution to the algorithmic portfolio's returns. The active bets portfolio have positive cumulative returns. The systematic bets portfolio earns a negative cumulative return, contrasting the strong performance during the in-sample period.

Figure 4.3: Out-of-Sample Comparison of Gross Cumulative Returns

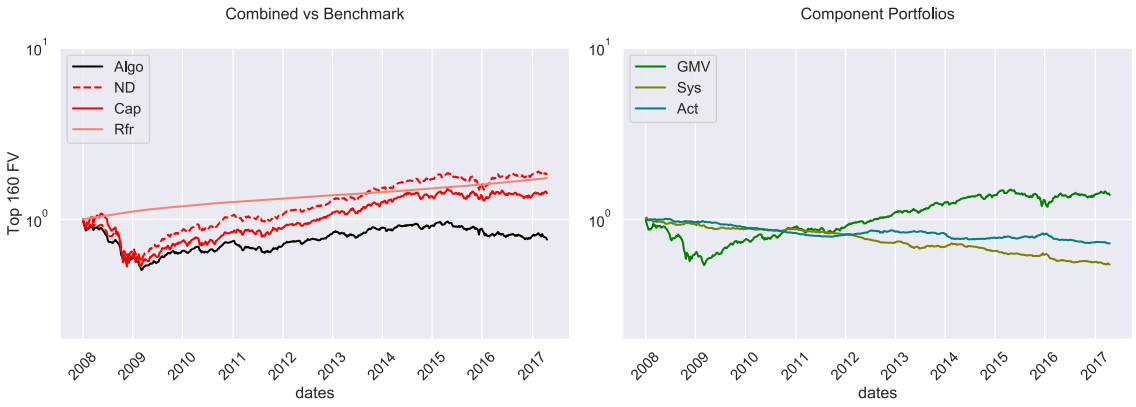
The gross cumulative returns during the out-of-sample period are compared for each of the investible universes: Top 40, Top 100, and Top 160. On the left, the algorithmic portfolio (*ALGO*) is compared against the benchmark portfolios (*ND* and *CAP*) and the risk-free instrument (*Rfr*). On the right, the component portfolios which comprise the algorithmic portfolio are plotted. *GMV* refers to the global minimum variance portfolio, *Sys* refers to the systematic bets portfolio, and *Act* refers to the active bets portfolio.



(a) Top40 Universe: The algorithmic portfolio outperforms the CAP benchmark and underperforms the ND benchmark. The GMV has the largest contribution to the algorithmic portfolio's returns. The active bets portfolio has strong positive performance, whilst cumulative returns to the systematic bets portfolio are negative.



(b) Top100 Universe: The algorithmic portfolio outperforms the CAP benchmark and underperforms the ND benchmark. The GMV has the largest contribution to the algorithmic portfolios returns. The systematic bets portfolio has minimal returns, whilst the active bets portfolio has negative returns.



(c) Top160 Universe: The algorithmic portfolio underperforms both benchmark portfolios and yields a negative cumulative net return. The The GMV has the largest positive contribution to the algorithmic portfolio's returns. The systematic and active bets portfolios both have negative returns.

Figure 4.4: Out-of-Sample Comparison of Net Cumulative Returns

The net cumulative returns during the out-of-sample period are compared for each of the investible universes: Top 40, Top 100, and Top 160. On the left, the algorithmic portfolio (*ALGO*) is compared against the benchmark portfolios (*ND* and *CAP*) and the risk-free instrument (*Rfr*). On the right, the component portfolios which comprise the algorithmic portfolio are plotted. *GMV* refers to the global minimum variance portfolio, *Sys* refers to the systematic bets portfolio, and *Act* refers to the active bets portfolio. Transaction costs are calculated on a proportional basis and are set at 50bp.

4.3.2 Portfolio Performance

Following the hypertuning experiments, the algorithmic portfolios for each universe have the following configurations:

Table 4.3: Optimal hyper-parameters for the algorithmic portfolio for each investible universe

	λ_f	λ_s	λ_a	κ_g	κ_s	κ_a	Char Model
Top 40	0.99	0.98	0.97	0.2	0.3	0.2	MV, BVTP
Top 100	0.99	0.98	0.97	0.2	0.2	0.2	BVTP, MV, CAPM β
Top 160	0.99	0.99	0.97	0.2	0.2	0.2	BVTP, MB, CAPM β, MOML, MOMS, DY

The point estimates for the annualised gross and net Sharpe Ratios over the in-sample and out-of-sample periods are presented in Table 4.4 and Table 4.6 respectively.

In-sample Performance

The in-sample gross Sharpe-Ratio estimates for the algorithmic portfolio are greater than the benchmark portfolios, over each universe. The in-sample PSR matrices suggests that a gross Sharpe ratio of the algorithmic strategy exceeds the cap-weighted portfolio at least 95% of the time, in the Top 100 and Top 160 universes. In addition, The in-sample PSR matrices suggests that a gross Sharpe ratio of the algorithmic strategy exceeds the cap-weighted portfolio at least 90% of the time, in the Top 160 universe.

The in-sample net Sharpe-Ratio estimates for the algorithmic portfolio are greater than the CAP weighted benchmark, over each universe. The in-sample net PSR suggests that the net Sharpe ratio exceeds that of the benchmark strategies with 95% frequency. The algorithmic strategy outperforms the ND benchmark over the in-sample period, only in the Top 160 universe. However, the outperformance is not statistically significant.

Table 4.4: In-sample Annualised Sharpe-Ratio Point Estimates

	Top 40			Top 100			Top 160		
	mean	volatility	SR	mean	volatility	SR	mean	volatility	SR
Gross									
ALGO	0.1768	0.1850	0.9559	0.1870	0.1521	1.2294	0.2679	0.1552	1.7270
ND	0.1738	0.1924	0.9036	0.1752	0.1602	1.0934	0.1910	0.1480	1.2906
CAP	0.1141	0.2114	0.5397	0.1033	0.2000	0.5166	0.1007	0.1975	0.5098
Net									
ALGO	0.1309	0.1848	0.7080	0.1258	0.1519	0.8281	0.1664	0.1547	1.0753
ND	0.1618	0.1923	0.8412	0.1528	0.1603	0.9535	0.1546	0.1480	1.0442
CAP	0.1058	0.2114	0.5004	0.0936	0.2000	0.4678	0.0903	0.1975	0.4572

The Algorithmic (ALGO) strategy has the highest in-sample Sharpe Ratio estimate on a gross basis. On a net basis the Algorithmic strategy has the highest in-sample Sharpe Ratio only for the Top 160 universe.

Table 4.5: In-sample PSR Matrix

	Top 40			Top 100			Top 160		
	ALGO	ND	CAP	ALGO	ND	CAP	ALGO	ND	CAP
Gross									
ALGO	0.5000	0.5617	0.8917	0.5000	0.6560	0.9823	0.5000	0.9006	0.9998
ND	0.4380	0.5000	0.8612	0.3449	0.5000	0.9547	0.1043	0.5000	0.9878
CAP	0.1056	0.1372	0.5000	0.0162	0.0418	0.5000	0.0001	0.0096	0.5000
Net									
ALGO	0.5000	0.3450	0.7329	0.5000	0.3538	0.8594	0.5000	0.5372	0.9683
ND	0.6547	0.5000	0.8458	0.6444	0.5000	0.9241	0.4639	0.5000	0.9566
CAP	0.2662	0.1527	0.5000	0.1395	0.0722	0.5000	0.0316	0.0388	0.5000

The PSR (Eq 4.17) represents the probability of the Sharpe-Ratio of a candidate strategy exceeding a benchmark strategy, and is used to conduct statistical tests. The row index indicates the candidate strategies whilst the column index indicates the benchmark strategies. The Algorithmic strategy outperforms the CAP benchmark at a 5% significance level for the Top 100 and Top 160 universes on a gross basis. On a net basis, the algorithmic strategy outperforms the CAP benchmark at a 5% significance level. PSRs are inflated by hyper-parameter tuning. Significance levels should be increased for multiple tests.

As mentioned, the in-sample results are inflated since we have not made a multiple-testing adjustment to account for all of the alternative hyper-parameter configurations. The in-sample results are included primarily for comparison with the out-of-sample results.

Out-of-sample Performance

The out-of-sample results presented in Table 4.6 show a large decrease in the performance of all strategies. The Global Financial Crisis occured during the out-of-sample period, and is the primary cause for poor performance[9]. The algorithmic strategy earns a higher Sharpe ratio than the cap-weighted portfolio in the Top 40 and Top 100 universes, on both a gross and net return basis. The algorithmic strategy also outperforms the ND benchmark in the Top 40 and Top 100 universes on a gross basis. However, the algorithmic strategy underperforms the ND benchmark in all universes after incorporating transactional costs.

The PSR matrix (Table 4.7) shows a marked deterioration in the relative performance of the algorithmic strategy from the in-sample results. The deterioration in relative performance increases as the size of the investible universe increases. In particular, the probability of gross out-performance of the algorithmic portfolio decreased by approximately 55% in the Top 160 universe. In the remaining stock universes, the algorithmic outperformance is not significant at the 5% or the 10% significance level. A multiple testing adjustment is not required since only one calibration is tested out-of-sample.

[9]There is a common narrative which suggets that the “credit freeze" motivated international investors to disinvest from emerging economies in favour of safe assets (the “flight to safety"). It is possible that the South African equities market was affected in this manner. See Baxter [108] for a detailed explanation.

Table 4.6: Out-of-sample Annualised Sharpe Ratio Point Estimates

	Top 40			Top 100			Top 160		
	mean	volatility	SR	mean	volatility	SR	mean	volatility	SR
Gross									
ALGO	0.0546	0.1791	0.3051	0.0446	0.1377	0.3239	-0.0173	0.1195	-0.1451
ND	0.0376	0.1792	0.2100	0.0368	0.1534	0.2398	0.0185	0.1392	0.1326
CAP	-0.0064	0.1900	-0.0339	-0.0134	0.1804	-0.0745	-0.0181	0.1775	-0.1020
Net									
ALGO	0.0146	0.1789	0.0818	0.0077	0.1377	0.0557	-0.0882	0.1194	-0.7386
ND	0.0333	0.1792	0.1856	0.0304	0.1534	0.1981	0.0047	0.1392	0.0337
CAP	-0.0092	0.1901	-0.0486	-0.0165	0.1805	-0.0917	-0.0217	0.1776	-0.1222

The Algorithmic (ALGO) strategy has the highest out-of-sample Sharpe Ratio estimate on a gross basis in the Top 40 and Top 100 stock universes, but has the lowest Sharpe Ratio in the Top 160 universe. On a net basis, the Algorithmic strategy outperforms the CAP benchmark in the Top 40 and Top 100 stock universes.

Table 4.7: Out-of-sample PSR matrix

	Top 40			Top 100			Top 160		
	ALGO	ND	CAP	ALGO	ND	CAP	ALGO	ND	CAP
Gross									
ALGO	0.5000	0.6134	0.8479	0.5000	0.6003	0.8856	0.5000	0.1961	0.4471
ND	0.3860	0.5000	0.7713	0.3991	0.5000	0.8302	0.8005	0.5000	0.7619
CAP	0.1502	0.2281	0.5000	0.1114	0.1681	0.5000	0.5526	0.2363	0.5000
Net									
ALGO	0.5000	0.3758	0.6544	0.5000	0.3321	0.6734	0.5000	0.0071	0.0253
ND	0.6242	0.5000	0.7623	0.6675	0.5000	0.8109	0.9907	0.5000	0.6828
CAP	0.3452	0.2370	0.5000	0.3259	0.1875	0.5000	0.9705	0.3164	0.5000

The PSR (Eq 4.17) represents the probability of the Sharpe-Ratio of a candidate strategy exceeding a benchmark strategy, and is used to conduct statistical tests. The row index indicates the candidate strategies whilst the column index indicates the benchmark strategies. The Algorithmic strategy outperforms the CAP benchmark at a 15% significance level for the Top 100 universe on a gross basis. On a net basis, the algorithmic strategy outperformance over the CAP benchmark is not statistically significant.

The net cumulative return plots (Figure 4.2 and Figure 4.4) reveal that the sytematic and active bets portfolios lose money after accounting for transaction costs. In the case of the active bets, the net loss may suggest that the cost of acting on an informational advantage exceeds the benefit. Alternatively, the net loss may reflect the previously mentioned inadequecies in execution. The net loss reiterates the potential benefit for explicitly including transaction costs in the portfolio optimization. In addition, the net loss can be reduced if the active bets size could decrease. Lastly, the net loss suggests that regularisation of the expected return estimates may be beneficial in order to reduce turnover.

The cumulative return plots suggest that the decline in performance of the systematic bets portfolio is the primary cause for the relative underperformance out-of-sample. The low realised performance of the systematic bets portfolio may indicate that there is a low differential between factor risk premums. Alternatively, the low realised performance may suggest that systematic bets are unable to capture the risk premia differentials, owing to the inaccuracy in the systematic model predictions. We make the claim that it is the latter reason. Performance deterioration from the in-sample data to the out-of-sample data is usually indicitive of in-sample overfitting. However, we claim that the in-sample models are not necessarily overfit (where noise is modeled as signal) but rather that the out-of-sample data is poorly fit because of changes in the constituents of the stock universe. The discussion follows below.

4.3.3 Investigating the Out-of-sample Results

The returns from the Factor Mimicking Portfolios (FMP) are plotted in top panel of Figure 4.5. The moving averages of the FMP returns suggest that there are differences between the factor premiums, which should be captured by the systematic bets portfolio. The systematic bets portfolio aims to capture the factor premium differences by investing in stocks. The bet made on each stock depends on the stock's exposure to each factor, which is predicted using the systematic factor model. The systematic factor model effectively ranks the stocks in terms of expected returns, which are proportional to the factor exposures. Long positions are taken on stocks with high expected returns (high exposures), and short positions are taken on stocks with low expected returns (low exposures). Errors in factor loadings are unavoidable. However, the accuracy needs to be sufficient to make an aggregated directional bet.

The ability to rank stocks and make accurate bets is compromised when there is low cross-sectional variation or high temporal variation in expected returns. When there is low cross-sectional variation in the estimated returns, estimation errors or statistically insignificant differences may be more influential than actual variation in determining the ranking. The inability to rank stocks is exacerbated when there is large temporal variation in estimates, causing unnecessary reshuffling of the stock rankings. It is worth reiterating that the amplification of estimation errors causes concentrated positions, which diminishes diversification and incurs large turnover.

The systematic returns were estimated online using a univariate RLM filter. The middle panel of Figure 4.5 demonstrates the temporal behaviour of the factor loading estimates for four randomly chosen large stocks. The left panel reflects the behaviour of the market beta estimate during the initialisation for MTN (a large stock). At the start of operating the filter, there is a short period where estimates have violent oscillations, until they stabilise. The duration of this volatile period is reduced by choosing an appropriate initial value, but estimates do not evolve smoothly. Once the estimates stabilise and appear to track the system, they evolve in a noisy but stable manner. As shown in the bottom panel of Figure 4.5, the volatility during initialisation and the subsequent temporal variation in the factor loadings of a small stock is much larger than for large stocks, causing large changes in rankings at each time period.

The prediction accuracy of systematic returns during the in-sample period had the advantage of an extensive initialisation period. As time wore on, the composition of the Top 160 universe experienced significant changes in comparison to the Top 40 and Top 100 universes. The deterioration in results from the in-sample to the out-of-sample period reflects the estimation errors attributable to new entrants into the investible set. The uncertainty matrix is used to

dampen weights of stocks which are poorly predicted. Since the uncertainty matrix is diagonal, it only offers protection against incorrect variance estimates. The misestimated factor loadings of new entrants still affect the portfolio composition through the off-diagonal (pairwise covariance) terms.

The results suggest that greater attention must be paid to estimating factor loadings accurately. Further thought must be given to choosing the initial search direction of the RLM filter, in order to dampen the initial volatility and reduce the time it takes for factor loadings to stabilise. Regularising the loadings during the initialisation period, and subsequently smoothing the factor loadings to prevent the filter from modelling the noise as signal, may provide improvement. Lastly, our implemented RLM filter used a single memory factor. The difference in temporal variation by size suggests that it may be appropriate to incorporate multiple memory factors for large, mid-cap, and small stocks.

Figure 4.5: **Top:** The exponentially smoothed FMP returns over the out-sample period indicate that it is plausible to expect differences in the systematic return premiums. **Middle Left:** The market factor loading estimate of MTN (a Top 40 stock), during the initialisation period. There is a large spike in the estimate which converges fairly quickly to an offline equivalent estimate. **Middle Right:** The market factor loading estimates of randomly selected large stocks during the update period. Market factor loadings tend to develop smoothly over time. The legend contains the ticker code of each stock. **Bottom:** the market factor loading estimates of randomly selected stocks in the Top 160 set, that do not belong to the Top 40 or Top 100 sets. Each stock shows different beta dynamics. WKF has a volatile market beta (which is characteristic of low liquidity stocks). The flat portion indicate where there is no trading. Both AET and NT1 evolve in a volatile manner, with periods of high local volatility. TDH entered the set during the in-sample period, and progresses smoothly in a manner similar to large stocks.

