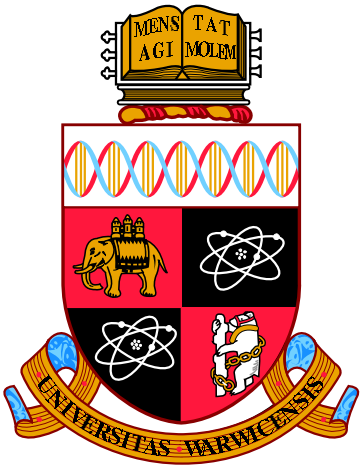


The Thermodynamics of Risk

by

Jonathan Hugo Mascie-Taylor

Thesis

Submitted to the University of Warwick

for the degree of

Doctor of Philosophy

Complexity Science

October 2014

THE UNIVERSITY OF
WARWICK

Contents

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Robin Ball, for his constant guidance and support. He has always found time for many useful discussions and supervisions over the last few years. He has, without exception, been willing to share his considerable knowledge and delve deeper into any problem presented.

I would also like to thank my second supervisor, Juergen Branke for many insightful and interesting discussions, both one and one, and through his organisation of operations research PhD seminars.

Further thanks goes to my friends and colleagues in the Complexity DTC for making the work enjoyable, providing many fruitful discussions and enjoyable games of squash.

Finally, I would like to thank my family and friends for their patience, endurance and continual support. Mary in particular has put up with (and fed) me during my many long nights studying mathematics and working with MATLAB code.

This work was funded by the Engineering and Physical Sciences Research Council as part of the University of Warwick Complexity Science Doctoral Training Centre.

This thesis was typeset with LaTeX 2$_\varepsilon$[1] by the author.

[1] LaTeX 2$_\varepsilon$ is an extension of LaTeX. LaTeX is a collection of macros for TeX. TeX is a trademark of the American Mathematical Society. The style package *warwickthesis* was used.

Dedication

To God, who has granted me wisdom, guidance and purpose.

> *Now all glory to God, who is able, through his mighty power at work within us, to accomplish infinitely more than we might ask or think.*
>
> (The Bible, Ephesians 3:20, New Living Translation)

Declarations

This work has been composed by myself and has not been submitted for any other degree or professional qualification.

- The first half of chapter 4 is in the process of being submitted for publication.
- Chapters 3 and 5 will be submitted for publication.
- Chapters 6 and 7 will be submitted for publication alongside some of the further work discussed in chapter 8.
- The introduction on Reverse Stress Testing (1.3) may be extended to form a formal review on Reverse Stress Testing.

Abstract

It is now routine to consider the full probability distribution of downturns in many sectors. In the financial services sector regulators (both internal and external) require corporations not only to measure their risk, but also to hold a sufficient amount of capital to cover potential losses given that risk. Another example is in emergency service vehicle routing, where one needs to be able to reliably get to a destination within a fixed limit of time, rather than taking a route which may have a shorter expected travel time but could, under certain travel conditions, take significantly longer [Samaranayake et al., 2012]. Further examples can be found in food hygiene [Pouillot et al., 2007] and technology infrastructure [Buyya et al., 2009].

In the first part of the thesis we consider the implications of risk in portfolio optimisation. We construct an algorithm which allows for the efficient optimisation of a portfolio at various risk points. During this work we assume that the value at risk can only be estimated via sampling; this is because it would be near impossible to analytically capture the probability distribution of a large portfolio. We focus initially on optimising a single risk point but later expand the work to the optimisation of multiple risk points. We study the ensemble defined by the algorithm, and also various approximations of it are then used to both improve the algorithm but also to question exactly what we should be optimising when we wish to minimise risk. The key challenge in constructing such an algorithm is to consider how much the optimisation method biases the samples used to estimate the value at risk. We wish to select genuinely better solutions; not just solutions which were somehow lucky, and hence treated more favourably, during the optimisation process.

In the second part of the thesis we switch our focus to considering how we can understand when large losses will occur. In the financial services sector this translates to asking the question: under what market conditions will I make a (very) significant loss, or even go bankrupt? We consider various methods of answering this question. The initial algorithm relies heavily on an understanding of how our portfolio is modelled but we work to extend this algorithm so that no prior knowledge of the system is required.

In the final chapter we discuss some further implications and possible future directions of this work.

Chapter 1

Introduction

In this thesis we will study the optimisation of risk across a variety of systems. We will draw on ideas from mathematics and physics and find applications in finance. The thesis focuses on the examples of portfolio optimisation and reverse stress testing, both of which are introduced in sections 1.2 and 1.3.

The thesis considers several example problems from finance and then tackles them using a combination of tools from both the physical and mathematical literatures. The thesis attempts to draw these literatures closer together as part of this process.

This combination of techniques leads to several highly successful results. The developed algorithms provide a fast and efficient method for solving the given financial problems. The results are generally found by gaining an intuitive understanding of the problem in question. This intuitive understanding then underpins the development of the new algorithms.

1.1 Thesis Structure

The thesis is formed of eight chapters, five of which contain new research. The thesis is broadly split into two parts.

Chapters 3, 4 and 5 discuss how to optimise a given system at a specified level of risk. This is motivated by portfolio optimisation where optimising returns at the mean may not be the best solution (1.2). Over the course of these chapters we will investigate and develop algorithms which are well suited to solving this class of problem. We will test these algorithms numerically, by using simple examples taken from portfolio optimisation. All of the annealing algorithms used in these chapters are summarised in appendix A.

Chapters 6 and 7 switch the focus to reverse stress testing (1.3). These chapters will consider how to estimate the probability of extreme events for various common financial models. We introduce some of the thematic tensions of reverse stress testing in section 1.3 and then suggest some answers to these tensions in the thesis conclusion (chapter 8). This discussion also leads to suggested further work in the field of reverse stress testing.

More specifically, chapter 2 contains information on the definitions and techniques required to fully understand this thesis. That said, it should be possible to gain a thematic and intuitive understanding of the thesis without reading this chapter.

Chapter 3 considers the basic risk optimisation problem where one wishes to optimise a stochastic system at a given level of risk. This is in contrast to the usual types of optimisation problems where it tends to be the expectation or mean of the system that is optimised. The chapter builds up an understanding of both the problem and proposed optimisation technique used to solve it. This foundation is used to develop a new algorithm to quickly solve this particular style of optimisation problem. It is numerically tested on a high dimensional portfolio optimisation problem from finance.

Chapter 4 considers the same problem, but from a more physical perspective. Various approximations, which are common in physics, are applied to the problem to gain further insight in how to improve the algorithm. These insights are tested on a problem similar to that in chapter 3. The chapter also touches on some of the purely physical properties of the system, such as the energy.

Chapter 5 combines the work from chapters 3 and 4 to expand the algorithm to a wider class of problems. Rather than just optimising a system at a single risk point, as in the two previous chapters, we will consider the optimisation of a system at multiple risk points. This is done by further developing the previous analytic results. As before, the resulting algorithm is demonstrated on a financial portfolio optimisation example.

Chapter 6 uses an intuitive understanding of the Normal Inverse Gaussian distribution (2.3.9) to find an effective method for estimating the probability of extreme events. The resulting algorithm is applied to a sample problem from finance (reverse stress testing).

Chapter 7 continues the work of chapter 6 on reverse stress testing. In this chapter an adaptation of thermal integration (a technique from physics) is applied to a reverse stress testing problem. The method shows encouraging results under certain conditions, however the tuning of such a method proves difficult.

The thesis is concluded in chapter 8. This chapter reviews the work undertaken throughout the thesis and extends the discussions at the end of the individual chapters. We also spend considerable time discussing some of the thematic ideas of reverse stress testing introduced in section 1.3 and continued in chapters 6 and 7. Avenues of future work are also discussed in this chapter.

1.2 Portfolio Optimisation

The idea of portfolio optimisation was first formularised by Markovitz [Markowitz, 1952]. The essential idea was to select a set of assets (stocks) which achieved the best risk return for the purchaser. Since stock returns are stochastic in nature some thought needed to be given as to the best target for optimisation. Should it simply be the expectation, or should one also consider the risk associated with the constructed portfolio? In Markovitz's original work he chose to select the stock(s) which had the best return after normalising for the variance. The work also allowed for the covariance of the stocks. When Markovitz's work was first produced, computing power was highly limited and so a very simple approach needed to be taken. Markovitz received a noble prize in 1990 for his work on portfolio selection.

The major criticism associated with Markovitz's original research is that it is based on a Gaussian understanding of stock returns, which is now thought to be highly unlikely [Taleb, 2007]. Furthermore, the resulting portfolio had a high dependence on the covariance matrix which is problematic to correctly estimate. Due to this, many alternatives to Markovitz's approach have been introduced [Kolm et al., 2014].

There are several alternative definitions of the problem which we could use to find the optimal choice of portfolio. These include minimising the risk and maximising the returns using a specified risk aversion factor, minimising the risk with no regard to the return and maximising or minimising the expected return [Fernando, 2000].

The level of risk proves to be an important consideration. Kelly's paper in 1956, [Kelly, 1956], lead to the introduction of Kelly's criterion. This essentially states that the higher the level of risk undertaken, the lower the percentage of capital that should be used to invest in it. Kelly's criterion has gained further importance in recent years after it emerged that it was used by Warren Buffet, a world famous investor [Pabrai, 2007].

During this thesis we will primarily attempt to solve the problem where we wish to minimise the portfolio risk. This could be in the context of hedging where

we wish to effectively close out the risk with respect to a particular risk class (for example foreign exchange rates) or where we are a seller of a variety of financial instruments and wish to reduce our risk in the simplest way possible.

Later in the thesis we will extend these ideas to allow us to calculate the optimal portfolio which both maximises the expected return and minimises the risk according to some risk aversion factor.

This thesis presumes that the most reliable way of stock returns is to use randomly chosen (normalised) historical samples, as opposed to constructing a model from which to draw samples or make observations. This view point is mainly driven by the author's industrial experience. The author appreciates that many financial models are essential in the calculation of market prices and that in many specific cases models are an improvement. The challenge of which model to use however can become highly complex. That said, this preferred approach significantly reduces the number of data points available to estimate the risk, and hence a model may have to be used. When a model is needed, we will generally model stock returns using a generalised hyperbolic distribution (see section 2.3). These distributions have had significant exposure in the literature as a viable model of stock returns in recent years [Venter and de Jongh, 2002; Schmidt et al., 2006; Eberlein and Prause, 1998].

During the thesis we will need to estimate the value of various (small) quantiles using sampled data. We will do this using either R_T^{-1} or R_T^{-2} samples, where we wish to estimate the value of the quantile at $R_T\%$. R_T^{-1} is clearly the minimal number of samples required and R_T^{-2} should prove to be the maximum number of samples one would reasonably expect to be required. The actual number of samples required is dependent on the true distribution of the samples (often unknown), the desired relative accuracy (usually 5%) and the mean of the distribution (also generally unknown). Both [Mazzocchi, 2008] and [Deming, 1990] contain more details and examples. Wilks [Wilks, 1941] took a different approach and constructed a framework which allowed the calculation of the minimum number of samples required to ensure that (at some confidence level) a value worse than that of a specified quantile would exist within a sample. For standard confidence level choices (95%) these choices of n fall in between R_T^{-1} and R_T^{-2} for the choices of R_T we use. For example, according to Wilks, 59 samples are required to ensure that we have at least one sample with a value lower than the 5% quantile.

1.3 Reverse Stress Testing

Stress testing has been part of a risk manager's tool kit for a long time, however it has gained increased prominence following the recent financial crisis. This has been highlighted by the recent (and ongoing) stress tests carried out by the various financial regulators. These include the Supervisory Capital Assessment Program (Federal Reserve), Comprehensive Capital Analysis and Review (Federal Reserve) and the European Union Banking Stress Test Exercise (European Banking Authority) [Schuermann, 2012; Ong et al., 2010].

Reverse stress testing was introduced by the Financial Services Authority (FSA) as a regulatory requirement from the 14th December 2010 with the below definition.

> We introduce reverse stress-testing requirements for firms to identify and assess scenarios most likely to cause their current business models to become unviable. We address concerns about proportionality in relation to these requirements by describing the range of approaches that firms might take. [F.S.A., 2009]

Reverse stress testing aims to find the scenarios which lead to a 'business failure'. This is in contrast to regular stress testing where the scenarios with which a portfolio is shocked are chosen by experts or taken from historical data. [Worrell, 2010]

Both quantitative and qualitative stress tests are required to tackle such a broad problem. In this work we deal with the more quantitative tests/methods looking at market risk.

There is a sparse academic literature on reverse stress testing in market risk, with each paper containing a different formal definition of what constitutes a business failure or a reverse stress test. We summarise the majority of the existing work and give a common mathematical framework for simple comparison.

A small portion of the literature also considers 'credit risk breaking points'. In this literature the relationship between the capital adequacy ratio of a financial institution and the loan default rate (or another suitable risk factor) is calculated. It is then possible to find the point at which a financial institution would need to recapitalise (i.e. when the bank has become insolvent). This method differs from the type of reverse stress covered in the market risk literature, as described below, because it is limited to only one risk factor. It rapidly becomes infeasible to do this type of analysis as the number of risk factors increases. See Worrell [Worrell, 2010]

and Ong [Ong et al., 2010] for further details.

We first attempt to identify what constitutes a business or regulatory failure. One could take the definition of [Grundke, 2011, 2012] and consider a failure situation to be one where a bank makes an unexpected loss on a portfolio (beyond any expected loss) coupled with a rise in the bank's capital requirements (such that the bank can no longer meet them). We could also take the more simplistic approach of [Glasserman et al., 2012] and consider only the loss in value of the portfolio with no regard for any current, or future, regulatory requirements. Skoglund takes a more heuristic approach that uses a similar metric to that of Glasserman [Skoglund and Chen, 2009].

Mathematically, we consider $V(P, t_2|X_{t_1})$ to be a random variable for the value of a portfolio P at time t_2 given (multidimensional) market data X_{t_1} recorded at time t_1 ($t_1 < t_2$). We further take $EC(P, t_1)$ to be a function describing the economic capital requirement at time t_1 given the same parameters as for $V(P, t_2|X_{t_1})$. Grundke's definition of a failure, given a tolerance (TOL), could be written as $\langle V(P, t_2|X_{t_1})\rangle - \langle V(P, t_1)\rangle + EC(P, t_1) \leq TOL$. That is the change in expected value and the economic capital requirement at time t_1. In contrast Glasserman's definition could be given by $\langle V(P, t_1)\rangle \leq TOL$. Both are calculated at time t_1.

When considering a definition for a failure function we must also take into account the user of these stress tests. A regulator is probably more interested in a reverse stress test that captures all of the bank's liabilities (including regulatory capital). Conversely, an individual trader is probably much more concerned with possible down turn events on only his portfolio. Different users may also have interests in different time horizons. You would expect a regulator to take a more long term view than the leadership team of an individual bank.

Having decided on a loss/failure function one also needs to consider what the scenarios of interest are. Grundke suggests that we need only be concerned with the scenarios along the boundary of failure, whereas Glasserman considers all scenarios that lead to a failure. Mathematically, Grundke considers $S_{Gr} := \{X_t|M(P, X_t) = TOL\}$ and Glasserman $S_{Gl} := \{X_t|M(P, X_t) \leq TOL\}$ where $M(P, X_t)$ is the metric used to classify the bank's financial position. $M(P, X_t)$ could be either of the definitions above. Grundke's approach implicitly requires some form of continuity constraint on $M(P, X_t)$ to ensure that it captures all of the appropriate scenarios. This seems unlikely in the context of complex options that would lead to a discontinuous pay-off function. Grundke's method also implies that the likelihood of individual scenarios versus the value of the portfolio under that scenario is strictly increasing (at least on the boundary locus).