Chapter 5

Conclusion

The aim of this research was to develop an algorithmic framework for online portfolio management, and contrast its performance against benchmark portfolios. We ask whether or calibrated algorithm is capable of demonstrating investment skill over an out-of-sample historical period. Investment skill is defined in terms of the ability of the algorithmic strategy to consistently outperform naive diversification and market cap-weighted benchmark strategies. Investment skill is assessed by calculating the Probabilistic Sharpe Ratio which provides a statistical comparison of Sharpe-Ratio estimates over a historical sample. We summarise the outcomes of this research below.

5.1 A Framework for Online Portfolio Management

The presented algorithmic framework comprises three tasks: prediction, regularisation, and portfolio optimization. At each time step, the algorithm sequentially updates mean and covariance predictions. Expected returns are decomposed into systematic and active returns. Portfolio Inputs are regularised to reduce error-propogation in the portfolio optimisation. The user must specify which risk factors and which information variables to include. In addition, the user can specify the estimation, prediction, regularisation technique to use. Lastly, the user must specify his systematic and active risk tolerance, and can specify portfolio constraints.

The framework developed in this thesis is based on traditional asset pricing and portfolio construction theory, which is adapted for incongruent empirical findings. The asset pricing models, and the portfolio construction techniques are modified using techniques that are common in industrial practice. The motivation behind the design is discussed further below.

For the quantitative investor, financial data is a finite resource that is wasted by extensive exploratory data analysis. Hypotheses motivated solely by empirical results, reflect a single sample path of a non-iid system, and often do not generalise out-of-sample. A convincing argument to discriminate between spurious or significant correlations is to establish causality. In the absence of conducted controlled experiments, consistency with an established theoretic principal may prove useful to interpret an empirical result, and to guide an investment strategy. However, financial markets are only partially observable, and are far from being well-understood. The prevailing theory is inadequate to explain all sustained empirical phenomena.

The framework is designed according to the premise that markets are complex adaptive systems; a philosophy which is able to encompass a vast array of paradoxical theoretical and empirical results. The market is modelled using a hierarchically causal model, where low-frequency returns (interday data) are described by systematic risks and a time-varying alpha. The time-varying

alpha can have multiple causes, including nonlinearity in the risk and investor irrationality. The hierarchical model suggests that optimal portfolio construction consists of choosing a portfolio to earn systematic risk premiums, and taking additional active bets to earn the time-varying alpha.

The hierarchical model suggests that a mean-variance optimal portfolio consists of a systematic benchmark portfolio, and an active tactical portfolio. The systematic benchmark portfolio consists of the global minimum variance portfolio and a systematic bets portfolio. The global minimum variance portfolio is used to minimise the volatility of realised returns whilst earning factor risk premia. The systematic bets portfolio is used to increase systematic risk, thereby earning greater risk premiums. The algorithmic framework uses a factor model to quantify systematic risk exposure of each stock. The risk exposures determine the appropriate allocation to each stock in order to earn the desired risk premium. The active tactical portfolio is used to form bets that incorporate additional predictive information, thereby enhancing risk-adjusted returns. The active returns are modelled using firm characteristics. Empirical studies have demonstrated that firm characteristics are superior to corresponding risk factors to explain cross-sectional differences in expected returns.

A large number of adjustments and modifications are necessary to traverse the distance from a theory, or some empirical finding, to an implementable strategy. Model predictions contain errors which are maximised by the mean-variance optimizer. Prediction errors are ubiquitious owing to our partial understanding of the system and the large amount of system noise. Prediction errors are greater for stocks with a short return history, missing data, or when there is asynchronous trading. Input parameters are regularised to prevent errors from dominating the portfolio. Shrinkage Estimators are used for covariance matrices and can also be applied to systematic expected returns. Active expected returns are modelled using a Bayesian approach, in a manner comparable to the Black-Litterman model.

5.2 Implementation and Empirical Performance

A single configuration of the developed framework was implemented to test whether the algorithm was capable of outperformance. Outperformance was assessed using Probabilistic Sharpe Ratios, which calculates the probability that a historical Sharpe Ratio of the algorithmic strategy exceeds the benchmark Sharpe Ratio. Testing was conducted using weekly data from the JSE over the period 04-Jan-2008 to 21-April-2017. The testing period represents the out-of-sample data. Additionally, separate tests were carried out for net and gross returns for each of the Top 40, Top 100, and Top 160 investible universes.

The tests indicated that the algorithmic strategy can consistently outperform the Cap-weighted benchmark on a gross basis, in the Top 40 and Top 100 universes, at a significance level of approximately 15%. On a net return basis, the algorithmic strategy underperformed the Naive-Diversification benchmark across all three universes. The relative performance of the algorithmic strategy was poorest in the Top 160 universe.

We claim that the systematic bets portfolio is responsible for underperformance performance of the algorithmic strategy. We argue that the ineffecacy of the systematic bets orginates from poor prediction accuracy. The consistituents of the Top 160 universe changes significantly from the in-sample and initialisation period, to the out-of-sample period. The chosen filter that estimates the systematic model is unable to make accurate predictions on the stocks that newly enter the investible universe.

5.3 Shortcomings, and Recomendations for Future Work

Each component of the implemented algorithm could be improved through more sophisticated modeling. Considering the results, the most pertitent shortcomings appear to be:

- the failure to include transactional costs within the portfolio optimisation
- the poor systematic return predictions
- the hard constraint on the active bet size

The current framework can account for transaction costs by modifying the objective function of the portfolio optimization. Systematic returns predictions can benefit from additional regularization, particularly for determining factor loadings in the case where new stocks enter the stock universe. Exponentially smoothing factor weights, James-Stein estimation, and specifying additional memory factors for groups of stocks may be useful for improving prediction accuracy and consequently the performance of the systematic bets portfolio. Leverage constraints can be incorporated by using an optimizer for the implementation, rather than an analytical solution.

Bibliography

[1] Eugene F Fama. Market efficiency, long-term returns, and behavioral finance1. *Journal of financial economics*, 49(3):283–306, 1998.

[2] SP Kothari. Capital markets research in accounting. *Journal of accounting and economics*, 31(1-3):105–231, 2001.

[3] Burton G Malkiel. The efficient market hypothesis and its critics. *Journal of economic perspectives*, 17(1):59–82, 2003.

[4] William F Sharpe. Capital asset prices: A theory of market equilibrium under conditions of risk. *The journal of finance*, 19(3):425–442, 1964.

[5] John Lintner. The valuation of risk assets and the selection of risky investments in stock portfolios and capital budgets. *The Review of Economics and Statistics*, pages 13–37, 1965.

[6] Jan Mossin. Equilibrium in a capital asset market. *Econometrica: Journal of the econometric society*, pages 768–783, 1966.

[7] Wai Lee. *Theory and methodology of tactical asset allocation*, volume 65. John Wiley & Sons, 2000.

[8] Bin Li and Steven Chu Hong Hoi. *Online Portfolio Selection: Principles and Algorithms*. Crc Press, 2015.

[9] Marcos Lopez de Prado. Quantitative meta-strategies. 2015.

[10] Andrew W Lo. The adaptive markets hypothesis: Market efficiency from an evolutionary perspective. 2004.

[11] R David McLean and Jeffrey Pontiff. Does academic research destroy stock return predictability? *The Journal of Finance*, 71(1):5–32, 2016.

[12] Campbell R Harvey and Yan Liu. Evaluating trading strategies. 2014.

[13] Campbell R Harvey, Yan Liu, and Heqing Zhu. âĂę and the cross-section of expected returns. *The Review of Financial Studies*, 29(1):5–68, 2016.

[14] David Bailey, Jonathan Borwein, Marcos Lopez de Prado, and Qiji Zhu. Pseudo-mathematics and financial charlatanism: The effects of backtest overfitting on out-of-sample performance. 2014.

[15] J. Doyne Farmer and John Geanakoplos. The virtues and vices of equilibrium and the future of financial economics. *Complexity*, 14(3):11–38, 2009.

[16] Eugene F Fama and Kenneth R French. The capital asset pricing model: Theory and evidence. *Journal of economic perspectives*, 18(3):25–46, 2004.

[17] Nicholas Barberis and Richard Thaler. A survey of behavioral finance. *Handbook of the Economics of Finance*, 1:1053–1128, 2003.

[18] Diane Wilcox and Tim Gebbie. Hierarchical causality in financial economics. *Available at SSRN: https://ssrn.com/abstract=2544327 or http://dx.doi.org/10.2139/ssrn.2544327*.

[19] Diane L Wilcox and Tim J Gebbie. On pricing kernels, information and risk. *Investment Analysts Journal*, 44(1):1–19, 2015.

[20] Tim Gebbie and Diane Wilcox. Evidence of characteristic cross-sectional pricing of stock returns in south africa. *Unpublished manuscript*, 2007.

[21] Tim Gebbie, Justin Solms, and Andrew Beavan. *Tactical Asset Allocation Technical Document*. Peregrine Quant, 2004.

[22] Harry Markowitz. Portfolio selection. *The journal of finance*, 7(1):77–91, 1952.

[23] Aswath Damodaran. *Investment valuation: Tools and techniques for determining the value of any asset*, volume 666. John Wiley & Sons, 2012.

[24] Campbell R Harvey and Akhtar Siddique. Conditional skewness in asset pricing tests. *The Journal of Finance*, 55(3):1263–1295, 2000.

[25] John H Cochrane. *Asset pricing: Revised edition*. Princeton university press, 2009.

[26] Paul A Samuelson et al. Lifetime portfolio selection by dynamic stochastic programming. *The Review of Economics and Statistics*, 51(3):239–246, 1969.

[27] Robert C Merton. Lifetime portfolio selection under uncertainty: The continuous-time case. *The review of Economics and Statistics*, pages 247–257, 1969.

[28] Robert Merton. Optimum consumption and portfolio rules in a continuous-time model. *Journal of Economic Theory*, 3(4):373–413, 1971.

[29] Ray Ball and Philip Brown. An empirical evaluation of accounting income numbers. *Journal of accounting research*, pages 159–178, 1968.

[30] Eugene F Fama and Kenneth R French. Dividend yields and expected stock returns. *Journal of financial economics*, 22(1):3–25, 1988.

[31] Eugene F Fama and Kenneth R French. The cross-section of expected stock returns. *the Journal of Finance*, 47(2):427–465, 1992.

[32] Eugene F Fama and Kenneth R French. Common risk factors in the returns on stocks and bonds. *Journal of financial economics*, 33(1):3–56, 1993.

[33] Narasimhan Jegadeesh and Sheridan Titman. Returns to buying winners and selling losers: Implications for stock market efficiency. *The Journal of finance*, 48(1):65–91, 1993.

[34] Christos H Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1998.

[35] John H Cochrane. Portfolio advice for a multifactor world. Technical report, National Bureau of Economic Research, 1999.

[36] Nicholas Barberis. Investing for the long run when returns are predictable. *The Journal of Finance*, 55(1):225–264, 2000.

[37] Michael J Brennan, Eduardo S Schwartz, and Ronald Lagnado. Strategic asset allocation. *Journal of Economic Dynamics and Control*, 21(8-9):1377–1403, 1997.

[38] Michael W Brandt. Estimating portfolio and consumption choice: A conditional euler equations approach. *The Journal of Finance*, 54(5):1609–1645, 1999.

[39] Richard O Michaud. The markowitz optimization enigma: Is âĂŸoptimizedâĂŹoptimal? *Financial Analysts Journal*, 45(1):31–42, 1989.

[40] Richard Roll. A mean/variance analysis of tracking error. *Journal of portfolio management*, 18(4):13–22, 1992.

[41] J David Jobson and Bob Korkie. Estimation for markowitz efficient portfolios. *Journal of the American Statistical Association*, 75(371):544–554, 1980.

[42] Michael J Best and Robert R Grauer. On the sensitivity of mean-variance-efficient portfolios to changes in asset means: some analytical and computational results. *The review of financial studies*, 4(2):315–342, 1991.

[43] Philippe Jorion. Bayes-stein estimation for portfolio analysis. *Journal of Financial and Quantitative Analysis*, 21(3):279–292, 1986.

[44] Olivier Ledoit and Michael Wolf. Improved estimation of the covariance matrix of stock returns with an application to portfolio selection. *Journal of empirical finance*, 10(5):603–621, 2003.

[45] Brian Munro and David Bradfield. Putting the squeeze on the sample covariance matrix for portfolio construction. *Investment Analysts Journal*, 45(1):47–62, 2016.

[46] Lundi Matoti. *Building a statistical linear factor model and a global minimum variance portfolio using estimated covariance matrices*. PhD thesis, University of Cape Town, 2009.

[47] Michael W Brandt. Portfolio choice problems. *Handbook of financial econometrics*, 1:269–336, 2009.

[48] Raymond Kan and Guofu Zhou. Optimal portfolio choice with parameter uncertainty. *Journal of Financial and Quantitative Analysis*, 42(3):621–656, 2007.

[49] Jun Tu and Guofu Zhou. Markowitz meets talmud: A combination of sophisticated and naive diversification strategies. *Journal of Financial Economics*, 99(1):204–215, 2011.

[50] Ravi Jagannathan and Tongshu Ma. Risk reduction in large portfolios: Why imposing the wrong constraints helps. *The Journal of Finance*, 58(4):1651–1683, 2003.

[51] Victor DeMiguel, Lorenzo Garlappi, and Raman Uppal. Optimal versus naive diversification: How inefficient is the 1/n portfolio strategy? *The review of Financial studies*, 22(5):1915–1953, 2007.

[52] Fischer Black and Robert Litterman. Global portfolio optimization. *Financial analysts journal*, 48(5):28–43, 1992.

[53] Henry Theil and Arthur S Goldberger. On pure and mixed statistical estimation in economics. *International Economic Review*, 2(1):65–78, 1961.

[54] Stephen A Ross. *Neoclassical finance*. Princeton University Press, 2009.

[55] Stephen A Ross. The arbitrage theory of capital asset pricing. In *HANDBOOK OF THE FUNDAMENTALS OF FINANCIAL DECISION MAKING: Part I*, pages 11–30. World Scientific, 2013.

[56] Philip H Dybvig and Stephen A Ross. Arbitrage. In *Finance*, pages 57–71. Springer, 1989.

[57] J Michael Harrison and David M Kreps. Martingales and arbitrage in multiperiod securities markets. *Journal of Economic theory*, 20(3):381–408, 1979.

[58] Kerry Back and Stanley R Pliska. On the fundamental theorem of asset pricing with an infinite state space. *Journal of mathematical economics*, 20(1):1–18, 1991.

[59] J Michael Harrison and Stanley R Pliska. A stochastic calculus model of continuous trading: complete markets. *Stochastic processes and their applications*, 15(3):313–316, 1983.

[60] Ravi Bansal and Salim Viswanathan. No arbitrage and arbitrage pricing: A new approach. *The Journal of Finance*, 48(4):1231–1262, 1993.

[61] Lars Peter Hansen and Scott F Richard. The role of conditioning information in deducing testable restrictions implied by dynamic asset pricing models. *Econometrica: Journal of the Econometric Society*, pages 587–613, 1987.

[62] Michael J Brennan, Tarun Chordia, and Avanidhar Subrahmanyam. Alternative factor specifications, security characteristics, and the cross-section of expected stock returns1. *Journal of Financial Economics*, 49(3):345–373, 1998.

[63] Wayne E Ferson and Campbell R Harvey. Conditioning variables and the cross section of stock returns. *The Journal of Finance*, 54(4):1325–1360, 1999.

[64] Wayne E Ferson and Campbell R Harvey. The variation of economic risk premiums. *Journal of Political Economy*, 99(2):385–415, 1991.

[65] Eric Ghysels. On stable factor structures in the pricing of risk: do time-varying betas help or hurt? *The Journal of Finance*, 53(2):549–573, 1998.

[66] Andrew W Lo and A Craig MacKinlay. Data-snooping biases in tests of financial asset pricing models. *The Review of Financial Studies*, 3(3):431–467, 1990.

[67] Robert C Merton. An intertemporal capital asset pricing model. *Econometrica: Journal of the Econometric Society*, pages 867–887, 1973.

[68] Douglas T Breeden. An intertemporal asset pricing model with stochastic consumption and investment opportunities. *Journal of Financial Economics*, 7(3):265–296, 1979.

[69] Mark Rubinstein. An aggregation theorem for securities markets. *Journal of Financial Economics*, 1(3):225–244, 1974.

[70] John F Muth. Rational expectations and the theory of price movements. *Econometrica: Journal of the Econometric Society*, pages 315–335, 1961.