Both Grundke and Glasserman highlight the importance of discovering the most likely scenario that causes failure as being the main aim of reverse stress testing ($\arg\max_{X_t \in S} P(X_t)$). Glasserman however goes on to discuss the method of drawing scenarios from the set S_{Gl}, where the scenarios are weighted according to their likelihood. To do this he makes some broad assumptions that are discussed later in the introduction. Grundke's method for calculating the reverse stress test first requires finding all of S_{Gr} and so he examines each of the scenarios he finds.

Skoglund takes a differing approach in that he uses (Kulback) information theory to find the scenarios most indicative of a financial failure. This enables him to make statements that are more generic about the regions of concern. The main body of his paper involves applying information theory to the portfolio value given the value of the underlying risk factors. This allows him to identify the most 'informative' risk factors. He argues that this heuristic can be used to highlight dangerous scenarios in a more loose sense. This method is less vulnerable to model errors, which may occur in extreme events. His method does appear to assume that the important risk factors are the same for whatever the region of risk factor space you are currently in. This seems unlikely as it is in large loss situations when hedges often fail - which would lead to a change in the 'key' risk factors in a portfolio.

Both Grundke and Glasserman state the importance of a bottom up approach. A bottom up approach involves modelling the underlying instruments and risk factors and then calculating the portfolio value (deterministically) from these. Further information can be found in [Rebonato, 2010; Grundke, 2011; Breuer et al., 2010; Ong et al., 2010].

Grundke uses a grid search combined with a Monte Carlo simulation at each point to find S_{Gr}. Once he has found S_{Gr} in its entirety he assumes that the risk factors follow a normal distribution to get a probability of each scenario occurring. This is a very computational intensive process and is infeasible for a large number of risk factors.

Glasserman uses an empirical likelihood estimator to estimate the conditional mean of market factors given large losses. Then, by assuming a linear loss function, he scales these means appropriately to give the result for his reverse stress test. It is worth noting that (as highlighted in his paper) a linear assumption is only valid in the case when there is only one large loss region. Another potential issue is that at the mean the bank may actually be solvent. Consider a risk factor which causes a bank to fail when it has an extremely low or high value. If both extreme events have an equal chance of occurring then the mean would be a 'normal' value of the risk factor. Glasserman also creates confidence intervals for the most likely

scenario leading to a large loss. Using these intervals he generates further scenarios which lead to large losses. The sampling method encapsulates the probability of the scenario occurring in some sense. However it does this heuristically as there is no clear reason why these confidence intervals should relate to the actual marginal distributions involved.

In this thesis we will focus on the idea that reverse stress testing focuses solely on the direct losses of a portfolio and not the losses inferred from changing capital requirements. This is partly because it is unlikely that a regulator would force a bank or similar institution to fail in these circumstances. We will focus on the estimation of the probability of extreme values occurring for stocks modelled using well known distributions. In the conclusion we will return our focus to some of the more thematic questions raised in this introduction.

Chapter 2

Background

2.1 Introduction

In this chapter various background definitions and results, which are helpful in understanding the thesis, are provided. Initially, there is a description of some common mathematical tools and definitions, including the definition of a Markov chain, which will be key to understanding the majority of the thesis.

Four further introductory sections are then provided. These cover several statistical distributions, basic approximations, sampling methods and finally optimisation under uncertainty.

Some limited grouping of the background is provided in this chapter. The relevance and connections between the different ideas introduced in this chapter should become clear as the reader progresses through the thesis.

2.2 Definitions

Several key definitions and ideas are provided in this section. Knowledge of these definitions is required for a clear understanding of the thesis.

2.2.1 Markov Chains

A stochastic process is a sequence of random variables, which are indexed by time. This sequence of random variables usually contains some history and time dependence. A stochastic process might be formally defined as $\{X_t : t \in \mathbb{N}\}$ on some state space $\mathbf{S}$. A Markov chain is a discrete time stochastic process where the next state is only dependent on the current state (X_t) and not the entire historical path of the chain, $\{X_{t'} : t' \leq t\}$.

Definition 2.2.1 (Markov Chain)**.** A Markov chain, X, is a sequence of random variables which obeys the Markov property [Finch, 1963]. A discrete time stochastic process $X := \{X_t : t \in \mathbb{N}\}$ on a state space $\mathbf{S}$ is a Markov chain if

$$P\left(X_t \in \mathbf{A} | X_0 = x_0, X_1 = x_1, \ldots, X_{t-1} = x_{t-1}, t\right) = P\left(X_t \in \mathbf{A} | X_{t-1} = x_{t-1}, t\right)$$

holds $\forall \mathbf{A} \subset \mathbf{S}, \forall x_1, \ldots, x_{t-1} \in \mathbf{S}$ and $\forall t \in \mathbb{N}$.

Furthermore, we only consider time-homogeneous Markov chains. This means that the chain moves have no time dependence.

Definition 2.2.2 (Time-Homogeneous Markov Chain)**.** A Markov chain $X := \{X_t : t \in \mathbb{N}\}$ on a state space $\mathbf{S}$ is a Time-Homogeneous Markov chain if

$$P\left(X_t \in \mathbf{A} | X_0 = x_0, X_1 = x_1, \ldots, X_{t-1} = x_{t-1}, t\right) = P\left(X_t \in \mathbf{A} | X_{t-1} = x_{t-1}\right)$$

holds $\forall \mathbf{A} \subset \mathbf{S}, \forall x_1, \ldots, x_{t-1} \in \mathbf{S}$ and $\forall t \in \mathbb{N}$.

A full discussion on Markov chains can be found in [Rubinstein and Kroese, 2008].

2.2.2 The Gamma and Beta Functions

Knowledge of the Gamma and Beta functions will aid us in various integration problems later in the thesis.

Definition 2.2.3 (The Gamma Function)**.** The Gamma Function, denoted $\Gamma(z)$, is defined to be

$$\Gamma(z) = \int_0^\infty x^{z-1} e^{-x} dx$$

for any choice of $z \in \mathbb{C}$. If $t \in \mathbb{N}$, the definition can be simplified to $\Gamma(t) = (t-1)!$. $\Gamma(z)$ converges absolutely for any choice of z such that $Re(z) > 0$.

Proposition 2.2.4 (Stirling's Approximation)**.** *Stirling's Approximation gives an estimate of the Gamma Function (2.2.3). The approximation is given by*

$$\Gamma(z) = z^{z-\frac{1}{2}} e^{-z} \sqrt{2\pi} \gamma(z) \text{ where}$$
$$\gamma(z) = \left(1 + \frac{1}{12z} + \frac{1}{288z^2} - \frac{139}{51840z^3} - \frac{571}{2488320z^4} + O\left(\frac{1}{z^5}\right)\right).$$

A proof can be found in [Feller, 1968].

Definition 2.2.5 (The Beta Function)**.** The Beta Function, denoted $B(x, y)$, is defined to be

$$B(x,y) = \int_0^1 t^{x-1}(1-t)^{y-1}dt = \frac{\Gamma(x)\Gamma(y)}{\Gamma(x+y)}$$

for any choice of $x, y \in \mathbb{C}$ such that $Re(x), Re(y) > 0$.

2.3 Distributions

The definitions of the three distinct distributions that are used during this thesis are provided in this section. Brief definitions of the Normal and Boltzmann distributions are provided, although it is largely assumed that the reader will be familiar with these distributions. More detail is provided regarding the Generalised Hyperbolic distributions, as it is likely the reader will be less familiar with this class of distribution.

In the definitions below we use the convention that an emphasised (bold) variable represents a random variable.

2.3.1 Normal Distribution

The Normal (or Gaussian) distribution is one of the most well known statistical distributions. Many sources can be used to find further information, for example [Glasserman, 2004]. The Normal distribution has been applied to a vast variety of problems. During the course of this thesis we will use both the univariate and multivariate definition.

Definition 2.3.1 (Normal Distribution)**.** A normal distribution is classified by $\mu \in \mathbb{R}$ and $\sigma^2 \in \mathbb{R}_{>0}$. It has a probability density function of

$$\phi_{\mu,\sigma}(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(x-\mu)^2}$$

and cumulative distribution function

$$\Phi_{\mu,\sigma}(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\frac{x-\mu}{\sigma}} e^{-\frac{1}{2}u^2} du.$$

We will use the common notation that $\phi(x) = \phi_{0,1}(x)$ and $\Phi(x) = \Phi_{0,1}(x)$.

Definition 2.3.2 (Multivariate Normal Distribution)**.** A k dimensional normal distribution is classified by a mean, $\mu \in \mathbb{R}^k$, and a covariance matrix variance,

$\Sigma \in \mathbb{R}^{k\times k}_{>0}$. It has density

$$\phi_{\mu,\Sigma}(x) = \frac{1}{(2\pi)^{\frac{k}{2}}|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}$$

where $x \in \mathbb{R}^k$.

2.3.2 Boltzmann Distribution

The Boltzmann distribution is ubiquitous in physics. It is also known as the Gibbs distribution. Further information can be found in [Minlos, 1999]. The Boltzmann distribution will play a central role in the optimisation methods used during the thesis.

Definition 2.3.3 (Boltzmann Distribution)**.** Given a constant, $k \in \mathbb{R}_{>0}$, a temperature, $T \in \mathbb{R}_{>0}$, and an energy function, $E(x)$, the Boltzmann distribution is the probability distribution over the possible states of the system. Each state has weight

$$W(x) = e^{-\frac{1}{kT}E(x)}.$$

During the course of the thesis we will relabel kT as β^{-1}.

2.3.3 Generalised Hyperbolic Distributions

Generalised hyperbolic distributions were first studied by Barndorff-Nielsen in [Barndorff-Nielsen, 1977; Barndorff-Nielsen et al., 1982]. This class of distributions was initially applied to grain size distributions of blown sand [Barndorff-Nielsen, 1977; Olbricht, 1991]. More recently, these distributions have also been extensively applied to financial asset returns, first in [Enerlein and Keller, 1995], and later in [Venter and de Jongh, 2002; Schmidt et al., 2006; Eberlein and Prause, 1998].

One of the plausible intuitions behind the use of these distributions in finance is the idea of market time. In this ideology we can imagine that stock returns are normally distributed with respect to the market's internal clock, however the relationship between market and 'real' time is itself stochastic. This stochastic shift in time could be considered to be the market's rate of response to new information. We assume that the market time is common across all stocks. There is empirical evidence for correlation in the market that supports this assumption. Some authors have gone further to consider the idea of each stock having its own time, but this

is beyond the scope of this work. These ideas are further explored in [Luciano and Semeraro, 2008].

In order to formally define the generalised hyperbolic class of distributions, we need to first define both the generalised inverse Gaussian distribution and normal variance-mean mixture models.

Definition 2.3.4 (Generalised Inverse Gaussian)**.** A random variable $\mathbf{X}$ is said to have a Generalised Inverse Gaussian (GIG) distribution if its density function is given by

$$f(x) = \left(\frac{\psi}{\chi}\right)^{\frac{\lambda}{2}} \frac{x^{\lambda-1}}{2K_\lambda(\sqrt{\chi\psi})} e^{-\frac{1}{2}\left(\frac{\chi}{x}+\psi x\right)}$$

where $\chi > 0, \psi \geq 0$ for $\lambda < 0$, $\chi \geq 0, \psi > 0$ for $\lambda = 0$ and $\chi > 0, \psi > 0$ for $\lambda < 0$, $\psi, \chi, \lambda \in \mathbb{R}$ are parameters. K_λ is the Modified Bessel Function of the third kind[1].

Definition 2.3.5 (Inverse Gaussian)**.** A random variable $\mathbf{X}$ is said to have an Inverse Gaussian (IG) distribution if it has a GIG distribution with $\lambda = -\frac{1}{2}$. The cumulative distribution function of the IG distribution is given by

$$F(x) = \Phi\left(\frac{x\sqrt{\psi} - \sqrt{\chi}}{\sqrt{x}}\right) + e^{\sqrt{4\psi\chi}}\Phi\left(\frac{-x\sqrt{\psi} - \sqrt{\chi}}{\sqrt{x}}\right)$$

where Φ is the normal cumulative distribution function.

We further define normal variance-mean mixture models, of which generalised hyperbolic distributions are a subclass.

Definition 2.3.6 (Normal Variance-Mean Mixture)**.** A random variable $\mathbf{X} \in \mathbb{R}^d$ is said to have a normal variance-mean mixture model distribution if

$$\mathbf{X} := \mu + \mathbf{W}\gamma + \sqrt{\mathbf{W}}A\mathbf{Z}$$

where

- $\mu, \gamma \in \mathbb{R}^d$ are drift parameters,
- $A \in \mathbb{R}^{d\times k}$ are correlation parameters,
- $\mathbf{Z}$ is a k dimensional normal distributed variable ($\mathbf{Z} \sim \mathcal{N}_k(\mathbf{0}, I_k)$) and
- $\mathbf{W}$ is a positive, scalar valued random variable which is independent of $\mathbf{Z}$.

[1] A Modified Bessel Function of the third kind has integral representation (for $x > 0$) $K_\lambda(x) = \frac{1}{2}\int_0^\infty w^{\lambda-1}e^{-\frac{1}{2}x(w+w^{-1})}dw$.

Remark 2.3.7. From the definition it is easy to see that for a normal variance-mean mixture model we have that

$$\mathbf{X}|\mathbf{W} \sim \mathcal{N}_d(\mu + \mathbf{W}\gamma, \mathbf{W}\Sigma)$$

where $\Sigma = AA'$.

Definition 2.3.8 (Generalised Hyperbolic Distribution)**.** A random variable $\mathbf{X}$ is said to have a Generalised Hyperbolic (GH) distribution if it is a Normal Variance-Mean Mixture distribution with a mixing variable, $\mathbf{W}$, that is distributed according to a GIG distribution. This is denoted $\mathbf{X} \sim GH(\lambda, \chi, \psi, \mu, \Sigma, \gamma)$ where the parameters are as described in 2.3.4, 2.3.6 and 2.3.7.

The GH class of distributions also has some special cases, which are defined below.

Definition 2.3.9 (Normal Inverse Gaussian)**.** A random variable $\mathbf{X}$ is said to have a Normal Inverse Gaussian (NIG) distribution if it has a GH distribution with $\lambda = -\frac{1}{2}$. In this case the mixture distribution is an IG distribution (see definition 2.3.5).

Definition 2.3.10 (Hyperbolic Distribution)**.** A random variable $\mathbf{X}$ is said to have a (d-dimensional) Hyperbolic distribution if it has a GH distribution with $\lambda = \frac{d+1}{2}$.

Definition 2.3.11 (Skewed t)**.** A random variable $\mathbf{X}$ is said to have a Skewed-t distribution if has a GH distribution with $\lambda = -\nu$, $\chi = \nu$ and $\psi = 0$, for some parameter ν.

Each of these distributions (Normal Inverse Gaussian, Hyperbolic and Skewed t) are being used increasingly in finance to model stock returns [Hu and Kercheval, 2007].

We further note that the GH class of distributions are closed under linear transformations.

Proposition 2.3.12. *If* $\mathbf{X} \sim GH_d(\lambda, \chi, \psi, \mu, \Sigma, \gamma)$ *and* $\mathbf{Y} \sim B\mathbf{X} + \mathbf{b}$ *where* $B \in \mathbb{R}^{k \times d}$ *and* $\mathbf{b} \in \mathbb{R}^k$*, then* $\mathbf{Y} \sim GH_k(\lambda, \chi, \psi, B\mu + \mathbf{b}, B\Sigma B', B\gamma)$.