[71] Michael C Jensen. Some anomalous evidence regarding market efficiency. *Journal of Financial Economics*, 6(2/3):95–101, 1978.

[72] Eugene F Fama. Efficient capital markets: Ii. *The journal of finance*, 46(5):1575–1617, 1991.

[73] Rolf W Banz. The relationship between return and market value of common stocks. *Journal of financial economics*, 9(1):3–18, 1981.

[74] Sanjoy Basu. Investment performance of common stocks in relation to their price-earnings ratios: A test of the efficient market hypothesis. *The journal of Finance*, 32(3):663–682, 1977.

[75] Laxmi Chand Bhandari. Debt/equity ratio and expected common stock returns: Empirical evidence. *The journal of finance*, 43(2):507–528, 1988.

[76] Mike Ward and Chris Muller. Empirical testing of the capm on the jse. *Investment Analysts Journal*, 41(76):1–12, 2012.

[77] P van Rensburq and M Robertson. Style characteristics and the cross-section of jse returns. *Investment Analysts Journal*, 32(57):7–15, 2003.

[78] P van Rensburg and M Robertson. Size, price-to-earnings and beta on the jse securities exchange. *Investment Analysts Journal*, 32(58):7–16, 2003.

[79] D Strugnell, E Gilbert, and R Kruger. Beta, size and value effects on the jse, 1994–2007. *Investment Analysts Journal*, 40(74):1–17, 2011.

[80] CJ Auret and RA Sinclaire. Book-to-market ratio and returns on the jse. *Investment Analysts Journal*, 35(63):31–38, 2006.

[81] Eugene F Fama and Kenneth R French. Multifactor explanations of asset pricing anomalies. *The journal of finance*, 51(1):55–84, 1996.

[82] Robert A Haugen, Nardin L Baker, et al. Commonality in the determinants of expected stock returns. *Journal of Financial Economics*, 41(3):401–439, 1996.

[83] PG Basiewicz and CJ Auret. Feasibility of the fama and french three factor model in explaining returns on the jse. *Investment Analysts Journal*, 39(71):13–25, 2010.

[84] Josef Lakonishok, Andrei Shleifer, and Robert W Vishny. Contrarian investment, extrapolation, and risk. *The journal of finance*, 49(5):1541–1578, 1994.

[85] Kent Daniel and Sheridan Titman. Evidence on the characteristics of cross sectional variation in stock returns. *the Journal of Finance*, 52(1):1–33, 1997.

[86] Sanford J Grossman and Joseph E Stiglitz. On the impossibility of informationally efficient markets. *The American economic review*, 70(3):393–408, 1980.

[87] Fischer Black. Noise. *The journal of finance*, 41(3):528–543, 1986.

[88] John H Cochrane. Presidential address: Discount rates. *The Journal of finance*, 66(4):1047–1108, 2011.

[89] Lennart Ljung and Torsten Söderström. *Theory and practice of recursive identification*. MIT press, 1983.

[90] Torsten Söderström and Petre Stoica. System identification. 1989.

[91] Bin Li and Steven CH Hoi. Online portfolio selection: A survey. *ACM Computing Surveys (CSUR)*, 46(3):35, 2014.

[92] Fayyaaz Loonat and Tim Gebbie. Learning zero-cost portfolio selection with pattern matching. *PloS one*, 13(9):e0202788, 2018.

[93] David Bailey and Marcos Lopez de Prado. The deflated sharpe ratio: correcting for selection bias, backtest overfitting and non-normality. *Harvard University-RCC*, 2014.

[94] LarsPeter Hansen and Thomas J Sargent. Robust control and model uncertainty. *American Economic Review*, 91(2):60–66, 2001.

[95] Paul Glasserman and Xingbo Xu. Robust portfolio control with stochastic factor dynamics. *Operations Research*, 61(4):874–893, 2013.

[96] Randy OâĂŹToole. The black–litterman model: active risk targeting and the parameter tau. *Journal of Asset Management*, 18(7):580–587, 2017.

[97] Larry Wasserman. *All of statistics: a concise course in statistical inference*. Springer Science & Business Media, 2013.

[98] Attilio Meucci. The black-litterman approach: Original model and extensions. *Shorter version in, THE ENCYCLOPEDIA OF QUANTITATIVE FINANCE, Wiley*, 2010.

[99] Y. Zou, S. C. Chan, and T. S. Ng. A recursive least m-estimate (rlm) adaptive filter for robust filtering in impulse noise. *IEEE Signal Processing Letters*, 7(11):324–326, Nov 2000.

[100] Dieter Hendricks and Diane Wilcox. A reinforcement learning extension to the almgren-chriss framework for optimal trade execution. In *Computational Intelligence for Financial Engineering & Economics (CIFEr), 2104 IEEE Conference on*, pages 457–464. IEEE, 2014.

[101] Ser-Huang Poon and Clive WJ Granger. Forecasting volatility in financial markets: A review. *Journal of economic literature*, 41(2):478–539, 2003.

[102] Johannesburg Stock Exchange. *JSE Limited Listings Requirements*.

[103] Futuregrowth Asset Management. *Equity Valuation and Investment*, 2005.

[104] Long Zhao, Deepayan Chakrabarti, and Kumar Muthuraman. Portfolio construction by mitigating error amplification: The bounded-noise portfolio. 2018.

[105] Andrew W Lo. The statistics of sharpe ratios. *Financial analysts journal*, 58(4):36–52, 2002.

[106] Elmar Mertens. Comments on variance of the iid estimator in lo (2002). Technical report, Working paper, 2002.

[107] David H Bailey and Marcos Lopez de Prado. The sharpe ratio efficient frontier. 2012.

[108] Roger Baxter. The global economic crisis and its impact on south africa and the countryâĂŹs mining industry. *Challenges for monetary policy-makers in emerging markets*, pages 105–116, 2009.

Appendix A

Literature Review

A.1 MV Frontier

The investor's faces the following constrained optimization problem:

$$\max_{w} \boldsymbol{w}^{\mathsf{T}}\boldsymbol{\mu} - \frac{\gamma}{2}\boldsymbol{w}^{\mathsf{T}}\Sigma\boldsymbol{w} \tag{A.1}$$

subject to the full investment constraint,

$$\boldsymbol{w}^{\mathsf{T}}\mathbf{1} = 1 \tag{A.2}$$

To find the optimum, form the Lagrangian:

$$L = \boldsymbol{w}^{\mathsf{T}}\boldsymbol{\mu} - \frac{\gamma}{2}\boldsymbol{w}^{\mathsf{T}}\Sigma\boldsymbol{w} - \lambda(\boldsymbol{w}^{\mathsf{T}}\mathbf{1} - 1) \tag{A.3}$$

where λ is a Lagrangian multiplier for the full-investment constraint. The first-order conditions are:

$$\begin{aligned} \nabla_w L = \boldsymbol{\mu} - \gamma\Sigma\boldsymbol{w} - \lambda\mathbf{1} = 0 \\ \implies \boldsymbol{w}^* = \frac{1}{\gamma}\Sigma^{-1}(\boldsymbol{\mu} - \lambda\mathbf{1}) \end{aligned} \tag{A.4}$$

and

$$\begin{aligned} \frac{\partial L}{\partial \lambda} = -(\boldsymbol{w}^{\mathsf{T}}\mathbf{1} - 1) = 0 \\ \implies \boldsymbol{w}^{\mathsf{T}}\mathbf{1} = 1 \end{aligned} \tag{A.5}$$

Substituting A.4 into A.5, we obtain the following expression for λ,

$$\lambda = \frac{\mathbf{1}^{\mathsf{T}}\Sigma^{-1}\boldsymbol{\mu}}{\mathbf{1}^{\mathsf{T}}\Sigma^{-1}\mathbf{1}} - \frac{\gamma}{\mathbf{1}^{\mathsf{T}}\Sigma^{-1}\mathbf{1}} \tag{A.6}$$

Substituting A.6 into A.4, we arrive at the solution for the optimal portfolio:

$$\boldsymbol{w}^* = \left(1 - \frac{\mathbf{1}^{\mathsf{T}}\Sigma^{-1}\boldsymbol{\mu}}{\gamma}\right)\frac{\Sigma^{-1}\mathbf{1}}{\mathbf{1}^{\mathsf{T}}\Sigma^{-1}\mathbf{1}} + \left(\frac{\mathbf{1}^{\mathsf{T}}\Sigma^{-1}\boldsymbol{\mu}}{\gamma}\right)\frac{\Sigma^{-1}\boldsymbol{\mu}}{\mathbf{1}^{\mathsf{T}}\Sigma^{-1}\mu} \tag{A.7}$$

A.2 Mean-Variance Frontier with Risk-Free Asset

Suppose the investible set includes the risk-free asset, with expected return r_f and standard deviation equal to zero. Let w_0 be the portfolio weight invested in the risk-free asset, such that $w_0 = (1 - \boldsymbol{w}^\top \mathbf{1})$. Then, the investor's constrained optimization problem becomes:

$$\begin{aligned} &\min_{w} \frac{\gamma}{2} \boldsymbol{w}^\top \Sigma \boldsymbol{w} \\ \text{subject to} \quad &\boldsymbol{w}^\top \boldsymbol{\mu} + (1 - \boldsymbol{w}^\top \mathbf{1}) r_f = \mu_p \end{aligned} \tag{A.8}$$

The optimisation problem is solved by forming the Lagrangian:

$$L = \boldsymbol{w}^\top \Sigma \boldsymbol{w} + \lambda \left(\mu_p - \boldsymbol{w}^\top \boldsymbol{\mu} - (1 - \boldsymbol{w}^\top \mathbf{1}) r_f \right) \tag{A.9}$$

Then, the first-order conditions are:

$$\nabla_w L = \Sigma \boldsymbol{w} - \lambda \left(\boldsymbol{\mu} - \mathbf{1} r_f \right) = 0 \tag{A.10}$$

$$\text{and} \quad \frac{\partial L}{\partial \lambda} = \mu_p - \boldsymbol{w}^\top \boldsymbol{\mu} - (1 - \boldsymbol{w}^\top \mathbf{1}) r_f = 0 \tag{A.11}$$

Solving for λ yields,

$$\lambda = \frac{\mu_p - r_f}{\left(\boldsymbol{\mu} - \mathbf{1} r_f \right)^\top \Sigma^{-1} \left(\boldsymbol{\mu} - \mathbf{1} r_f \right)} \tag{A.12}$$

Substituting A.12 into A.10 results in the solution for the optimal portfolio in the risky assets:

$$\boldsymbol{w}^* = \Sigma^{-1} \left(\boldsymbol{\mu} - r_f \right) \left[\frac{\mu_p - r_f}{\left(\boldsymbol{\mu} - \mathbf{1} r_f \right)^\top \Sigma^{-1} \left(\boldsymbol{\mu} - \mathbf{1} r_f \right)} \right] \tag{A.13}$$

A.3 Tactical Asset Allocation

The derivation for eq. 3.12 provided in [7] is presented below.

A mean-variance efficient portfolio has the solution:

$$\boldsymbol{w} = \left(1 - \frac{\mathbf{1}^\top \Sigma^{-1} \boldsymbol{\mu}}{\gamma} \right) \frac{\Sigma^{-1} \mathbf{1}}{\mathbf{1}^\top \Sigma^{-1} \mathbf{1}} + \left(\frac{\mathbf{1}^\top \Sigma^{-1} \boldsymbol{\mu}}{\gamma} \right) \frac{\Sigma^{-1} \boldsymbol{\mu}}{\mathbf{1}^\top \Sigma^{-1} \boldsymbol{\mu}} \tag{A.14}$$

Let $\boldsymbol{w}_g$ denote the global minimum variance portfolio. Then,

$$\boldsymbol{w}_g = \frac{\Sigma^{-1} \mathbf{1}}{\mathbf{1}^\top \Sigma^{-1} \mathbf{1}} \tag{A.15}$$

Eq. A.14 can be re-written as:

$$\boldsymbol{w} = \boldsymbol{w}_g + \frac{\Sigma^{-1}}{\gamma} \left(\boldsymbol{\mu} - \mathbf{1} \frac{\mathbf{1}^\top \Sigma^{-1} \boldsymbol{\mu}}{\mathbf{1}^\top \Sigma^{-1} \mathbf{1}} \right) \tag{A.16}$$

Noticing that $\mathbf{1}^\top \Sigma^{-1} \boldsymbol{\mu}$ is a scalar, then the result $\mathbf{1}^\top \Sigma^{-1} \boldsymbol{\mu} = \boldsymbol{\mu}^\top \Sigma^{-1} \mathbf{1}$ can be used to write eq. A.16 as:

$$\boldsymbol{w} = \boldsymbol{w}_g + \frac{1}{\gamma} \frac{\Sigma^{-1}}{\mathbf{1}^1 \Sigma^{-1} \mathbf{1}} \left(\boldsymbol{\mu} \mathbf{1}^\top \Sigma^{-1} \mathbf{1} - \mathbf{1} \boldsymbol{\mu}^\top \Sigma^{-1} \mathbf{1} \right) \tag{A.17}$$

Making the substitution $\boldsymbol{\mu} = \boldsymbol{\mu} + \boldsymbol{\pi} - \boldsymbol{\pi}$,

$$\begin{aligned} \boldsymbol{w} = \boldsymbol{w}_g + \frac{1}{\gamma}\frac{\Sigma^{-1}}{\mathbf{1}^{\mathsf{T}}\Sigma^{-1}\mathbf{1}}\left(\boldsymbol{\pi}\mathbf{1}^{\mathsf{T}}\Sigma^{-1}\mathbf{1} - \mathbf{1}\boldsymbol{\pi}^{\mathsf{T}}\Sigma^{-1}\mathbf{1}\right) \\ + \frac{1}{\gamma}\frac{\Sigma^{-1}}{\mathbf{1}^{\mathsf{T}}\Sigma^{-1}\mathbf{1}}\left((\boldsymbol{\mu}-\boldsymbol{\pi})\mathbf{1}^{\mathsf{T}}\Sigma^{-1}\mathbf{1} - \mathbf{1}(\boldsymbol{\mu}-\boldsymbol{\pi})^{\mathsf{T}}\Sigma^{-1}\mathbf{1}\right) \end{aligned} \tag{A.18}$$

Which can be simplified to:

$$\boldsymbol{w} = \boldsymbol{w}_g + \frac{1}{\gamma}\frac{\Sigma^{-1}(\boldsymbol{\pi}\mathbf{1}^{\mathsf{T}} - \mathbf{1}\boldsymbol{\pi}^{\mathsf{T}})\Sigma^{-1}}{\mathbf{1}^{\mathsf{T}}\Sigma^{-1}\mathbf{1}} + \frac{1}{\gamma}\frac{\Sigma^{-1}\left((\boldsymbol{\mu}-\boldsymbol{\pi})\mathbf{1}^{\mathsf{T}} - \mathbf{1}(\boldsymbol{\mu}-\boldsymbol{\pi})^{\mathsf{T}}\right)\Sigma^{-1}}{\mathbf{1}^{\mathsf{T}}\Sigma^{-1}\mathbf{1}} \tag{A.19}$$

A.4 CAPM derivation

If we consider the single-period risk-averse investor, solving the Euler equation provides an expression in the same form as 2.36:

$$\begin{aligned} U'(C_t, C_{t+1}) &= -U'(C_t) + \delta U'(C_{t+1}) r_{t+1} \\ \implies 1 &= \delta \mathbb{E}\left[\frac{U'(C_{t+1})}{U'(C_{t+1})} r_{t+1}\right] \end{aligned} \tag{A.20}$$

where C is the utility-maximising consumption.
Thus, the Neoclassical set-up allows us to specify the stochastic discount factor for the investor in terms of his state-dependent intertemporal rate of substitution:

$$M_{t+1} = \frac{U'(C_{t+1})}{U'(C_{t+1})} \tag{A.21}$$

If investors are assumed to have quadratic utilities at each time period,

$$U(C_t) = -\frac{1}{2}(C_0 - C_t) \tag{A.22}$$

where c_0 is an arbitrary constant representing the satiation point, and the coefficient $-\frac{1}{2}$ is included for convenience.
Then the marginal rate of substitution (and equivalently the discount factor) is given by:

$$M_{t+1} = \delta\frac{(C_0 - C_{t+1})}{(C_0 - C_t)} \tag{A.23}$$

It is assumed that investors receive an intial endowment, have no additional income, and consume all terminal wealth:

$$W_{t+1} = r^p_{t+1}(W_t - C_t) \tag{A.24}$$

$$\text{and} \quad W_{t+1} = C_{t+1} \tag{A.25}$$

where r^p_{t+1} is the rate of return on a fully-invested aggregate wealth optimal portfolio.
Thus, by direct substitution the discount factor can be expressed as a linear function of the return on the aggregate wealth portfolio:

$$M_{t+1} = \delta\left[\frac{c_0}{c_0 - c_t} - \frac{W_t - c_t}{c_0 - c_t} r^p_{t+1}\right] \tag{A.26}$$

Since the pricing kernel is linear in the factor, it has an equivalent linear factor expression for expected returns:

$$\mathbb{E}[r_{t+1}] = r_f + \beta(r^W_{t+1} - \mathbb{E}[r^W_{t+1}]) \tag{A.27}$$

By considering the investors preferences, the CAPM discovers that the identity of the factor model as the aggregate wealth portfolio. Since all investors' wealth is held in assets, the aggregate wealth portfolio is the same as the market portfolio.

A.5 Woodbury's Matrix Identity

Let $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$ be appropriately sized matrices such that the product $\mathbf{BCD}$ and sum $\mathbf{A}+\mathbf{BCD}$ exist. Furthermore, $\mathbf{A}$ and $\mathbf{C}$ are assumed to be invertible. Then,

$$[\mathbf{A}+\mathbf{BCD}]^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}[\mathbf{DA}^{-1}\mathbf{B}+\mathbf{C}^{-1}]^{-1}\mathbf{DA}^{-1} \tag{A.28}$$

A.6 Derivation of the Recursive M-Estimator

Let $\boldsymbol{y}$ be the column vector of the response with n observations. The value at time i is denoted $y(i)$. Let $\boldsymbol{x}_i$ be a row vector of regressors at time i. Let ρ be a function that is nonegative, monotonic, and symmetric. Let e be the estimation residual.
Formally, we want to solve:

$$\min_{\boldsymbol{\beta}} \mathscr{J}(\boldsymbol{\beta}) = \sum_{i=1}^{n} \lambda^{n-i} \rho(y_i - \boldsymbol{x}_i \boldsymbol{\beta}) \tag{A.29}$$

Taking the derivative of the cost function $\mathscr{J}$, and setting to zero, we have:

$$\nabla_{\boldsymbol{\beta}} \mathscr{J} = \sum_{i=1}^{n} \lambda^{n-i} \rho'(y_i - \boldsymbol{x}_i \boldsymbol{\beta}) \boldsymbol{x}_i^{\mathsf{T}} \tag{A.30}$$

Setting the quantity,

$$\rho'(y_i - \boldsymbol{x}_i \boldsymbol{\beta}) = w(e_i) e_i \tag{A.31}$$

We have,

$$\nabla_{\boldsymbol{\beta}} \mathscr{J} = \sum_{i=1}^{n} \lambda^{n-i} w(e_i) \boldsymbol{x}_i^{\mathsf{T}} e_i = 0 \tag{A.32}$$

Which, after expanding e_i, provides the standard normal equations for the m-estimator:

$$\sum_{i=1}^{n} \lambda^{n-i} w(e_i) \boldsymbol{x}_i^{\mathsf{T}} \boldsymbol{x}_i \boldsymbol{\beta} = \sum_{i=1}^{n} \lambda^{n-i} w(e_i) \boldsymbol{x}_i^{\mathsf{T}} y_i \tag{A.33}$$

$$\mathbf{R}_n \boldsymbol{\beta} = \mathbf{p_n} \tag{A.34}$$

Thus,

$$\boldsymbol{\beta} = \mathbf{R}_n^{-1} \mathbf{p_n} \tag{A.35}$$

Where $\mathbf{R_n}$ is the auto-correlation matrix and $\mathbf{p_n}$ is the correlation vector between the response and the input at time n.
We now find a recursive relationship for $\boldsymbol{\beta_n}$, where the subscript indicates the time.

First, we show the recursive relationship of the matrix $\mathbf{R_n}$, and vector $\mathbf{p_n}$

$$\mathbf{R_n} = \sum_{i=1}^{n} \lambda^{n-i} w(e_i) \boldsymbol{x}_i^{\mathsf{T}} \boldsymbol{x}_i \tag{A.36}$$

$$\mathbf{R_n} = \lambda \mathbf{R}_{n-1} + w(e_n) \boldsymbol{x}_n^{\mathsf{T}} \boldsymbol{x}_n \tag{A.37}$$

and similarly,

$$\mathbf{p_n} = \lambda p_{n-1} + w(e_n) \boldsymbol{x}_n^{\mathsf{T}} y(n) \tag{A.38}$$

Solving for the recursion of $\boldsymbol{\beta}$:

$$\begin{aligned}
\boldsymbol{\beta}_n &= \mathbf{R}_n^{-1}\mathbf{P_n} \\
&= \mathbf{R}_n^{-1}(\lambda \mathbf{p_{n-1}} + w(e_n)\boldsymbol{x}_n^{\mathsf{T}} y(n)) \\
&= \mathbf{R}_n^{-1}(\lambda \mathbf{R}_{n-1}\boldsymbol{\beta}_{n-1} + w(e_n)\boldsymbol{x}_n^{\mathsf{T}} y(n)) \\
&= \mathbf{R}_n^{-1}\left[\ (\mathbf{R_n} - w(e_n)\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{x}_n)\boldsymbol{\beta}_{n-1} + w(e_n)\boldsymbol{x}_n^{\mathsf{T}} y(n)\right] \\
&= \boldsymbol{\beta}_{n-1} + \mathbf{R}_n^{-1} w(e_n)\boldsymbol{x}_n^{\mathsf{T}}(y(n) - \boldsymbol{x}_n\boldsymbol{\beta}_{n-1}) \\
&= \boldsymbol{\beta}_{n-1} + \mathbf{G}_n e_n
\end{aligned} \tag{A.39}$$

$\mathbf{G}_n$ is referred to the gain at time n and is calculated by:

$$\mathbf{G}_n = \mathbf{R}_n^{-1} w(e_n)\boldsymbol{x}_n^{\mathsf{T}} \tag{A.40}$$

It is advantageous to recursively calculate R_n^{-1}. To do this, we use the Woodbury's matrix inversion lemma:
If we let

$$\begin{aligned}
\mathbf{A} &= \lambda \mathbf{R}_{n-1} \\
\mathbf{B} &= \boldsymbol{x}_n^{\mathsf{T}} \\
\mathbf{C} &= w(e_n) \\
\mathbf{D} &= \boldsymbol{x}_n
\end{aligned} \tag{A.41}$$

Then,

$$\begin{aligned}
R_n^{-1} &= \frac{1}{\lambda}[\mathbf{R}_{n-1} + \boldsymbol{x}_n^{\mathsf{T}} w(e_n)\boldsymbol{x}_n]^{-1} \\
R_n^{-1} &= \frac{1}{\lambda}\left(\mathbf{R}_{n-1}^{-1} - \mathbf{R}_{n-1}^{-1}\boldsymbol{x}_n^{\mathsf{T}}\left[\boldsymbol{x}_n\mathbf{R}_{n-1}\boldsymbol{x}_n^{\mathsf{T}} + \frac{\lambda}{w(e_n)}\right]^{-1}\boldsymbol{x}_n\mathbf{R}_{n-1}^{-1}\right)
\end{aligned} \tag{A.42}$$

Appendix B

Code Patterns

The code patterns can be downloaded from https://github.com/apaskara/Algo_Invest.