Proof. A proof can be found in [Breymann and Luthi, 2013]. □

Proposition 2.3.13. *Generalised Inverse Gaussian Mean If* $\mathbf{X} \sim GIG(\lambda, \chi, \psi)$ *then the expectation of* $\mathbf{X}$ *is given by*

$$\mathbb{E}[\mathbf{X}] = \sqrt{\frac{\chi}{\psi}} \frac{K_{\lambda+1}\left(\sqrt{\chi\psi}\right)}{K_\lambda\left(\sqrt{\chi\psi}\right)}$$

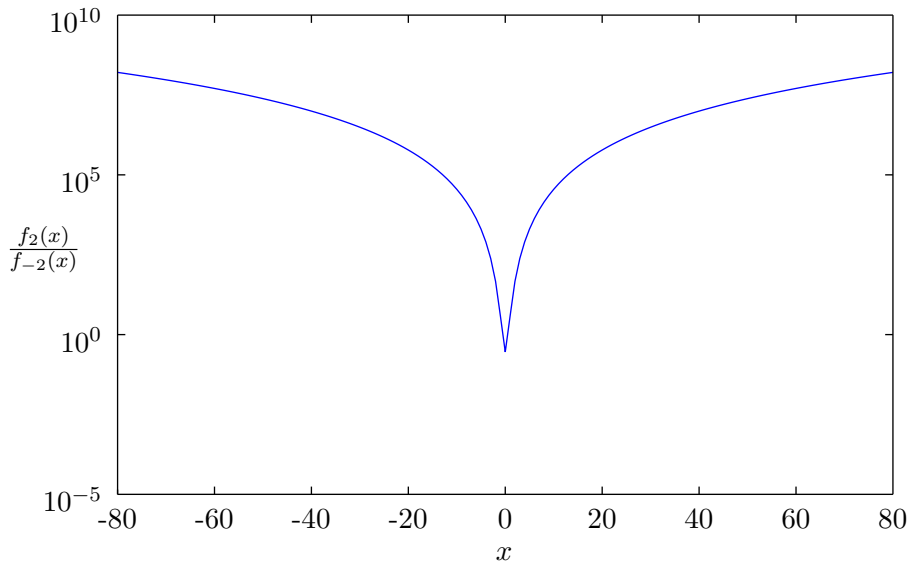


Figure 2.1: The ratio of the Generalised Hyperbolic probability density function, f, when $\lambda = 2$ and $\lambda = -2$. The remaining parameters are constant, with $\chi = \frac{1}{2}, \psi = 2, \mu = 0, \sigma = 1, \gamma = 0$. The exponentially increasing ratio between the two density functions indicates that the distribution given by $\lambda = 2$ has significantly heavier tails.

where K_λ is the Modified Bessel Function of the third kind.

Proof. A proof can be found in [Breymann and Luthi, 2013]. □

Figure 2.1 shows the significant impact on the tail shape of the Generalised Hyperbolic distribution that larger choices of λ have. In our real world examples we cover cases with a wide variety of parameter choices.

2.4 Approximations

In this section two common physical approximations are provided, along with some proof. These will each prove essential in understanding and developing results throughout the thesis.

Notationally, both here and throughout the thesis we use $\approx$ to denote an approximation whose error goes to zero in the limit stated (or inferred from a preceding result). (We reserve $\sim$ for being distributed as.)

Proposition 2.4.1 (Laplace's Method)**.** *For a given function $f(x)$, that is twice differentiable on the real interval $[a, b]$, maximised at $x_0 \in (a, b)$, $f(x_0) \neq 0$ and has*

$f''(x_0) < 0$ *we have that*

$$\int_a^b e^{\beta f(x)} dx \approx \frac{\sqrt{2\pi}}{\sqrt{-\beta f''(x_0)}} e^{\beta f(x_0)}$$

where $\beta \in \mathbb{R}_{>0}$. *This becomes equality in the limit of large* β *(as* $\beta \to \infty$*).*

Proof. Variants of the proof can be found in [Azevedo-Filho and Shachter, 1994; Wong, 2001; Gajjar, 2010; Barndorff-Nielsen and Cox, 1989]. □

Remark 2.4.2 (Laplace's Method: Leading Order)**.** To leading order this becomes

$$\int_a^b e^{\beta f(x)} dx \doteq \frac{\sqrt{2\pi}}{\sqrt{\beta}} e^{\beta f(x_0)}$$

where $\doteq$ represents a leading order approximation for the logarithm of the right hand side of proposition 2.4.1.

Corollary 2.4.3. *For a given choice of* $p \geq 1$, $x < 0$ *and* $\beta \in \mathbb{R}_{>0}$ *we have*

$$\int_{-\infty}^{x} e^{-\beta|x'|^p} dx' \approx \frac{e^{-\beta|x|^p}}{\beta p|x|^{p-1}}.$$

This becomes equality in the limit of large β *(as* $\beta \to \infty$*).*

Proof. Note that $\frac{d}{dx}(|x|^p) = -p|x|^{p-1}$ (as $x < 0$). Proof then follows from [Gajjar, 2010] with variants of the proof contained in [Wong, 2001; Azevedo-Filho and Shachter, 1994; Barndorff-Nielsen and Cox, 1989]. □

2.5 Sampling Methods

In this section we provide background on a variety of techniques used to sample probability distributions.

2.5.1 Monte Carlo Sampling

Monte Carlo sampling at its simplest level is the estimation of an expectation of a distribution by drawing random samples from the distribution. For example, we could estimate the mean of a normal distribution (if we did not already know it) by $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i$ where $x_i \sim N(0, 1)$. Further information can be found in [Liang et al., 2010; Frenkel and Smit, 2002; Gamerman and Lopes, 2006].

2.5.2 Importance Sampling

Importance sampling is a well known variance reduction method for Monte Carlo sampling. It has been applied to a large variety of problems, from physics to finance [Glasserman, 2004; Asmussen and Glynn, 2007; Rubinstein and Kroese, 2008].

Importance sampling works to reduce the variance (error) of a sampled expectation by adjusting the probability measure used to select the random samples. This new probability measure ensures that we sample the area of interest (which generally contains most variance) more frequently than we would using the original distribution [Glasserman, 2004; Asmussen and Glynn, 2007]. In other words importance sampling allows us to focus our search.

Mathematically, suppose we wish to estimate the expectation of a function, $V(X)$, over a probability distribution with density $f(x)$. By expanding the expectation we see that

$$\langle V(X) \rangle_f = \int_{\mathbf{X}} V(x) f(x) dx.$$

We further assume $V(X)$ is dominated by the probable values of another density function, $g(x)$, and require that $g(x) = 0 \implies V(x)f(x) = 0$. We can then rewrite the expectation above as

$$\langle V(X) \rangle_f = \int_{\mathbf{X}} V(x) \frac{f(x)}{g(x)} g(x) dx = \left\langle V(X) \frac{f(X)}{g(X)} \right\rangle_g.$$

The ratio between the probability densities, $\frac{f(x)}{g(x)}$, is known as the likelihood ratio [Rubinstein and Kroese, 2008]. This means we can now randomly draw samples from $g(x)$ and yet still obtain an estimate of $\langle V(X) \rangle_f$.