B.1 Workflow

```
%% Algo_Invest Script File
% Authors: AB Paskaramoorthy
% Version: AlgoInvest-20170904-004aa

% # Problem Specification:  ...
% # Data Specification: EPC_HML_SMB_J203_FMP_workspace_R2007a
% # Configuration Control:
%        userpath/MATLAB/Algo_Invest
%        userpath/MATLAB/Algo_Invest/data
%        userpath/MATLAB/Algo_Invest/scripts
%        userpath/MATLAB/Algo_Invest/functions
%        userpath/MATLAB/Algo_Invest/html
%        userpath/MATLAB/Algo_Invest/test_code
%        userpath/MATLAB/Algo_Invest/junk
%        userpath/MATLAB/Algo_Invest/workspace
% # Version Control: ... (SVN/Tortoise SVN/CVS)
% # References: ...
% # Current Situation:
%   Split into initialization and update sections
%   Split components into different scripts
%   took out the index application to the betas
%
%
%
% # Future Situation:
% EWMA characteristic payoffs
%
% Uses:
%
%% Notes:
% 1. FIXME -
% 2. TBDL -
%       1. Low pass filter on raw data to smooth out spikes (and errors)
%       2. Definition and Application of Entropy Measures
%       3. Robust transformation on loss functions
%       4. Can move FMP construction to data prep - alternatively, create
%       FMP function
%       5. Check forecast accuracy corresponds to loss portfolio performance
%       6. Beta and R_0 intialisation
%
% 4. Use a startup.m file where possible to manage your paths/java libraries


%% 1. Data Description
% fts
% 1 DY          -   dividend yield
% 2 MV          -   market capitalisation (public float?)
% 3 NT          -   ??
% 4 PE          -   price-to-earnings ratio (earnings reporting frequency?)
% 5 PTBV        -   price to book value (book value reporting frequency?)
% 6 RI          -   monthly prices of each ticker
% 7 VO          -   volume traded
% rfr           - risk-free rate proxy - (90-day NCD)

%% 3. Clear workspace
close all;
clear;
clc;
warning off;

%% 4. Set Paths (implement configuration control)
% Specify the path directory containing the data files
```

```
% addpath c:\Users\user\Documents\MATLAB
% use the userpathstr.
%
%  # Project Path:              userpath/MATLAB/Algo_Invest
%  # Data Path:                 userpath/MATLAB/Algo_Invest/data
%  # Script Files:              userpath/MATLAB/Algo_Invest/scripts
%  # Classes and Function:      userpath/MATLAB/Algo_Invest/functions
%  # Test Code:                 userpath/MATLAB/Algo_Invest/test_code
%  # Published Output:          userpath/MATLAB/Algo_Invest/html
%  # Legacy Code:               ...
%

userpathstr = userpath;
userpathstr = userpathstr(~ismember(userpathstr,';'));
% Project Paths:
% -- Modify this line to be your prefered project path ----->
projectpath = 'Algo_Invest';
% <-----------------------------------------------------------
addpath(fullfile(userpathstr,projectpath,'data'));
addpath(fullfile(userpathstr,projectpath,'functions'));
addpath(fullfile(userpathstr,projectpath,'scripts'));
addpath(fullfile(userpathstr,projectpath,'test_code'));
addpath(fullfile(userpathstr,projectpath,'html'));
addpath(fullfile(userpathstr,projectpath,'plots'));
addpath(fullfile(userpathstr,projectpath,'workspace'));

%% 1. Load Data
% prep data in Data_prep_002aa

% load data
load('procdata1_w');

% indexes
indexes     = {'top40', 'top100', 'top160'};
nindex      = [40 100 160];

% choose index
% for ii = 1:numel(indexes)
% uni=indexes{ii};
uni = 'top160';
findex = db_procdata.indexes.(uni);
index = double(fts2mat(db_procdata.indexes.(uni)));
index(index==0) = NaN;

%% split data into initialization and update
% end of in-sample period
insample = datenum('28-Dec-2007');
insample = find(findex.dates==insample);

% split into initialisation and update (one-third / two-third)
ini = [1, round(insample/3) ];
updt = [round(insample/3) + 1, insample];

%% Sort companies by characteristics into quantiles
% index has not been lagged (findex has)
% characteristics have been lagged by 4 weeks

% we lag variables from published financial records.
clear nb nm nm3;
% create temp local characteristic variables
mv = lagts(db_procdata.char_data{2},4); %mv
bvtp = lagts(db_procdata.char_data{4},4); %bvtp

% quintile sorts - value
nb = ntile(transpose((index .* fts2mat(bvtp))),3,'descend');
db_quintile.sorts.nb = fints(bvtp.dates,transpose(nb),fieldnames(bvtp,1),bvtp.freq,bvtp.desc);
% quintile sort - size - 2 quintiles
nm = ntile(transpose(index .* fts2mat(mv)),2,'descend');
db_quintile.sorts.nm = fints(mv.dates,transpose(nm),fieldnames(mv,1),mv.freq,mv.desc);
% quintile sort - size - 3 quintiles
nm3 = ntile(transpose(index .* fts2mat(mv)),3,'descend');
db_quintile.sorts.nm3 = fints(mv.dates,transpose(nm3),fieldnames(mv,1),mv.freq,mv.desc);
% delete temp variables
clear mv bvtp;
clear nb nm nm3;


%% Create Fama-French Factor portfolios
% lag portfolios to apply return
% eg. at time t, intersection portfolio return: HS(t-1)*ret(t)

nbx = fts2mat(db_quintile.sorts.nb);
nmx = fts2mat(db_quintile.sorts.nm);
% nm3x = fts2mat(db_quintile.sorts.nm3);
% create the index for the different bins
small  = (nmx==2);
big    = (nmx==1);
high   = (nbx==1);
medium = (nbx==2);
low    = (nbx==3);
% create LS (low small) intersection of nbvtp and nmv equally weighted
db_quintile.intersection.LS = low & small;
db_quintile.intersection.MS = medium & small;
db_quintile.intersection.HS = high & small;
db_quintile.intersection.LB = low & big;
```

```
db_quintile.intersection.MB = medium & big;
db_quintile.intersection.HB = high & big;

% Compute the normalizations
db_quintile.norms.normLS = transpose(sum(transpose(db_quintile.intersection.LS)));
db_quintile.norms.normMS = transpose(sum(transpose(db_quintile.intersection.MS)));
db_quintile.norms.normHS = transpose(sum(transpose(db_quintile.intersection.HS)));
db_quintile.norms.normLB = transpose(sum(transpose(db_quintile.intersection.LB)));
db_quintile.norms.normMB = transpose(sum(transpose(db_quintile.intersection.MB)));
db_quintile.norms.normHB = transpose(sum(transpose(db_quintile.intersection.HB)));

% clear local variables
clear nbx nmx nm3x small big high medium low;

%% Create the indices for the intersection portfolios
% retrieve stock returns
ret = db_procdata.returns.ret;
retx = fts2mat(db_procdata.returns.ret);

%create temp vars
tmp_inputs1 = {'LS', 'MS', 'HS', 'LB', 'MB', 'HB'};
[LS, MS, HS, LB, MB, HB] = create_vars(db_quintile.intersection, tmp_inputs1);
tmp_inputs2 = {'normLS', 'normMS', 'normHS', 'normLB', 'normMB', 'normHB'};
[normLS, normMS, normHS, normLB, normMB, normHB] = create_vars(db_quintile.norms, tmp_inputs2);

% compute the returns - equal  weighting
db_quintile.intersection.rLS = transpose(nansum(transpose(retx .* LS))) ./ normLS;
db_quintile.intersection.rMS = transpose(nansum(transpose(retx .* MS))) ./ normMS;
db_quintile.intersection.rHS = transpose(nansum(transpose(retx .* HS))) ./ normHS;
db_quintile.intersection.rLB = transpose(nansum(transpose(retx .* LB))) ./ normLB;
db_quintile.intersection.rMB = transpose(nansum(transpose(retx .* MB))) ./ normMB;
db_quintile.intersection.rHB = transpose(nansum(transpose(retx .* HB))) ./ normHB;
form_returns = [db_quintile.intersection.rLS, db_quintile.intersection.rMS, ...
    db_quintile.intersection.rHS, db_quintile.intersection.rLB, ...
    db_quintile.intersection.rMB, db_quintile.intersection.rHB];
db_quintile.intersection.retfts = fints(ret.dates,form_returns,{'LS','MS','HS','LB','MB','HB'},ret.freq,'Returns');

% clear temp variables
clear retx form_returns ret;
clear tmp_inputs1 LS MS HS LB MB HB;
clear tmp_inputs2 normLS normMS normHS normLB normMB normHB;


%% Create Fama-French Factor Replicating Portfolios

% local temp variables
tmp_inputs = {'rHB','rHS','rMB','rMS','rLB','rLS'};
[rHB, rHS, rMB, rMS, rLB, rLS] = create_vars(db_quintile.intersection, tmp_inputs);
% Create the quarterly sorts of H(igh)-M(inus)-L(ow) Book-to-Market
rHML = (1/2)*((rHB + rHS) - (rLB + rLS));
% Create the quarterly sorts of S(mall)-M(inus)-B(ig) on Log-size
rSMB = (1/3)*((rHS + rMS + rLS) - (rHB + rMB +rLB));
% Create a market index (market cap weighted - balanced portfolio)
mvx = exp(index .* fts2mat(lagts(db_procdata.char_data{2})));
retx = fts2mat(db_procdata.returns.ret);
normMV = transpose(nansum(transpose(mvx)));
% normND = min(nansum(index,2), nansum(~isnan(retx),2)); % min of index or populated returns
normND = nansum(~isnan(index.*retx),2);

% rMKT = (transpose(nansum(transpose(retx .* mvx))) ./ normMV) - fts2mat(db_procdata.returns.rfr);
rMKT = (transpose(nansum(transpose(retx .* mvx))) ./ normMV) - fts2mat(db_procdata.returns.rfr);
% naive diversification
rIND = nansum(index.*retx,2)./normND;
% naive weights
ndw = index ./ normND;
% naive turonver
ndTO = turnover2(ndw, retx);
% net return assuming 50 basis point proportional transaction costs
c = 0.005;
netIND = rIND + log(1-c*ndTO);

% cap weighted
rCAP = (transpose(nansum(transpose(retx .* mvx))) ./ normMV);
% cap weights
capw = mvx ./ normMV;
% cap weights turnover - will be zero if we consider whole market, but
% there are rebalancing costs associated with change in the index
% constituents
capTO = turnover2(capw, retx);
% net return
netCAP = rCAP + log(1-c*capTO);

% store returns struct
db_factors.returns.rHML = rHML;
db_factors.returns.rSMB = rSMB;
db_factors.returns.rMKT = rMKT;
db_factors.returns.rIND = rIND;
db_factors.returns.rCAP = rCAP;
db_factors.returns.rIND = rIND;
db_factors.returns.netIND = netIND;
db_factors.returns.netCAP = netCAP;

% clear vars
clear mvx retx normMV;
```

```
% risk-free rate
rfr = fts2mat(db_procdata.returns.rfr);

% Store factor returns in separate fints structure
fmpfts = fints(db_procdata.returns.ret.dates,[rHML, rSMB, rMKT],{'HML','SMB','MKT'},db_procdata.returns.ret.freq,'Returns');
indpft = fints(db_procdata.returns.ret.dates,[rIND,rCAP, netIND, netCAP, rfr ],{'ND','CAP', 'netND', 'netCAP', 'rfr'},db_procda
prcfmp = fillts(fmpfts(12:end),0);
prcfmp(1) = 0;
prcfmp = exp(cumsum(prcfmp));

prcind = fillts(indpft(12:end),0);
prcind(1) = 0;
prcind = exp(cumsum(prcind));

% store vars
db_factors.returns.fmpfts = fmpfts;
db_factors.returns.prcfmp = prcfmp;
db_factors.returns.indpft = indpft;
db_factors.returns.prcind = prcind;
% store turnover
db_factors.returns.ndTO =  fints(db_procdata.returns.ret.dates, ndTO ,{'ND'},db_procdata.returns.ret.freq,'Turnover');
db_factors.returns.capTO = fints(db_procdata.returns.ret.dates, capTO ,{'CAP'},db_procdata.returns.ret.freq,'Turnover');

%clear variables
clear tmp_inputs rHB rHS rMB rMS rLB rLS rHML rSMB rMKT fmpfts prcfmp;
clear capTO capw ndTO ndw netCAP netIND normND rCAP rfr rIND prcind indpft;

%% Specify Factor Models
mdl_type = 'ol';
db_mdlindx.(mdl_type){1,1} = {[3]};
db_mdlindx.(mdl_type){1,2} = {'MKT'};
db_mdlindx.(mdl_type){2,1} = {[1 2 3]};
db_mdlindx.(mdl_type){2,2} = {'HML','SMB','MKT'};
clear mdl_type;


%% Choose Lambda for factor models

% iniisation is done by estimating factor models and taking mean \
% median of betas at final date of iniisation
% 0.91 is best lambda

% Initialisation - choose lambda
% calculate MSE for different lambdas
%{
% forgetting factor
lambda = 0.88:0.01:1;
% outputs and inputs
ydata = fts2mat(db_procdata.returns.eret(ini(1): ini(2)));
fdata = fts2mat(db_factors.returns.fmpfts(ini(1): ini(2)));
% fit for different lambdas
fit = NaN*ones(length(lambda),3);
% factor index
fct_ind = [1 2 3];

for i = 1:length(lambda)
    model = online_ts_reg(fdata(:,fct_ind),ydata, 'y', {'lambda',lambda(i)}, {'weight', 'huber'});
    fit(i,1) = lambda(i);
    fit(i,2) = model.MSE;
    fit(i,3) = model.MFE;
end
save('FMfit_lambda', 'fit' );

% clear fit model ydata fdata lamba fct_ind;

%}

%% Initialise Factor models

% estimate factor models
% forgetting factor
lambda = 0.98;
% outputs and inputs (in-sample)
ydata = fts2mat(db_procdata.returns.eret(ini(1): ini(2)));
fdata = fts2mat(db_factors.returns.fmpfts(ini(1): ini(2)));
% define mdl_type
mdl_type = 'ol';
% choose weight function
% wfunc = {'huber', 'std'};
wfunc = {'huber'};
% models definedeted by factor selection - see db_mdlindx.(mdl_type)
for k = 1:size(db_mdlindx.(mdl_type),1)
% for k = 2
    fct_ind = cell2mat(db_mdlindx.(mdl_type){k,1});    % select factors
%     db_models.rw{1,k} = rolling_ts_reg(fdata(:,fct_ind),ydata,mdof); %rolling
    for j = 1:length(wfunc)
        db_models.ol{j,k} = online_ts_reg(fdata(:,fct_ind),ydata, 'y', {'lambda',lambda}, {'weight', wfunc{j}}); %online
    end
end

clear fct_ind ydata fdata k mdl_type wfunc;

%% store beta in fints to keep track of dates
% set temp vars
ret = db_procdata.returns.ret(ini(1): ini(2)) ;
```

```
% choose model type
mdl_type = 'ol'; % rw - rolling window; ol - online
all_factors = {'bBIAS', 'bHML', 'bSMB', 'bMKT'};
% index over iniisation period
Iindex = index(ini(1): ini(2),:);
% Iindex = ones(size(index(ini(1): ini(2),:))); single indexing

for k = 1:size(db_mdlindx.(mdl_type),1)
% for k = 2
    clear fct_ind;
    fct_ind = cell2mat(db_mdlindx.(mdl_type){k,1}); % choose factors
    fct_ind = [1 (fct_ind + 1)]; % include bias
    for i = 1:length(fct_ind) % for each factor
        clear fname;
        fname = all_factors{fct_ind(i)}; %fints title
        for j = 1:size(db_models.(mdl_type),1) % include in model index?
            db_models.(mdl_type){j,k}.beta(:,:,i) = db_models.(mdl_type){j,k}.beta(:,:,i) ;
            beta = squeeze(db_models.(mdl_type){j,k}.beta(:,:,i)); %beta matrix
            db_models.(mdl_type){j,k}.fbeta{i} = fints(ret.dates, beta, fieldnames(ret,1), ret.freq, fname); %store as fints
            % mean beta
            db_models.(mdl_type){j,k}.b_mean(i) = nanmean(Iindex(end,:) .* beta(end,:));
            % median beta
            db_models.(mdl_type){j,k}.b_median(i) = nanmedian(Iindex(end,:) .* beta(end,:));
        end
    end
end
% clear vars
clear ret all_factors fct_ind fname beta k i;

%% store betas for cross-sectional regressions


% online CAPM beta
db_procdata.beta_data{1} = db_models.ol{1}.fbeta{2};
db_procdata.beta_index{1} = 'online CAPM MKT beta';
% online FF3F HML beta
db_procdata.beta_data{2} = db_models.ol{2}.fbeta{2};
db_procdata.beta_index{2} = 'online FF3F HML beta';
% online FF3F SMB beta
db_procdata.beta_data{3} = db_models.ol{2}.fbeta{3};
db_procdata.beta_index{3} = 'online FF3F SMB beta';
% online FF3F MKT beta
db_procdata.beta_data{4} = db_models.ol{2}.fbeta{4};
db_procdata.beta_index{4} = 'online FF3F MKT beta';
% online FF3F MKT alpha
db_procdata.beta_data{5} = db_models.ol{2}.fbeta{1};
db_procdata.beta_index{5} = 'online FF3F alpha';

% save to disk
filepath = fullfile(userpathstr,'Algo_Invest/workspace/Betas');
ol_capm_beta = db_models.ol{1}.fbeta{2};
ol_ff_alpha = db_models.ol{2}.fbeta{1};
save(fullfile(filepath,strcat('ol_capm_beta_',uni)), 'ol_capm_beta');
save(fullfile(filepath, strcat('ol_ff_alpha_',uni)), 'ol_ff_alpha');

clear filepath ol_capm_beta ol_ff_alpha;

%% Factor Realisations means and covariances

% lambda
lambda = 0.99;
% initial means and covariance
fretx = fts2mat(db_factors.returns.fmpfts(ini(1):ini(2)));
[fcov0,~,fmu0] = ewma_cov(lambda,fretx, {'mu0',  nanmean(fretx)}, {'COV_0', nancov(fretx)});

% initial R matrix (hessian matrix (search direction?))
for i = 1:size(db_mdlindx.ol,1)
    clear fct_ind inputs;
    fct_ind = cell2mat(db_mdlindx.ol{i,1});
    inputs = [ones(size(fretx,1),1) fretx(:, fct_ind)];
    R0{i} = ewma_autocorr(lambda, inputs, {'COV_0', inputs'*inputs});
end
clear fretx fct_ind inputs;

%% Initialisation Systematic Model Forecasts (expected returns only)

% factor model initial forecast
lambda = 0.98;
% Uindex = index(ini(1):ini(2),:);
ret = db_procdata.returns.eret(ini(1):ini(2));
rfr = db_procdata.returns.rfr(ini(1):ini(2));
mdl_type = 'ol';
% index over initialisation period
Iindex = index(ini(1): ini(2),:);
% Iindex = ones(size(index(ini(1): ini(2),:))); single indexing


for i = 2 %% FF3F
    % factor selection
    fct_ind = cell2mat(db_mdlindx.ol{i,1});
    for j = 1:size(db_models.ol,1)
        % stock factor beta
        beta = db_models.ol{j,i}.beta;
        % pre-allocation
        exp_ret = NaN*ones(size(beta,1)+1,size(beta,2));
```

```
        sys_error = NaN*ones(size(beta,1)+1,size(beta,2));
        % factor premiums
        fact_prem = fts2mat(db_factors.returns.fmpfts(ini(1):ini(2)+1));
        fact_prem = fact_prem(:,fct_ind); % select factors
        [fcov,~,fmu] = ewma_cov(0.99, fact_prem);
        % for each time period
        for t = 1:size(beta,1)
            % time t index
            index_t = Iindex(t,:);
            % time t beta - exclude intercept for factor expected return
            beta_t = reshape(squeeze(beta(t,:,2:end)), size(index_t,2),[]);
            % expected return for t+1
            exp_ret(t+1,:) = (beta_t * fmu(t,:)')' + repmat(fts2mat(rfr(t)),1,size(exp_ret,2));
            % factor innovation at t+1 (not sure about the index_t, just copying from previous line) - also fact_prem t+1 and f
            sys_error(t+1,:) = (beta_t * (fact_prem(t+1,:) - fmu(t,:))')';
        end
        % store as fints
        db_models.(mdl_type){j,i}.fexp = lagts(fints(ret.dates, exp_ret(2:end,:), fieldnames(ret,1), ret.freq),1,NaN) + repmat(
        db_models.(mdl_type){j,i}.sys_error = fints(ret.dates, sys_error(1:end-1,:), fieldnames(ret,1), ret.freq);
    end
end

clear t fct_ind fact_prem ret mdl_type beta beta_t exp_ret index_t;

%% Factor Model Update Phase

% forgetting factor
lambda = 0.98;
% outputs and inputs
% ydata = fts2mat(db_procdata.returns.eret(updt(1):updt(2)));
ydata = fts2mat(db_procdata.returns.eret(updt(1):end));
% fdata = fts2mat(db_factors.returns.fmpfts(updt(1):updt(2)));
fdata = fts2mat(db_factors.returns.fmpfts(updt(1):end));
% choose weight function
wfunc = {'huber'};
% models defined by factor selection - see db_mdlindx.(mdl_type)
for k = 1:size(db_mdlindx.ol,1)
    fct_ind = cell2mat(db_mdlindx.ol{k,1});
    for j = 1:length(wfunc)
        % betas from initialisation
        B0 = squeeze(db_models.ol{j,k}.beta(end,:,:))';
        % find where no betas
        bnan = isnan(B0);
        % use median beta
        Bmed = repmat(db_models.ol{j,k}.b_median',1, size(B0,2)) ;
        % intial beta
        B0(bnan) = Bmed(bnan);
        % estimate model
        db_models.olu{j,k} = online_ts_reg(fdata(:,fct_ind), ydata, 'y', {'lambda',lambda}, {'weight', wfunc{j}}, {'B0',B0 }, {
    end
end

clear fct_ind ydata fdata k mdl_type wfunc bnan Bmed B0;

%% store beta in fints to keep track of dates
% set temp vars
% ret = db_procdata.returns.eret(updt(1): updt(2)) ;
ret = db_procdata.returns.eret(updt(1): end) ;
% choose model type
mdl_type = 'olu'; % rw - rolling window; ol - online
all_factors = {'bBIAS', 'bHML', 'bSMB', 'bMKT'};
% index over update period
% Uindex = index(updt(1): updt(2),:);

for k = 1:size(db_mdlindx.ol,1)
% for k = 2
    clear fct_ind;
    fct_ind = cell2mat(db_mdlindx.ol{k,1}); % choose factors
    fct_ind = [1 (fct_ind + 1)]; % include bias
    for i = 1:length(fct_ind) % for each factor
        clear fname;
        fname = all_factors{fct_ind(i)}; %fints title
        for j = 1:size(db_models.(mdl_type),1) % include in model index?
            db_models.(mdl_type){j,k}.beta(:,:,i) = db_models.(mdl_type){j,k}.beta(:,:,i) ; % set betas not in unverse to NaN
            beta = squeeze(db_models.(mdl_type){j,k}.beta(:,:,i)); % beta matrix for factor i
            db_models.(mdl_type){j,k}.fbeta{i} = fints(ret.dates, beta, fieldnames(ret,1), ret.freq, fname); %store as fints
            % mean beta of stocks in i
            db_models.(mdl_type){j,k}.b_mean(:,i) = nanmean(beta,2);
            % median beta
            db_models.(mdl_type){j,k}.b_median(:,i) = nanmedian(beta,2);
        end
    end
end
% clear vars
clear ret all_factors fct_ind fname  k i;

%% store for characteristics models

% online CAPM beta
db_procdata.beta_data{1} = db_models.olu{1}.fbeta{2};
db_procdata.beta_index{1} = 'online CAPM MKT beta';
% online FF3F HML beta
db_procdata.beta_data{2} = db_models.olu{2}.fbeta{2};
db_procdata.beta_index{2} = 'online FF3F HML beta';
% online FF3F SMB beta
```

```
db_procdata.beta_data{3} = db_models.olu{2}.fbeta{3};
db_procdata.beta_index{3} = 'online FF3F SMB beta';
% online FF3F MKT beta
db_procdata.beta_data{4} = db_models.olu{2}.fbeta{4};
db_procdata.beta_index{4} = 'online FF3F MKT beta';
% online FF3F MKT alpha
db_procdata.beta_data{5} = db_models.olu{2}.fbeta{1};
db_procdata.beta_index{5} = 'online FF3F alpha';

% save to disk
filepath = fullfile(userpathstr,'Algo_Invest/workspace/Betas');
olu_capm_beta = db_models.olu{1}.fbeta{2};
olu_ff_alpha = db_models.olu{2}.fbeta{1};
save(fullfile(filepath,strcat('olu_capm_beta_',uni)), 'olu_capm_beta');
save(fullfile(filepath, strcat('olu_ff_alpha_',uni)), 'olu_ff_alpha');

clear olu_capm_beta olu_ff_alpha;

%% Factor Models expected return forecast
% applied time t index here - check if time t+1 index is applied at
% portfolio construction stage


flambda = 0.99; % lambda for factor realisations
% Uindex = index(updt(1):updt(2),:);
Uindex = index(updt(1):end,:);
% ret = db_procdata.returns.ret(updt(1):updt(2));
ret = db_procdata.returns.ret(updt(1):end);
% rfr = db_procdata.returns.rfr(updt(1):updt(2));
rfr = db_procdata.returns.rfr(updt(1):end);
mdl_type = 'olu';

for i = 2 % FF3F
    clear sys_error;
    % factor selection
    fct_ind = cell2mat(db_mdlindx.ol{i,1});
    % factor premiums
%     fact_prem = fts2mat(db_factors.returns.fmpfts(updt(1):updt(2)+1));
    fact_prem = fts2mat(db_factors.returns.fmpfts(updt(1):end));
    fact_prem = fact_prem(:,fct_ind); % select factors
    [fcov,fact_prem_cov,fmu] = ewma_cov(flambda, fact_prem, {'mu0',  fmu0(end,:)}, {'COV_0', fcov0});
    % for each weight function (eachg filter type)
    for j = 1:size(db_models.ol,1)
        clear temp tempRet
        % stock factor beta
        beta = db_models.olu{j,i}.beta;
        % pre-allocation
        exp_ret = NaN*ones(size(beta,1)+1,size(beta,2));
        % for each time period
        for t = 1:size(beta,1)
            % time t index
            index_t = Uindex(t,:);
            % NaN to zero (needed to form index matrix)
            index_t(isnan(index_t)) = 0;
            % covariance index at t
            %{
            INDEX_t = index_t'*index_t;
            INDEX_t(INDEX_t==0) = NaN;
            index_t(index_t==0) = NaN;
            %}
            % time t beta - exclude intercept for factor expected return
            beta_t = reshape(squeeze(beta(t,:,2:end)), size(index_t,2),[]);
            % expected return for t+1
            exp_ret(t+1,:) = (beta_t * fmu(t,:)')' + repmat(fts2mat(rfr(t)),1,size(exp_ret,2));
            % systematic risk innovation at t+1
%             sys_error(t,:) = index_t.*(beta_t * (fact_prem(t,:) - fmu(t,:))')';
            % risk model for t+1
%             db_models.(mdl_type){j,i}.risk{t+1} = INDEX_t.*(beta_t(:,:)*fact_prem_cov{t}*beta_t(:,:)'); % exclude bias term
            db_models.(mdl_type){j,i}.risk{t+1} = (beta_t(:,:)*fact_prem_cov{t}*beta_t(:,:)'); % exclude bias term
        end
        % store as fints
        db_models.(mdl_type){j,i}.fexp = lagts(fints(ret.dates, exp_ret(2:end,:), fieldnames(ret,1), ret.freq),1,NaN) + repmat(
%         db_models.(mdl_type){j,i}.sys_error = fints(ret.dates, sys_error(1:end,:), fieldnames(ret,1), ret.freq);

        % remove latest risk model so that size / dates coincide with fexp
        db_models.(mdl_type){j,i}.risk(end) = [];
        % install first risk model from intialisation data

        % save risk model to disk
        SysRisk = db_models.(mdl_type){j,i}.risk;
        filepath = fullfile(userpathstr,'Algo_Invest/workspace/Factor_Risk');
        save(fullfile(filepath,strcat(mdl_type,num2str(i),'_','Risk_',uni)), 'SysRisk', '-v7.3');

        % save expected returns to disk
        SysRet = db_models.(mdl_type){j,i}.fexp;
        filepath = fullfile(userpathstr,'Algo_Invest/workspace/Expected_Returns');
        save(fullfile(filepath,strcat(mdl_type,num2str(i),'_','ER_',uni)), 'SysRet');

    end
end

clear t fct_ind fact_prem ret mdl_type beta beta_t index_t INDEX_t sys_error fcov fmu fact_prem_cov temp SysRet SysRisk exp_ret

%% Out-of-sample forecast accuracy
%
```

```
% specify models
all_mdls = {'ol', 'olu'};
% set forgetting factor
lambda = 0.98;

% for each model
for i = 1:length(all_mdls)
    % select model
    mdl_type = all_mdls{i};
    % specify models for each model type
    switch mdl_type
    case {'ol','olu'}
        mdl_indx = [2];
    case 'ch'
        mdl_indx = [6];
    case 'adch'
        mdl_indx = [2 5 6];
    end
    for j = 1:numel(mdl_indx)
%         expected returns - excess or absolute?
        clear fexp temp;
        fexp = db_models.(mdl_type){mdl_indx(j)}.fexp;
        % calculate Theil Stats
        if strcmp(mdl_type(end), 'u') % if update
            % systematic return innovations (FF3F)
%             sys_error = db_models.olu{2}.sys_error;
            % actual returns minus systematic return innovations
%             ret = db_procdata.returns.ret(updt(1):updt(2)) - sys_error;

%             ret = db_procdata.returns.ret(updt(1):updt(2));
            ret = db_procdata.returns.ret(updt(1):end);

            % set initial values
            mu0 = db_models.(mdl_type(1:2)){mdl_indx(j)}.Theil.MFE_T(end,:);
            COV_0 = db_models.(mdl_type(1:2)){mdl_indx(j)}.Theil.COV_FE;
            % clear initial Theil from memory
            db_models.(mdl_type(1:2)){mdl_indx(j)}.Theil = [];
            % calculate Theil Stats
%             db_models.(mdl_type){mdl_indx(j)}.Theil = fewma_theil2(lambda, fexp, ret, {'mu0', mu0 }, {'COV_0', COV_0} , {'cov
            % save to disk
            SysTheil = fewma_theil2(lambda, fexp, ret, {'mu0', mu0 }, {'COV_0', COV_0} , {'cov_type', 'updt'});
%             SysTheil = db_models.(mdl_type){mdl_indx(j)}.Theil;
            filepath = fullfile(userpathstr,'Algo_Invest/workspace/Theil_matrices');
            save(fullfile(filepath,strcat(mdl_type,num2str(mdl_indx(j)),'_Theil_', uni)), 'SysTheil', '-v7.3');
            clear SysTheil;
        else % if initial
            % systematic return innovations (FF3F)
%             sys_error = db_models.ol{2}.sys_error;
            % actual returns minus systematic return innovations
            ret = db_procdata.returns.ret(ini(1):ini(2));
            % calculate Theil stats
            db_models.(mdl_type){mdl_indx(j)}.Theil = fewma_theil2(lambda, fexp, ret, {'cov_type', 'init'});
            % save to disk
%             SysTheil = fewma_theil2(lambda, fexp, ret, {'cov_type', 'init'});
            SysTheil = db_models.(mdl_type){mdl_indx(j)}.Theil;
            filepath = fullfile(userpathstr,'Algo_Invest/workspace/Theil_matrices');
            save(fullfile(filepath,strcat(mdl_type,num2str(mdl_indx(j)),'_Theil_',uni)), 'SysTheil');
            clear SysTheil;

        end
    end
end

clear all_mdls lambda ret i mdl_type j fexp mu0 COV_0 AyaTheil;

%% characteristics models

% create characteristic data matrix
clear char_data char_data_ind;
for i = 1:size(db_procdata.char_data,2)
    ini_char_data = fts2mat(db_procdata.char_data{1,i});
    char_data(:,:,i) = ini_char_data(ini(1):ini(2),:);
    char_data_ind{i} = db_procdata.char_index{i};
end
clear ini_char_data;
% add CAPM beta to characteristic data

%{
% online CAPM beta
db_procdata.beta_data{1} = db_models.ol{1}.fbeta{2};
db_procdata.beta_index{1} = 'online CAPM MKT beta';
% online FF3F HML beta
db_procdata.beta_data{2} = db_models.ol{2}.fbeta{2};
db_procdata.beta_index{2} = 'online FF3F HML beta';
% online FF3F SMB beta
db_procdata.beta_data{3} = db_models.ol{2}.fbeta{3};
db_procdata.beta_index{3} = 'online FF3F SMB beta';
% online FF3F MKT beta
db_procdata.beta_data{4} = db_models.ol{2}.fbeta{4};
db_procdata.beta_index{4} = 'online FF3F MKT beta';
% online FF3F MKT alpha
db_procdata.beta_data{5} = db_models.ol{2}.fbeta{1};
db_procdata.beta_index{5} = 'online FF3F alpha';
%}
```

```
% char_data(:,:,size(db_procdata.char_data,2)+1) = db_models.(mdl_type){1}.beta(:,:,2);
filepath = fullfile(userpathstr,'Algo_Invest/workspace/Betas');
load(fullfile(filepath,strcat('ol_capm_beta_',uni)));
char_data(:,:,8) = fts2mat(ol_capm_beta);
char_data_ind{8} = 'CAPM \beta';
clear mdl_type i;

% add FF3F alpha to characteristic data
filepath = fullfile(userpathstr,'Algo_Invest/workspace/Betas');
load(fullfile(filepath,strcat('ol_ff_alpha_',uni)));
char_data(:,:,9) = fts2mat(ol_ff_alpha);
char_data_ind{9} = 'FF3F \alpha';

clear ol_capm_beta ol_ff_alpha;

%% characteristic models

db_mdlindx.ch{1,1} = {[4 2]};         % bvtp & mv
db_mdlindx.ch{2,1} = {[4 2 8]};       % bvtp & mv % capm beta
db_mdlindx.ch{3,1} = {[4 2 6]};       % bvtp & mv & moml
db_mdlindx.ch{4,1} = {[4 2 6 8]};     % bvtp & mv & moml & capm
db_mdlindx.ch{5,1} = {[4 2 6 7]};     % bvtp & mv & moml & moms
db_mdlindx.ch{6,1} = {[4 2 6 7 8]};   % bvtp & mv & capm beta & moml & moms
db_mdlindx.ch{7,1} = {[4 2 6 7 8 9]}; % bvtp & mv & capm beta & moml & moms & FF3F alpha
db_mdlindx.ch{8,1} = {[4 2 6 7 8 1]}; % bvtp & mv & capm beta & moml & moms & dy
db_mdlindx.ch{9,1} = {[4 2 6 7 8 9 1]}; % bvtp & mv & capm beta & moml & moms & FF3F alpha & dy

% variable descriptions
for i = 1:length(db_mdlindx.ch)
    db_mdlindx.ch{i,2} = char_data_ind(db_mdlindx.ch{i,1}{1});
end
clear i;

%% estimate characteristic models (initisalisation)
% we do not consider different lags

% db_models.ch = [];
ret = db_procdata.returns.eret(ini(1): ini(2));
ydata = fts2mat(db_procdata.returns.eret);
% ini data
ydata = ydata(ini(1):ini(2),:);
% choose model
% for k = 1:size(db_mdlindx.ch,1)
lag = 4;
% for k = 1:9
for k = 8
    ch_ind = cell2mat(db_mdlindx.ch{k,1});
    xdata = char_data(:,:,ch_ind);
    % choose stock universe
    for i = 1:length(ch_ind)
        xdata(:,:,i) = Iindex.*xdata(:,:,i);
    end
    % cross-sectional regressions
    db_models.ch{k} = cs_reg_001ab(xdata,Iindex.*ydata, 'robustfit', lag);
    % store payoffs in fints
    db_models.ch{k}.payoffs = fints(ret.dates, db_models.ch{k}.beta, {} ,ret.freq, 'Payoffs');
end
% clear temp vars
clear ydata ch_ind xdata k i;

%% Expected Return for Characteristic Models

rfr = db_procdata.returns.rfr(ini(1):ini(2));
ret = db_procdata.returns.eret(ini(1):ini(2));
lambda = 0.97;
% count number of observations
% sumnan = nansum(~isnan(fts2mat(Iindex.*ret)),1);
% for i = 1:length(db_models.ch)
for i = 8
    clear exp_ret
    exp_ret(:,:) = NaN*ones(size(ret,1)+1, size(ret,2));
    % smooth payoffs
    db_models.ch{i}.EWpayoffs = fewma_mean(lambda, db_models.ch{i}.payoffs);
    for t = lag+1:length(db_models.ch{i}.payoffs)
        clear char_mtx;
        % index at t
%         index_t = Uindex(t,:);
        % matrix of firm characteristics (predictors)
        char_mtx = db_models.ch{i}.X(:,:,t-lag+1);
        % expected return smoothed payoffs
        exp_ret(t+1,:) = (char_mtx * fts2mat(db_models.ch{i}.EWpayoffs(t))')';
    end
    % store in fints
    db_models.ch{i}.fexperet = lagts(fints(ret.dates, exp_ret(2:end,:), fieldnames(ret,1), ret.freq),1,NaN);
    % forecast (includes risk-free rate)
    db_models.ch{i}.fexpret = db_models.ch{i}.fexperet + repmat(fts2mat(rfr(t)),1,size(exp_ret,2));
    % forecast error
    fc_err_raw = fts2mat(ret - db_models.ch{i}.fexperet);
    % count number of forecasts for each company
    raw_sumnan = nansum(~isnan(fc_err_raw),1);
    % mean forecast error per model
    db_models.ch{i}.MFE_raw =  nansum((nanmean(fc_err_raw.*fc_err_raw,1).*raw_sumnan))/nansum(raw_sumnan);
end

clear fc_err_raw fc_err_smooth raw_sumnan smooth_sumnan ;
```

```
%% characteristic data for update phase

clear char_data char_data_ind;
for i = 1:size(db_procdata.char_data,2)
    updt_char_data = fts2mat(db_procdata.char_data{1,i});
%     char_data(:,:,i) = updt_char_data(updt(1):updt(2),:);
    char_data(:,:,i) = updt_char_data(updt(1):end,:);
    char_data_ind{i} = db_procdata.char_index{i};
end
clear updt_char_data;
% add CAPM beta to characteristic data
% mdl_type = 'ol';

%{
% online CAPM beta
db_procdata.beta_data{1} = db_models.olu{1}.fbeta{2};
db_procdata.beta_index{1} = 'online CAPM MKT beta';
% online FF3F HML beta
db_procdata.beta_data{2} = db_models.olu{2}.fbeta{2};
db_procdata.beta_index{2} = 'online FF3F HML beta';
% online FF3F SMB beta
db_procdata.beta_data{3} = db_models.olu{2}.fbeta{3};
db_procdata.beta_index{3} = 'online FF3F SMB beta';
% online FF3F MKT beta
db_procdata.beta_data{4} = db_models.olu{2}.fbeta{4};
db_procdata.beta_index{4} = 'online FF3F MKT beta';
% online FF3F MKT alpha
db_procdata.beta_data{5} = db_models.olu{2}.fbeta{1};
db_procdata.beta_index{5} = 'online FF3F alpha';
%}

% char_data(:,:,size(db_procdata.char_data,2)+1) = db_models.(mdl_type){1}.beta(:,:,2);
filepath = fullfile(userpathstr,'Algo_Invest/workspace/Betas');
load(fullfile(filepath,strcat('olu_capm_beta_',uni)));
char_data(:,:,8) = fts2mat(olu_capm_beta);
% char_data(:,:,8) = fts2mat(olu_capm_beta(1:updt(2)-updt(1)+1));
char_data_ind{8} = 'CAPM \beta';
clear mdl_type i;

% add FF3F alpha to characteristic data
filepath = fullfile(userpathstr,'Algo_Invest/workspace/Betas');
load(fullfile(filepath,strcat('olu_ff_alpha_',uni)));
char_data(:,:,9) = fts2mat(olu_ff_alpha);
% char_data(:,:,9) = fts2mat(olu_ff_alpha(1:updt(2)-updt(1)+1));
char_data_ind{9} = 'FF3F \alpha';

clear ol_capm_beta ol_ff_alpha;
%% estimate Characteristics models (updt)

% db_models.chu = [];
%{
Uindex = index(updt(1):updt(2));
ret = db_procdata.returns.eret(updt(1): updt(2));
ydata = fts2mat(db_procdata.returns.eret(updt(1):updt(2)));
%}

Uindex = index(updt(1):end,:);
ret = db_procdata.returns.eret(updt(1): end);
ydata = fts2mat(db_procdata.returns.eret(updt(1):end));


% choose model
lag = 4;
ind = 1;
% for k = 1:size(db_mdlindx.ch,1)
for k = 8
    ch_ind = cell2mat(db_mdlindx.ch{k,1});
    xdata = char_data(:,:,ch_ind);
    % choose stock universe
    for i = 1:length(ch_ind)
        xdata(:,:,i) = Uindex.*xdata(:,:,i);
    end
    % cross-sectional regressions
    db_models.chu{k} = cs_reg_001ab(xdata,Uindex.*ydata, 'robustfit', lag);
    % store payoffs in fints
    db_models.chu{k}.payoffs = fints(ret.dates, db_models.chu{k}.beta, {} ,ret.freq, 'Payoffs');
end

clear xdata ydata;

%% Expected Excess Returns Characteristics Models

% ret = db_procdata.returns.eret(updt(1):updt(2));
% rfr = db_procdata.returns.rfr(updt(1):updt(2));

lambda = 0.97;

ret = db_procdata.returns.eret(updt(1):end);
rfr = db_procdata.returns.rfr(updt(1):end);

% count number of observations
% sumnan = nansum(~isnan(fts2mat(Iindex.*ret)),1);
% for i = 1:length(db_models.chu)
```

```
% lagged characteristics by one month
lag = 4;

for i = 8

    clear exp_ret CharRet
    exp_ret(:,:) = NaN*ones(size(ret,1)+1, size(ret,2));
    % smooth payoffs
    db_models.chu{i}.EWpayoffs = fewma_mean(lambda, db_models.chu{i}.payoffs);
    for t = lag+1:length(db_models.chu{i}.payoffs)
        clear char_mtx;
        % index at t
%         index_t = Uindex(t,:);
        % matrix of firm characteristics (predictors)
        char_mtx = db_models.chu{i}.X(:,:,t-lag+1);
        % expected return smoothed payoffs
        exp_ret(t+1,:) = (char_mtx * fts2mat(db_models.chu{i}.EWpayoffs(t))')';
    end
    % store in fints
    db_models.chu{i}.fexperet = lagts(fints(ret.dates, exp_ret(2:end,:), fieldnames(ret,1), ret.freq),1,NaN);
    % forecast (includes risk-free rate)
    db_models.chu{i}.fexpret = db_models.chu{i}.fexperet + repmat(fts2mat(rfr(t)),1,size(exp_ret,2));

    % save expected return model
    CharRet = db_models.chu{i}.fexpret;
    filepath = fullfile(userpathstr,'Algo_Invest/workspace/Expected_Returns');
    save(fullfile(filepath,strcat('chu',num2str(i),'_ER_',uni)), 'CharRet');

    % forecast error
    fc_err_raw = fts2mat(ret - db_models.chu{i}.fexperet);
    % count number of forecasts for each company
    raw_sumnan = nansum(~isnan(fc_err_raw),1);
    % mean forecast error per model
    db_models.chu{i}.MFE_raw =  nansum((nanmean(fc_err_raw.*fc_err_raw,1).*raw_sumnan))/nansum(raw_sumnan);

end

clear index_t char_mtx exp_ret fc_err_raw fc_err_smooth raw_sumnan smooth_sumnan;

%% Out-of-sample forecast accuracy
%
% specify models
all_mdls = {'ch', 'chu'};
% set forgetting factor
lambda = 0.97;

% for each model
for i = 1:length(all_mdls)
    % select model
    mdl_type = all_mdls{i};
    % specify models for each model type
    switch mdl_type
    case {'ol','olu'}
        mdl_indx = [2];
    case {'ch','chu'}
        mdl_indx = [8];
    case 'adch'
        mdl_indx = [2 5 6];
    end
    for j = 1:numel(mdl_indx)
%         expected returns - excess or absolute?
        clear fexp CharTheil;
        fexp = db_models.(mdl_type){mdl_indx(j)}.fexpret;
        % calculate Theil Stats
        if strcmp(mdl_type(end), 'u') % if update
            % systematic return innovations (FF3F)
%             sys_error = db_models.olu{2}.sys_error;
            % actual returns minus systematic return innovations
%             ret = db_procdata.returns.ret(updt(1):updt(2)) - sys_error;
%             ret = db_procdata.returns.ret(updt(1):updt(2));
            ret = db_procdata.returns.ret(updt(1):end);
            % set initial values
            filepath = fullfile(userpathstr,'Algo_Invest/workspace/Theil_matrices');
            load(fullfile(filepath,strcat(mdl_type(1:2),num2str(mdl_indx(j)),'_Theil_', uni )));
            mu0 = CharTheil.MFE_T(end,:);
            COV_0 = CharTheil.COV_FE;
            clear CharTheil
%             mu0 = db_models.(mdl_type(1:2)){mdl_indx(j)}.Theil.MFE_T(end,:);
%             COV_0 = db_models.(mdl_type(1:2)){mdl_indx(j)}.Theil.COV_FE;
            % calculate Theil Stats
%             db_models.(mdl_type){mdl_indx(j)}.Theil = fewma_theil2(lambda, fexp, ret, {'mu0', mu0 }, {'COV_0', COV_0} , {'cov
            % save to disk
%             CharTheil = db_models.(mdl_type){mdl_indx(j)}.Theil;
            CharTheil = fewma_theil2(lambda, fexp, ret, {'mu0', mu0 }, {'COV_0', COV_0} , {'cov_type', 'updt'});

            save(fullfile(filepath,strcat(mdl_type,num2str(mdl_indx(j)),'_Theil_', uni)), 'CharTheil', '-v7.3');

        else % if initial
            % systematic return innovations (FF3F)
%             sys_error = db_models.ol{2}.sys_error;
            % actual returns minus systematic return innovations
%             ret = db_procdata.returns.ret(ini(1):ini(2)) - sys_error;
            ret = db_procdata.returns.ret(ini(1):ini(2));
            % calculate Theil stats
%             db_models.(mdl_type){mdl_indx(j)}.Theil = fewma_theil2(lambda, fexp, ret, {'cov_type', 'init'});
```

```
            % save to disk
%             CharTheil = db_models.(mdl_type){mdl_indx(j)}.Theil;
            CharTheil = fewma_theil2(lambda, fexp, ret, {'cov_type', 'init'});
            filepath = fullfile(userpathstr,'Algo_Invest/workspace/Theil_matrices');
            save(fullfile(filepath,strcat(mdl_type,num2str(mdl_indx(j)),'_Theil_', uni)), 'CharTheil', '-v7.3');

        end
    end
end
clear all_mdls lambda ret i mdl_type j fexp CharTheil sys_error;



%% GMV Portfolio Construction

% absolute returns
% ret = db_procdata.returns.ret(updt(1):updt(2)).*Uindex;

%{
% ret = db_procdata.returns.ret(updt(1):end).*Uindex;
% standard GMV (double index)

clear GMV;
GMV = cell(1,11);
for i = 0:0.1:1
    GMV{round(10*i+1)} = GMV_panel001aa(ret, db_models.olu{2}.risk, i );
end

filepath = fullfile(userpathstr,'Algo_Invest/workspace/portfolios');
cd(filepath);
save(strcat('GMV_',uni)'GMV');
save(strcat('STRAT_',uni), 'STRAT');
%}


%% STRAT Portfolio Construction

clearvars -except index db_procdata updt userpathstr uni;
% in-sample
%{
Uindex = index(updt(1):updt(2),:);
ret = db_procdata.returns.ret(updt(1):updt(2)).*Uindex;
rfr = db_procdata.returns.rfr(updt(1):updt(2));
%}

% out-sample
Uindex = index(updt(1):end,:);
ret = db_procdata.returns.ret(updt(1):end).*Uindex;
rfr = db_procdata.returns.rfr(updt(1):end);

% risk
filepath = fullfile(userpathstr,'Algo_Invest/workspace/Factor_Risk');
load(fullfile(filepath,strcat('olu2_Risk_',uni))) ;
% SysRisk = temp(:) ;
% clear temp;

% return
filepath = fullfile(userpathstr,'Algo_Invest/workspace/Expected_Returns');
load(fullfile(filepath,strcat('olu2_ER_',uni))) ;
% SysRet = tempRet ;
% clear temp;

% uncertainty
filepath = fullfile(userpathstr,'Algo_Invest/workspace/Theil_matrices');
load(fullfile(filepath,strcat('olu2_Theil_',uni))) ;
% SysTheil = temp(:) ;
% clear temp;

%{
% in-sample portfolio
% STRAT = cell(11,15);
STRAT = cell(1,11);
% including tuning parameter
for k = 0:0.1:1 % shrinkage intensity
    j = 1; % tuning parameter
%     for j = 0.1:0.1:1.5 % tuning parameter
        STRAT{round(10*k+1)} = STRAT_panel001ab(ret, rfr, SysRisk, SysTheil.COV_FE, fts2mat(SysRet), k, j);
%     end
end

filepath = fullfile(userpathstr,'Algo_Invest/workspace/portfolios');
cd(filepath);
save(strcat('STRAT_',uni), 'STRAT');
%}

% out-of-sample portfolio
k = 3;
j = 1;
STRAT_final = STRAT_panel001ab(ret, rfr, SysRisk, SysTheil.COV_FE, fts2mat(SysRet), k, j);
filepath = fullfile(userpathstr,'Algo_Invest/workspace/portfolios');
cd(filepath);
save(strcat('STRAT_final_',uni), 'STRAT_final');

%% TACT
```

```
% absolute returns
%{
Uindex = index(updt(1):updt(2),:);
ret = db_procdata.returns.ret(updt(1):updt(2)).*Uindex;
rfr = db_procdata.returns.rfr(updt(1):updt(2));
%}

Uindex = index(updt(1):end,:);
ret = db_procdata.returns.ret(updt(1):end).*Uindex;
rfr = db_procdata.returns.rfr(updt(1):end);

clear db_procdata;

% Factor risk
filepath = fullfile(userpathstr,'Algo_Invest/workspace/Factor_Risk');
load(fullfile(filepath,strcat('olu2_Risk_',uni))) ;
% SysRisk = temp(:) ;
% clear temp;

% Systematic Uncertainty
filepath = fullfile(userpathstr,'Algo_Invest/workspace/Theil_matrices');
load(fullfile(filepath,strcat('olu2_Theil_',uni))) ;
% SysTheil = temp;
% clear temp;

% Systematic return
filepath = fullfile(userpathstr,'Algo_Invest/workspace/Expected_Returns');
load(fullfile(filepath,strcat('olu2_ER_',uni))) ;
% SysRet = tempRet ;
% clear tempRet;

% Characteristic Uncertainty
for i = 8
    filepath = fullfile(userpathstr,'Algo_Invest/workspace/Theil_matrices');
    load(fullfile(filepath,strcat('chu', num2str(i) ,'_Theil_',uni))) ;

    % Characteristic return
    filepath = fullfile(userpathstr,'Algo_Invest/workspace/Expected_Returns');
    load(fullfile(filepath,strcat('chu', num2str(i) ,'_ER_',uni))) ;

    % shrinkage intensity
%     TACT = cell(1,11);
    for k = 2
%         TACT{round(10*k+1)}  = TACT_panel001aa(ret, rfr,  SysRisk, SysTheil.COV_FE, CharTheil.COV_FE, fts2mat(SysRet), fts2ma
        TACT  = TACT_panel001aa(ret, rfr,  SysRisk, SysTheil.COV_FE, CharTheil.COV_FE, fts2mat(SysRet), fts2mat(CharRet), k, 1
    end

    filepath = fullfile(userpathstr,'Algo_Invest/workspace/portfolios');
    cd(filepath)
%     save(strcat('TACT_',num2str(i),'_',uni), 'TACT');
    save(strcat('TACT_',num2str(i),'_final_',uni), 'TACT');
end
```

B.2 RLM Filter

```
function [s] = online_ts_reg_new(xdata, ydata, const, varargin)
% Curent Situation:
% This function performs a recursive regression for each output signal in
% ydata onto the input signals xdata.
% The parameters are caluclated by recursively calculating the normal
% equation:
% B1 = inv(X1'X1)(X1'y1)
% B1 = {inv(X0'X0) + C(inv(x1'x1))}{(X0'y0 + lambda*weight*x1'y1)}
% Usually B1 is calculated as:
% B1 = B0 + gain*error
% This has the advantage of controlling the gain directly
%
% The weight function decreases the weights of new observations that deviate
% from the current estimate. ie the weight is a function of the apriori
% error
%
% TBDL:
%   NB: recursively calculate deviation from median - DONE
%   change calculation method to B1 = B0 + delta if necessary.
%   include bisquare weight function
%   online formulation - ie include arguments for previous estimates
%
% INPUTS
% --------------
% xdata     (req)   -   2-dimensional array unfiltered input
% ydata     (req)   -   2-d array of output signals
% const     (req)   -   {'y','n'}. include regression constant?
% lambda    (opt)   -   scalar. memory factor
% arg_weght (opt)   -   {'huber','std'} robust weight functions
% arg_invR0 (opt)   -   initial inv(X'X)
% arg_p0    (opt)   -   initial (X'y)
%
```

```
%% default parms for optional aguments
lambda          = 0.98;
arg_weight      = 'huber';
c               = 999999;
if const       == 'y'
    xdata       = [ones(size(xdata,1),1) xdata(:,:)];
end
arg_invR0       = c*eye(size(xdata,2));
arg_p0          = zeros(size(xdata,2), size(ydata,2));

%update default parms with user inputs
nvarargin = numel(varargin);
if nvarargin > 0
    for i=1:nvarargin
        switch varargin{i}{1,1}
            case 'lambda'
                lambda  = varargin{i}{1,2};
            case 'p0'
                arg_p0  = varargin{i}{1,2};
            case 'invR0'
                arg_invR0   = varargin{i}{1,2};
            case 'B0'
                arg_B0  = varargin{i}{1,2};
                arg_p0  = arg_invR0 \ argB0;
            case 'weight'
                arg_weight  = varargin{i}{1,2};
        end
    end
end

%% Estimate median average deviation estimator of Residual standard deviation
% whole sample
% MdAD = mad(ydata,1);
% moving median
wl = round(1/(1-lambda));
MdAD = movmed(ydata,[wl 0],'omitnan');
% recursive update for determining median


%% weight functions
switch arg_weight
    case 'std' %no robust adjustment
        w = @(e1,MAD)1;
    case 'huber'
        k = 1.385; % tuning constant
        w = @(e1,MAD)(min(1,k/(abs(e1/MAD))));
end

%% adjust dataset for NaNs -
%set 0s to NaN
xdata(xdata==0) = NaN;
ydata(ydata==0) = NaN;
%set NaN rows to zero in body of code => beta stays constant.



%% recursive regression
% run regression for each company
for i = 1:size(ydata,2)
            y = ydata(:,i);
            x = xdata;
            % identify rows with
            nidx = (isnan(y) | transpose(sum(transpose(isnan(x)))>0));
            % set NaN to zero
            y(nidx,:) = 0;
            x(nidx,:) = 0;
                        % initial invR0
                invR0 = arg_invR0;
                % initial p0
                p0 = arg_p0(:,i);
                % initial B0
                B0 = invR0 * p0;
            for t = 1:size(ydata,1) %every month



                % new response
                y1 = y(t);
                % new input
                x1 = x(t,:);
                % calculate a priori error
                e1 = y1 - x1*B0;
                % calculate robust weight
                weight(t,i) = w(e1,MdAD(t,i));
                % update inverse of autocorrelation matrix
                invR1 = (1/lambda)*(invR0 - invR0*x1'*inv(x1*invR0*x1'+lambda*inv(weight(t,i)))*x1*invR0);
                % update autocorrelation vector
                p1 = lambda*p0 + weight(t,i)*x1'*y1;
                % update beta
                B1 = invR1*p1;
                % stats
                s.beta(t,i,:)       = B1;
                s.y_hat(t,i)        = x1*B1;
```

```
                s.res(t,i)          = y(t) - s.y_hat(t,i);
                s.weight(t,i)       = weight(t,i);
                s.B0(t,i,:)         = B0;
                s.fc_err(t,i)       = e1;
                s.p0{t,i}           = p0;
                s.p0{t,i}           = p1;
%                 s.invR0{j,i}        = invR0;
%                 s.invR1{j,i}        = invR1;

                % testing stat
%                 s.huber{j,i}        = k/(abs(e1/MdAD(j,i)));
                % update for next iteration
                invR0 = invR1;
                p0 = p1;
                B0 = B1;
            end
            s.res(nidx,i) = NaN;
            s.fc_err(nidx,i) = NaN;
end
sumnan = sum(~isnan(s.res));
sqRES = nanmean(s.res.*s.res,1);
s.MSE = nansum(sqRES.*sumnan)/sum(sumnan);
s.MFE = nansum(nanmean(s.fc_err.*s.fc_err,1).*sumnan)/sum(sumnan);

%{
MAE = abs(s.res);       % mean absolute (estimation) error
MAFE = abs(s.fc_err);   % mean absolute (forecast) error

MAE_NF = nanmean(abs(fts2mat(diff(fret))))  ;   % mean absolute error of naive forecast
%}



%% temp fix
s.beta(s.beta == 0) = NaN;
s.varargin          = varargin;
```