It has been argued that this likelihood ratio degenerates when it is over a high dimensional random variable, $\mathbf{X}$. [Rubinstein and Glynn, 2009; Rubinstein and Kroese, 2008]. The argument is constructed by assuming that the components of $\mathbf{X}$ are i.i.d. with respect to both f and g. This implies that both f and g are formed by multiplying their marginal probabilities $f_1, \ldots, f_n$ and $g_1, \ldots, g_n$ respectively. Using this information we can write the likelihood ratio, $W(X)$, as

$$W(X) = e^{\sum_{i=1}^{n} \ln \frac{f_i(X_i)}{g_i(X_i)}} = e^{\sum_{i=1}^{n} \ln \frac{f_1(X_i)}{g_1(X_i)}}$$

where X_i are the components of $\mathbf{X}$, given that all components have the same

marginal. By the law of large numbers this is expected to approach

$$W(X) \approx e^{-n\left\langle -\ln \frac{f_1(X_1)}{g_1(X_1)} \right\rangle_{g_1}}$$

as n become large. Since we know by Jensen's inequality [Jensen, 1906] that

$$\left\langle -\ln \frac{f_1(X_1)}{g_1(X_1)} \right\rangle_{g_1} \geq -\ln \left(\left\langle \frac{f_1(X_1)}{g_1(X_1)} \right\rangle_{g_1} \right) = -\ln 1 = 0,$$

and given we do not expect this bound to be reached, we can see that $W(X) \to 0$ as $n \to \infty$. However, by definition $\mathbb{E}[W(X)] = 1$, implying that $W(X)$ becomes very skewed for large n. A consequence of this skew is that the variance of $W(X)$ may become very high [Rubinstein and Kroese, 2008]. This in turn will drive up the variance of our estimator for $\langle V(X) \rangle_f$.

2.5.3 Rare Event Sampling

Rare event sampling allows for the calculation of extreme probabilities, often of the form $P(S(\mathbb{X}) > \gamma)$ where $\mathbb{X}$ is a set of N random variables. There is a significant literature covering this field. The basic test problems often consist of either estimating $P\left(\sum_{i=1}^{N} X_i > \gamma\right)$ or considering a simple bridge network [Chan and Kroese, 2011].

Asmussen [Asmussen and Glynn, 2007] provide further background on rare event sampling. Asmussen describes a variety of algorithms used in rare event sampling, the most prevalent being multi-level sampling, which is beyond the scope of this thesis.

The ideas of Chan [Chan and Kroese, 2011] are developed as part of this thesis. Chan considers the probability that the sum of d heavy tailed (Pareto) distributions has a very large value. He notes that the probability of a rare event in this problem is likely to be driven by a single variable (due to its heavy tailed nature), rather than multiple variables. This simple insight is then used to derive estimators that have reduced variance, compared to a simple likelihood ratio. This can be done as the cumulative distribution function is known for a one dimensional Pareto distribution. The application of such an argument to other distributions will be considered later in the thesis.

For example, imagine that there are n heavy tailed (Pareto) random variables, X_i for $1 \leq i \leq n$, and that we wish to calculate $P\left(\sum_{i=1}^{n} X_i > L\right)$. By drawing random samples from the first $n-1$ random variables, $x_i \sim X_i$, we can calculate

the value required from the last random variable for the sum to be greater than L, label this $v = L - \sum_{i=1}^{n-1} x_i$. Since the one dimensional cumulative distribution function is known, we can calculate the probability of X_n having a value of v or greater, i.e. $P\left(\sum_{i=1}^{n} X_i > L | x_1, \ldots, x_{n-1}\right) = P(X_n > v)$. Averaging this calculation over a large number of samples will give an estimate of the conditional probability $P\left(\sum_{i=1}^{n} X_i > L | X_1, \ldots, X_{n-1}\right)$. Repeating this process for each choice of random variable, X_i, will give similar conditional probabilities for each random variable. By the key observation above, these conditional probabilities should quickly converge and summing them should give a good estimate of $P\left(\sum_{i=1}^{n} X_i > L\right)$. Note that such a method is closely linked to the Rao-Blackwell Theorem [Gelfand and Smith, 1991].

2.5.4 Thermal Integration

Thermal integration is most commonly used in physics to study the solid-liquid transition [Frenkel and Smit, 2002].

The process works by considering two different systems, A and B, and their associated potentials, U_A and U_B respectively. The two systems are then coupled to form a new potential $U = U_A + \lambda(U_B - U_A)$, where $0 \leq \lambda \leq 1$. The delta between the free energy of the systems can then be calculated as $F(B) - F(A) = \int_0^1 d\lambda \left\langle \frac{dU(\lambda)}{d\lambda} \right\rangle$ where the expectation is over the Boltzman distribution with potential $U(\lambda)$. Note that the partition function of a system is effectively the normalisation constant of the potential function over all possible configurations. The free energy is proportional to the natural logarithm of the partition function.

The result allows the calculation of unknown free energies, assuming a similar system has a known free energy. In the example above (solid-liquid transition) the solid state free energy was known but the liquid state free energy was not.

A much more detailed explanation, alongside physical examples, can be found in [Frenkel and Smit, 2002; White and Meirovitch, 2004].

2.5.5 Extreme Value Theory

Extreme Value Theory focuses on the tails of a distribution.

There are two main types of extreme value models. The first are block maxima models. These model the maximum value in a block of values. For example, if we were to record daily losses this type of model could be used to estimate the largest daily loss to occur in a given quarter or year.

The second set of models are known as Peak Over Threshold (POT) models. These model the distribution of samples which exceed a certain threshold. For

example, this could be the distribution of losses which exceed a certain value. This method uses data more effectively than the block maxima method (i.e. less data is required to fit the model) and hence appears to have become preferred [Gilli and Kellezi, 2006].

A full description of how these models are constructed can be found in [Gilli and Kellezi, 2006; Neves and Alves, 2008].

Whilst these methods are very well established, they force a top down modelling approach for financial portfolios. This has two critical consequences:

1. It becomes very hard to drill down into where risk is coming from. Whilst knowing the extreme values a particular portfolio might take in the next quarter is useful, if you do not know what is causing the risk then there is little you can do about it!

2. The models cannot be aggregated in a sensible way. That is, the predicted extreme risks cannot be aggregated across a set of portfolios as the different portfolios may contain correlations which might reduce (or increase) the risk of an extreme loss.

The above arguments mean that separate models would need to be fitted to each layer of possible aggregation. This obviously creates a huge amount of work and the resulting models might not be self-consistent (depending on the data used). This additional modelling burden is further complicated by the fact that there is no standard way to calculate the optimal threshold (for POT models) [Wong and Li, 2010].

In chapters 6 and 7 on reverse stress testing we instead take a bottom up approach to modelling extreme risk. In these chapters we model each stock individually (although not independently) before combing individual models to generate a model for a portfolio. In chapter 7 we begin to discuss how the described methods could be used to efficiently search a risk space to locate the source of different risks. The methods used also allow us to use the same model to look at either the portfolio risk, or the individual stock's risk, without the need to fit further models.

2.6 Optimisation Under Uncertainty

Performing optimisation under uncertainty has become an increasingly important challenge in the optimisation literature. The challenge here is to optimise a function which can be only be estimated or sampled for some given choice of design. This is

often true whenever we wish to optimise a real system. It is hard to find a 'real life' decision that is devoid of uncertainty, and hence risk [Better et al., 2008].

There are a huge number of possible optimisation problems and associated techniques to solve them [Press et al., 2007]. In this thesis we focus on problems where we wish to find the global optimum, or at least a good local optimum, within a large search space and only have access to estimates of the objective function.

The work of [Beyer and Sendhoff, 2007] splits the common approaches to optimisation under uncertainty into three broad categories (and provides further details):

1. Monte Carlo Strategies: Given a fixed design point, various statistics are calculate (such as expectation) and then a deterministic optimisation method is used.

2. Meta-Model Approach: A meta-model is constructed using a set of carefully chosen design points. The meta-model is then optimised.

3. Estimates of the objective function are used directly in an optimisation algorithm that is specifically designed for noisy optimisation.

We note that methods in the first category are likely to be biased to design points that receive unusually good sampling (as they are more likely to be selected). This can be explicitly seen later in chapter 5, when we attempt to optimise the expected outcome of a portfolio. This is of particular concern if the error distribution is not the same for each design point.