B.3 Cross-Sectional Regressions

```
function [multi] = cs_reg_001ab(xdata, ydata, reg_type, lag)
%
% version: 001ab - include lag for x data
%
% Cross-Sectional Regressions - Estimate characteristic payoffs at each time period
%
% Author: T Gebbie, A B Paskaramoorthy
%
% Current Situation: Test code needs to be written. But code produces same
% payoffs for the [bias butp mu] model from the EPC_HML_SMB_S203_FMP_250 script.
%
% Future Situation: have to write the test code
%
% Function Details:
% -----------------
% {multi} = cs_reg(x_data, y_data, reg_type)
%
% OUTPUTS
% -------
% s         - struct array containing model inputs, output, paramter
%             estimates and statistics
%
% INPUTS
% ------
% x_data    - 3 dimensional array of company characteristics
%             (time x company x characteristic)
% y_data    - 2 dimensional array of (excess) company returns
% reg-type  - robustfit, ridge

% default reg_type

%pre-allocate multi
multi.beta       = NaN*ones(size(xdata,1), size(xdata,3)+1); % +1 for constant
multi.se         = NaN*ones(size(xdata,1), size(xdata,3)+1);
multi.t          = NaN*ones(size(xdata,1), size(xdata,3)+1);
multi.adjr2      = NaN*ones(size(xdata,1), 1);
multi.pval       = NaN*ones(size(xdata,1), size(xdata,3)+1);


for t = lag+1:size(ydata,1), % for each time period t
        clear x y;
        x = [squeeze(xdata(t - lag,:,:))]; % x = char data at time i
        x(abs(x)==Inf)=NaN;
        x1 = x;
        y = ydata(t,:)';
        y(abs(y)==Inf)=NaN;
        % set beta and stats to NaN
        beta = NaN*ones(size(x,2),1);
        % find NaNs
```

```
        nidx = ~(isnan(y) | transpose(sum(transpose(isnan(x)))>0));
        % remove data without all characteristics (selection effect)
        y=y(nidx);
        x=x(nidx,:);
        % winsorize the data
        if (size(x,1) > size(x,2)+2),  %if 2 more rows than columns - arbitrary
            % winsorize the data
            x = winsorize(x);
            switch reg_type
                case 'robustfit'
                    % carry out regressions
                    [beta, stats] = robustfit(x,y);
                case  'ridge'
                    % z-score data
                    x = nanzscore(x);
                    [beta] = ridge(y,x,k,0); % no centering and scaling
                    % generate the regression statistics
                    stats = regstats(y,x);
                    stats.t = stats.tstat.t;
                    stats.p = stats.fstat.pval;
                    stats.se = sqrt(diag(stats.covb));
                otherwise
            end


        % raw model statisics
%           s.X{i,:}    = [ones(size(x,1),1) x];
%           s.t{i,:}    = stats.t;
%           s.p{i,:}    = stats.p;
%           s.b{i,:}    = beta;
%           s.se{i,:}   = stats.se;

%             s.X         = [ones(size(x1,1),1) nanzscore(x1)];
            s.X         = [ones(size(x1,1),1) x1];
            s.t         = stats.t;
            s.p         = stats.p;
            s.b         = beta;
            s.se        = stats.se;
            s.yhat      = s.X * s.b;
            s.y         = ydata(t,:)';
            s.y2         = y;
            s.yhat2      = [ones(size(x,1),1) x1(nidx,:)] * beta;
            s.Mres      = s.y2 - s.yhat2; % fitted model residual (in-sample)
            s.resid     = (s.X * s.b) - ydata(t,:)'; %residual including points not in training set
            s.ybar      = nanmean(s.y2);
            s.SSE       = nansum(s.Mres.*s.Mres);
            s.SSR       = nansum((s.yhat2 - s.ybar).*(s.yhat2 - s.ybar));
            s.SST       = nansum((s.y2 - s.ybar).*(s.y2 - s.ybar));
            s.dfe       = length(y) - length(s.b);
            s.dfr       = length(s.b) - 1;
            s.dft       = length(y) - 1;
            s.F         = (s.SSR / s.dfr) * (s.SSE / s.dfe);
            s.Fpval      = 1 - fcdf(s.F,s.dfr,s.dfe);
            s.R2         = 1 - s.SSE / s.SST;
            s.adjR2      = 1 - s.SSE / s.SST * (s.dft /s.dfe);

            % output statistics etc.
            multi.X(:,:,t)        = s.X;
            multi.beta(t,:)       = s.b;
            multi.se(t,:)         = s.se;
            multi.t(t,:)          = s.t;
            multi.pval(t,:)       = s.p;
            multi.resid(t,:)      = s.resid;
            multi.F(t,:)          = s.F;
            multi.Fpval(t,:)      = s.Fpval;
            multi.R2(t,:)         = s.R2;
            multi.adjrR2(t,:)     = s.adjR2;
            multi.yhat(t,:)       = s.yhat;
%             multi.yhat2(t,:)      = s.yhat2;

        end;

end;
```