Meta-Model optimisation can be effective when the search space is small but such methods often struggle for larger spaces [El-Beltagy et al., 1999]. Furthermore, even fitting the model to the underlying objective function is a problem of interest. A high level of confidence is needed in the meta-model to ensure that the problem has been correctly optimised [Beyer and Sendhoff, 2007].

Due to these issues we will focus on developing an algorithm which works directly with noisy estimates of the objective function. To this end, we provide further background on some methods that are designed specifically for optimisation under uncertainty.

2.6.1 Stochastic Approximation Techniques

Stochastic approximation techniques are akin to gradient search methods in deterministic optimisation. The ideology behind these techniques was first published in the 1950s by [Robbins and Monro, 1951].

Essentially, the method solves the problem where we wish to find the maximum of an unknown function, $M(x)$. The method assumes that we have a function, $N(x)$, that provides observations of $M(x)$ such that $\langle N(x) \rangle = M(x)$.

The algorithm then follows the below (gradient like) method (for some sensible choice of x_0) with each iteration generated as

$$x_{i+1} = x_i + a_i \left(\frac{N(x_i + c_i) - N(x_i - c_i)}{c_i} \right)$$

where the sequences $\{c_i\}$ and $\{a_i\}$ specify the finite difference widths and the positive step sizes taken in that direction respectively.

It can be shown that such an algorithm converges to a local maximum (after making some quite stringent assumptions on $M(x)$) but that convergence is very slow [Papadrakakis et al., 2005]. Due to these challenges it is not a suitable method for our proposed problem as there may well be local maxima in the space and we wish to find a global maximum.

2.6.2 Pattern Search Methods

Pattern search methods are an adaptation of gradient search methods that do not require the estimation of the gradient of the objective function. They instead generate search points according to a pattern and accept those points that appear as improvements over the prior search points. A review of the different patterns used can be found in [Wright, 1995].

A noisy adaptation of pattern search methods has been formulated in [Anderson and Ferris, 2001] alongside a proof of convergence. However, the proof is based on the assumption that the standard deviation of the noise reduces faster than the step size when approaching the optimum (and that the error is normally distributed). Theoretical investigation as to the solution quality in the case that the noise does not decrease (as is normally the case) has not been done [Beyer and Sendhoff, 2007].

Unfortunately, pattern search methods do not perform well on highly complex or high dimensional problems [Wen et al., 2013]. Given we are focusing on problems that are both complex and high dimensional we do not discuss pattern search methods further.

2.6.3 Evolutionary Algorithms

Evolutionary algorithms were originally developed for deterministic systems but have now been extended to noisy problems. [Eiben and Smith, 2007] provide a full introduction to (deterministic) Evolutionary Algorithms and a summary is provided below.

Evolutionary algorithms are based on the principles of biological evolution. Starting with an initial population of candidate designs and the corresponding observed objective function values, an offspring population of designs is created from the parents using various operators. These operators can both combine (known as recombination) and mutate the parent candidates. The candidates which show best fit with regards to the objective function then survive and go on to form the next generation. The bad candidates 'die'. This (repeated) process creates a set of evolved solutions, which should have good fitness to the given objective function.

Deterministic evolutionary algorithms could be simply used to solve noisy optimisation problems as a type of "Monte Carlo Strategy" (see above), however, numerous efforts have been made to adapt the algorithm specifically to noisy problems. Most of these adaptations rely on resampling methods which gradually reduce the noise of the chosen estimator of the objective function. The immediate question these algorithms attempt to solve is how many samples of the noisy system should be taken and for which candidates. [Beyer and Sendhoff, 2007] provides an extensive list of examples where noisy evolutionary algorithms have been constructed.

It should be noted that simulated annealing (see below) could be seen as a type of evolutionary algorithm that has only one member in each generation.

2.6.4 Simulated Annealing

Simulated annealing is based on annealing in metallurgy [Cerny, 1985; Kirkpatrick et al., 1983]. The basis of the method is that a physical system which is cooled quickly will generally not reach an optimal energy configuration, and will be prone to defects. By contrast, a physical system which is cooled slowly is much more likely to reach a globally optimal energy configuration and be without defects.

The simulated annealing algorithm applies the same idea to numerical problems. At the start of the optimisation, moves through the search space that lead to a solution which is worse than the current solution are readily accepted. As the optimisation progresses the likelihood of accepting these bad moves reduces. This idea is akin to gradually reducing the temperature of the system. At a high temperature the optimisation is free to move around the search space, irrespective of the quality

of the moves. However, as the temperature reduces the mobility of the optimisation reduces and it is likely to only accept moves that show an improvement over the current choice (state).

To define the algorithm more formally, consider an optimisation where we wish to minimise a function, $f(x)$, over $x \in A \subset \mathbb{R}^d$. First, we consider how to sample from the distribution described by the (relative) density $g(x) = e^{-\beta f(x)}$ for some fixed $\beta \in \mathbb{R}$. Markov Chain Monte Carlo sampling techniques can be used to draw samples from this distribution.

To construct such a chain we must define a transition density between two points in the state space, $q(x \to y)$, where $x, y \in A$. This transition density is composed of two components. The first is the probability of attempting a move to a particular point; we label this the perturbation density, $h(x \to y)$. The second is the probability of accepting such a move, labelled $a(x \to y)$. We then have that $q(x \to y) = h(x \to y)a(x \to y)$. In other words, the transition density is equal to the probability density of considering a move and then accepting that move.

Since we already know the distribution we wish our chain to follow, we can use this to define our choice of acceptance function. The most common choice for $a(x \to y)$ in this case is the Metropolis acceptance function. This is defined to be

$$a(x \to y) = \min\left(\frac{g(y)h(y \to x)}{g(x)h(x \to y)}, 1\right)$$

where $g(x)$ is the density we wish to be sample. Justification as to why a chain using this acceptance function has a stationary distribution with relative density $g(x)$ is provided in [Roberts and Rosenthal, 2004].

In most cases $h(x \to y)$ is chosen to be symmetrical, for example, $h(x \to y) = \phi(x - y)$, where ϕ is the density function of the normal distribution. If $h(x \to y)$ is symmetric and we set $g(x) = e^{-\beta f(x)}$ (as above) then the Metropolis acceptance function in our case becomes

$$a(x \to y|\beta) = \min\left(e^{-\beta(f(y)-f(x))}, 1\right).$$

In simulated annealing we initially run this chain for a fixed amount of time (for some initial choice of β). We then reduce the temperature (i.e. $\beta \uparrow$) and run the chain for a further block of time. Each new chain is initiated using the last value of x accepted by the previous chain. To define this precisely we further require a function which defines how the (inverse) temperature of our system changes, $b(k)$; the number of temperature changes, M; the number of iterations for each temperature change,

L; and an initial state, x_0. The simulated annealing algorithm is then as defined in algorithm 2.1.

The resulting algorithm will, given enough time, converge (in probability) to the global minima of the object function ($f(x)$) [Henderson et al., 2003]. [Gelfand and Mitter, 1989] further showed convergence (in probability) to the global optimum when $f(x)$ could only be estimated and the resulting estimates had a Gaussian error distribution. [Gutjahr and Pflug, 1996] further showed convergence as long as the error distribution was both symmetric and suitably peaked around the correct value. However, in all of these proofs the required rate of cooling would take too long to be feasible in practice and so convergence to the global minima cannot be guaranteed. As described above, we know the exact chain explored for fixed temperature (β^{-1}). However, each time the temperature changes it will take some time for the chain to return to the equilibrium distribution. The behaviour of the chain during this period is unknown.