B.4 EW Mean

```
function [MU,mu_T] = ewma_mean(lambda, xdata, varargin)
% exponentially weighted recursive estimation of mean
% applies zero-order hold on mean for nans
%
% INPUTS
% -------
% lambda is forgetting factor
% xdata is matrix of sequential data with earliest data at top
% mu0 (vararg) is initial mean
%
% OUTPUTS
% --------
% mu_T is vector of last mean
```

```
% MU is matrix of EWMA
%



% find size of data matrix
[T,a] = size(xdata);
% make sure that initial means have same number of columns to xdata
if numel(varargin) > 0,
    % initial means
    mu0 = varargin{1};
else
    % set to NaN if mean not specified
    mu0 = NaN*ones(1,a);
end;

%% EWMA calc
% pre-allocate MU
MU = NaN*ones(T+1,a);
% initialise ewma mean
MU(1,:) = mu0;
% at each period
for i = 1:T,

    % initialize at each point
%     curr_nan = zeros(1,size(MU,2));
    % find nans in current
    curr_nan = isnan(MU(i,:));
    % clean index
%     inc_nan = zeros(1,size(xdata,2))
    % find nans
    inc_nan = isnan(xdata(i,:));

    % case 1: nan prior mean, nan incoming data => NaN post mean
    case1_nan = curr_nan & inc_nan;
    MU(i+1,case1_nan) =  NaN;
    % case 2: nan prior mean, incoming data => post mean = inc data
    case2_nan = (curr_nan==1) & (inc_nan==0);
    MU(i+1,case2_nan) = xdata(i,case2_nan);
    % case 3: prior mean but nan in incoming data
    case3_nan = (curr_nan==0) & (inc_nan==1);
    MU(i+1,case3_nan) = MU(i,case3_nan);
    % case 4: prior mean and incoming data
    case4_nan = (curr_nan==0) & (inc_nan==0);
    MU(i+1,case4_nan) = (1-lambda)*xdata(i,case4_nan) + lambda*MU(i,case4_nan);


end;
% drop initial means
MU = MU(2:end,:);
% most recent mean
mu_T = MU(T,:);
```

B.5 EW Covariance

```
function [COV_T, ewCOV, MU] = ewma_cov(lambda, xdata, varargin)
%
% version:  001ae   - include parameter to seperate data into
% initialization period and test period

%
% ewmacov calculates the exponentially weighted covariance matrix for panel
% data. ewmacov applies zero-order-hold on covariances if data is nan (for now).
% lambda = 1 should recover the rolling window covariance matrix
% what mean should be used? here we use the exponentially weighted moving
% average.
%
% NB: if the length of each time series is different in the data matrix,
% then it is possible that the resulting covariance matrix will not be
% positive definite. the asset with data will have a variance estimate,
% whilst the other asset(s) will not. when the first pairwise deviation is
% calculated, if the prior estimate of the asset's variance is less than
% current squared deviation the posterior variance will be smaller than the
% prior. However, the other elements of the posterior covariance will be
% equal to the current deviation matrix. The resulting matrix is not
% positive definite. (for illustration take 2x2 case)
%
% prior Cov             current Dev                post Cov
%   N | N               0.09 | (0.3)(0.2)          0.09 | (0.3)(0.2)
%   ------              ----------------           ----------------
%   N | 0.03       (0.3)(0.2)| 0.04           (0.3)(0.2)| a0.03 + (1-a)0.04
%
% The deviation matrix is singular (rank 1), whilst the post Cov has
% negative determinant
%
% INPUTS
% ------
% lambda        - forgetting factor [0,1].
% xdata         - panel data, multiple time series, top row is 1st obs,
```

```
%                   bottom row is latest
% COV_0 (opt)    - initial covariance matrix
% mu0 (opt)      - initial mean
%
% variable arguments are entered as cell arrays with 2 elements where the
% first element is the name of the parameter eg {'mu0',xxx}
%
% OUTPUTS
% -------
% COV_T          - latest covariance matrix (time T)
% COV            - 3d matrix. covariance matrices over time
% MU             - matrix of exponentially weighted means
%
% TBDL
%   1. ideally, want robust estimate of mean and covariance
%
%% default initialization
mu_0 = NaN*ones(1,size(xdata,2));
COV_0 = NaN*ones(size(xdata,2),size(xdata,2));
p = 0;

%% load optional variables (initialize variables)
if numel(varargin) > 0
    for i = 1:numel(varargin),
        switch varargin{i}{1}
            case 'mu0'
                mu_0 = varargin{i}{2};
            case 'COV_0'
                COV_0 = varargin{i}{2};
            case 'trdata'
                p = round(varargin{i}{2}*size(xdata,1));
        end; % switch
    end; % for
end; % if

% pre-allocate memory
[T,a] = size(xdata);
ewCOV = cell(T+1,1);
pw_DEV = cell(T+1,1);
dev = NaN*ones(T+1,a);
MU = NaN*ones(T+1,a);

%% calculate covariances
if p == 0

    % calculate means
    [MU,~] = ewma_mean(lambda, xdata, mu_0);
    % initialize ewma covariance
    ewCOV{1} = COV_0;
    % initialize ewma means
    MU = [mu_0; MU];
    xdata = [NaN*ones(1,a); xdata];
    % calculate ew cov matrix
    for t = 1:T,
            % find nans in current covariance
            curr_nan = isnan(ewCOV{t});
            % find nans in incoming deviations
            % calculate deviations
            dev(t+1,:) = xdata(t+1,:)-MU(t+1,:);
            % calculate pairwise deviations
            pw_DEV{t+1}(:,:) = dev(t+1,:)'*dev(t+1,:);
            % find nans in incoming data
            inc_nan = isnan(pw_DEV{t+1}(:,:));

            % case 1: nan prior cov, nan incoming data => nan post Cov
            case1_nan = curr_nan & inc_nan;
            ewCOV{t+1}(case1_nan) = NaN;
            % case 2: nan prior cov, incoming data => post cov = NaN
            case2_nan = (curr_nan==1) & (inc_nan==0);
                % case 2.1: case2_nan = 1, pw_DEV = 0 (first observation has 0 mean)
                case21_nan = (case2_nan==1) & (pw_DEV{t+1}) == 0;
                ewCOV{t+1}(case21_nan) = NaN;   % need 2 observations to calculate dev
                % case 2.2: case2_nan = 1, pw_DEV ~= 0 (2nd observation)
                case21_nan = (case2_nan==1) & (pw_DEV{t+1}) ~= 0;
                ewCOV{t+1}(case21_nan) = pw_DEV{t+1}(case21_nan);   % need 2 observations to calculate dev
            % case 3: prior cov, nan incoming data => post cov = prior cov
            case3_nan = (curr_nan==0) & (inc_nan==1);
            ewCOV{t+1}(case3_nan) = ewCOV{t}(case3_nan);
            % case 4: prior mean and incoming data
            case4_nan = (curr_nan==0) & (inc_nan==0);
            ewCOV{t+1}(case4_nan) = (1-lambda)*pw_DEV{t+1}(case4_nan) + lambda*ewCOV{t}(case4_nan);
            % reshape ewCOV{t+1} from vector to matrix
            ewCOV{t+1} = reshape(ewCOV{t+1},size(xdata,2),size(xdata,2));
    end;

    %% last covariance matrix
    % COV_T = squeeze(ewCOV(T,:,:));
    COV_T = ewCOV{T};

% if want to initialize using data
elseif p > 0
    % initialize mean and covariance using p datapoints
    trCOV = nancov(xdata(1:p,:));
    [~,trMu] = ewma_mean(lambda, xdata(1:p,:));
    % calculate
```

```
%     [COV_T, ewCOV, MU] = ewma_cov(lambda, xdata(p+1:end,:), {'mu0',trMu}, {'COV_0',trCOV});
%   p+2 since initial values are first element
    [COV_T, ewCOV(p+2:end), MU(p+2:end,:)] = ewma_cov(lambda, xdata(p+1:end,:), {'mu0',trMu}, {'COV_0',trCOV});
end

% remove initial points
ewCOV(1) = [];
pw_DEV(1) = [];
MU = MU(2:end,:);
```

B.6 EW Theil Statistics

```
function [s] = fewma_theil(lambda, fy_hat, fy)
%
%
% function recursively calculates the first and second moments of the
% simulated series, y_hat, and the actual series, y, to produce the Theil
% related stats.
% Theil stats are time series statistics. Here, they are recursively
% calculated using exponential weightings.
% inputs are FINTS.
%
% INPUTS
% fy_hat (fints matrix)     - estimated series / forecasts
% fy (fints matrix)         - actual return series
% ICOV (matrix)             - initial matrix for covariance of forecast
%                                 errors
%
% OUTPUTS
% s.U
% s.Um (fints)  -
% s.Us (fints)  -
% s.Uc (fints)  -
%
% TBDL
% 1. used Tim's ewmacov
%% convert to matrix
y_hat   = fts2mat(fy_hat);
y       = fts2mat(fy);

%% nan-handling
nan_ind = isnan(y_hat) | isnan(y);
y(nan_ind) = NaN;
y_hat(nan_ind) = NaN;


%% additional series
% square simulated series
y_hat_sq = y_hat.^2;
% square actual series
y_sq     = y.^2;
% dot product series
y_y_hat  = y.*y_hat;
% forecast error series
FE = y_hat - y;
% squared forecast error
FE_sq = FE.^2;

%% calculate moments
% [x, y] = ewma_mean(a,b)   x ~ latest mean, y ~ matrix/vector of means
% EWMA simulated series
[MU_hat, mu_hat_T] = ewma_mean(lambda, y_hat);
% EWMA actual series
[MU, mu_T] = ewma_mean(lambda, y);
% EWMA square simulated series
[MU_hat_sq, mu_hat_sq_T] = ewma_mean(lambda, y_hat_sq);
% EWMA square actual series
[MU_sq, mu_sq_T] = ewma_mean(lambda, y_sq);
% EWMA cross terms
[MU_sa, mu_sa_T] = ewma_mean(lambda, y_y_hat);
% EWMA forecast error - should equal (MU_hat-MU);
[MU_FE, mu_FE_T] = ewma_mean(lambda, FE);
% EWMA squared forecast error
[MU_FE_sq, mu_FE_sq_T] = ewma_mean(lambda, FE_sq);
% EW covariance of forecast errors
[~,COV_FE,~] = ewma_cov(lambda, FE);

%% calc forecast error statistics
% Theil U denomiator
U_den = MU_sq.^(1/2) + MU_hat_sq.^(1/2);
% mean simulation error
MSimE = MU_hat - MU;
% Theil U - normalized mean square simulation error
U = (MU_FE_sq.^(1/2))./U_den;
% Theil Um - difference of means
Um = ((MU_hat - MU).^2)./MU_FE_sq;
% Theil Us - difference of standard deviations
Sig_s = (MU_hat_sq - MU_hat.^2).^(1/2);
Sig_a = (MU_sq - MU.^2).^(1/2);
```

```
Us = ((Sig_s - Sig_a).^2)./MU_FE_sq;
% Theil Uc - deviations from poor correlation between forecast and actual
Sig_sa = (MU_sa - MU_hat.*MU);
Uc = 2*(Sig_s.*Sig_a - Sig_sa)./MU_FE_sq;

%% outputs
% Theil stats
s.U     = fints(fy.dates,U,fieldnames(fy,1),fy.freq, 'Theil U');
s.U2    = Um+Us+Uc;
s.Um    = fints(fy.dates,Um,fieldnames(fy,1),fy.freq, 'Theil Um (means)');
s.Us    = fints(fy.dates,Us,fieldnames(fy,1),fy.freq, 'Theil Us (stddev)');
s.Uc    = fints(fy.dates,Uc,fieldnames(fy,1),fy.freq, 'Theil Uc (Cov)');

% General out-of-sample statistics
s.FE        = FE;
s.FE_sq     = FE_sq;
s.MFE       = MU_FE;
s.MSFE      = MU_FE_sq;
s.COV_FE    = COV_FE;
```

B.7 Invert Diagonal Matrix with NaNs

```
function [invA, I, cn_o, cn_cl ] = naninv_001ad(A, varargin)
%
% version: 001ad - change nan handling to remove rows and column with nan.
%                  naninv inverses submatrix without nans.
%
% removes nans from matrix and takes inverse, and produces matrix with NaNs
% this is only used for matrix which includes stocks THAT ARE NOT
% INVESTABLE. ie stocks which are not part of the market
% In this case, the NaN does not represent missing data.
%
% if the NaN in the came from illiquid stocks, it would be best to impute
% the missing data rather than remove.
%
% Note the following required properties of the input matrix:
%   1. symmetric
%   2. if A[ij] = NaN, then either A[i.] = NaN or A[.j]=NaN

%
% inverse type default
type = 'inv';
% optional inputs
if numel(varargin) > 0
    type = varargin{1};
end
% find size of matrix
[rn,cn] = size(A);
% find nans in the matrix
% nanind = (sum(~isnan(A))>0);
% nanind = ~(sum(isnan(A))>0);
nanind = ~isnan(diag(A));
% remove nans from matrix - will be whole columns and rows (see 2. above)
A1 = A(nanind',nanind);

% fill invA with NaNs
invA = NaN*ones(rn,cn);
% if matrix is not empty
if ~isempty(A1)
    % condition numbers original
    cn_o = cond(A1);
    % clean matrix
    switch type
        case 'inv'
            tmp_invA = A1^(-1);
            cn_cl = NaN;
        case 'cl'
            [A1, ~, ~] = cleancov(A1);
            % condition number cleaned
            cn_cl = cond(A1);
            % take inverse
            tmp_invA = A1^(-1);
        case 'pinv'
            tmp_invA = pinv(A1);
            cn_cl = NaN;
    end
    % fill invA elements
    invA(nanind',nanind)=tmp_invA;
    % check
    I = tmp_invA*A1;
else
    I = NaN*ones(rn,cn);
    cn_o = NaN;
    cn_cl = NaN;
end % if
end % function
```