Algorithm 2.1 Simulated Annealing Algorithm

1: $x := x_0$, $v := f(x)$ and $\beta = 0$
2: **for** $1 \leq k \leq M$ **do**
3: $\quad \beta := b(k)$
4: $\quad$ **for** L steps **do**
5: $\quad\quad$ Pick $\tilde{x}$ subject to $h(x \to \tilde{x})$
6: $\quad\quad$ **if** $a(x \to \tilde{x}|\beta) > u$ s.t. $u \sim U(0,1)$ **then**
7: $\quad\quad\quad x := \tilde{x}$
8: **return** x

Picking an appropriate choice of cooling schedule (as encompassed by $b(k)$) is a known problem of interest. Two common strategies are to either cool the system linearly ($b(k) = \beta_0 + kw$ where $\beta_0, w \in \mathbb{R}$), or to cool the system exponentially ($b(k) = \alpha^{-k}\beta_0$ where $\alpha \in [0,1]$ and $\beta_0 \in \mathbb{R}$) [Chen et al., 2007; Guoa and Zhengb, 2005]. It is worth noting that [Strenski and Kirkpatrick, 1991] do not find any measurable difference between the performance of linear and geometric cooling schedules. However, for convergence to be certain (in probability) a logarithmic cooling schedule needs to be used ($b(k) = \frac{log(k+d)}{c}$ where $c, d \in \mathbb{R}$ although normally $d = 1$). Unfortunately, this strategy is too slow for normal usage [Nourani and Andresen, 1998].

Another consideration is how often to cool the system. In the simple algorithm above the system is cooled every L steps. This is a static schedule. An adaptive schedule could also be used. An adaptive schedule varies the cooling rate using information obtained during the algorithm's execution [Henderson et al., 2003].

For example, the system could instead be cooled every L accepted moves instead of every L attempted moves.

There is no commonly accepted cooling method [Henderson et al., 2003]. In reality, the optimal cooling schedule is often problem specific. Examples of further problem specific cooling schedules can be found in [Kolonko, 1999; Bertsimas and Tsitsiklis, 1993; Thompson and Dowsland, 1998].

During the course of this work we will focus on simple cooling problems. As discussed above, our intention is to consider the scenario when $f(x)$ cannot be calculated exactly and there is some associated error. Ideas similar to this have previously been considered in [Ball et al., 2003].

Chapter 3

Optimising Quantile Risk

3.1 Introduction

In this and the following chapters we discuss and develop ideas on how to optimise a system at a given level of risk. This optimisation, as highlighted in the introduction, allows for the robust decision making in a variety of areas.

The key aim is to optimise a quantile of a stochastic variable where only an estimate of the quantile can be obtained at each step. We do not generally know how good this estimate is, but we can improve this estimate by performing more work.

More specifically, we take the distribution of the random variable to be parametrised by a set of parameters, encapsulated by D. We wish to optimise in the sense of finding the choice of D that leads to the highest value of the quantile at $R_T\%$. Since we do not have access to an inverse cumulative distribution function for the random variable, we are forced to estimate this optimisation quantity using sampling.

This problem has direct links to finance. It is equivalently the optimisation of the Value at Risk (at a specified risk level, $1 - R_T$) of a portfolio. We note that in the financial literature a Value at Risk of $(1 - R_T)\%$ is equivalent to the quantile at $R_T\%$. For example, "95% VaR" is the loss which will be exceeded (on average) 5% of the time, or equivalently, once in every twenty days. In chapter 5 we extend this optimisation problem to include an acceptable risk-reward ratio.

It is now routine in financial services and related sectors to consider the full probability distribution of out-turns, for example for the change in value of a portfolio of investments or of a whole company, projected to some future point in time. Regulators will often impose the requirement to hold a matching amount of capital

to the amount of risk (at a specified level) the institution is currently undertaking, which significantly adds to the cost of risky trading. The connection of value at risk to these capital adequacy requirements is enshrined in leading international accords [Ban, 2011; Cuocoa and Liu, 2006] and national regulatory frameworks.

This thesis will focus on examples from finance but the same methodology could be applied to a wide range of fields from emergency service vehicle routing [Samaranayake et al., 2012] to stock management [Gallego and Ryzin, 1994]. Further examples exist in food hygiene [Pouillot et al., 2007] and technology infrastructure [Buyya et al., 2009].

As is common in complexity science we take inspiration from a variety of fields. Over the course of the next three chapters we look at solving the above optimisation problem with simulated annealing.

In this chapter we build an analytical understanding of the proposed optimisation method. Initially, we define the Markov Chain such an optimisation would follow. Using this result we are able to derive further results, based on the chain, to allow us to create a highly productive feedback loop into our optimisation method. Using these results we construct a new algorithm (based on simulated annealing) which leads to significant time and accuracy gains over the basic algorithm.

In chapter 4 we build on the results contained in this chapter. We explore the thermodynamic ensemble created by the above optimisation method. We use the insights gained from this analysis to drive further improvement of the algorithm proposed in this chapter.

Finally, in chapter 5 we build on the strong foundations constructed in chapters 3 and 4 to further improve and refine our algorithm. These refinements allow us to solve a wider range of problems.

In this chapter we first set up a mathematical description of the problem we are tasked with solving. Using this description we derive the Markov chain that the annealing of this problem would create.

Using this chain we can calculate two key results: the level of risk we are truly optimising at (given the stochastic nature of our estimator) and an estimator of how this risk level changes with the number of portfolio evaluations performed at each step.

With these results in hand we are able to construct an improved annealing algorithm to increase the speed of our optimisation problem. We then discuss some implementation concerns and finally show some numerical results, which show the improvements to be highly effective.

3.2 Problem Definition

In this chapter we wish to optimise the choice of parameters for a random variable such that the value of a specified quantile of the distribution of the random variable is maximised. As highlighted in the introduction, this has relevance in many fields including finance and vehicle routing.

Mathematically, we consider a design space $\mathcal{D}$ of possible designs. Each member of this design space, $D \in \mathcal{D}$, parametrises the probability density for a real valued random variable F_D. We assume $\mathcal{D}$ to be a highly multidimensional continuous measurable space. We define $V_D(R)$ to be the Rth quantile of the distribution of F_D, that is the outcome value of F_D which we have an $R\%$ chance of undershooting, or more formally, $V_D(R) = \inf\{x \in \mathbb{R} : G_D(x) \geq R\}$ where $G_D(x)$ is the cumulative distribution function of F_D $(0 \leq R \leq 1)$. Note that should $G_D(x)$ fail to be injective then this definition of $V_D(R)$ will choose the smallest choice of x which satisfies $G_D(x) = R$. We anticipate that typically we will not have a closed analytical form for $V_D(R)$.

We now define the hedging problem as follows

Definition 3.2.1 (The Quantile Optimisation Problem)**.** Given a design space $\mathcal{D}$ and a fixed choice of $R_T \in [0, 1]$ we wish to find $\arg\max_{D\in\mathcal{D}} V_D(R_T)$; i.e. the value of D that maximises $V_D(R_T)$.

For example, if a choice of D and F_D describes the contents and the possible values of a portfolio (respectively) at a fixed future point in time we find the choice of D which maximises the $(1 - R_T)\%$ Value at Risk. In this case D could have a significant number of dimensions. For example, D might describe the quantities of certain stock or options owned.

An alternative example is a choice of route D where F_D describes the possible times the chosen route could take. We then might aim to find the route that has the shortest arrival time at the 10% quantile. In other words, we wish to minimise $V_D(0.10)$. In a simple scenario there may be two routes. One of these is a rural route, which always takes 10 minutes and the second is a more urban route, which on average takes 8 minutes, but takes 12 minutes 20% of the time. An optimisation on the expected value would select the urban route as it has the lowest average time (8 minutes). However, as described in the problem above, we would select the rural route as it has the shortest travel time at the 10% quantile (10 minutes). In other words, we choose to increase our expected travel time so that we can be more confident of what our individual realisation of the travel time will be.

During this work we further assume that $V_D(R_T)$ cannot be calculated through any practicable formula, so consequently the most economical evaluation of $V_D(R_T)$ is via sampling of the distribution of F_D. The number of samples required to make such an estimation is discussed at the end of section 1.2.

The method we will use to solve this problem is simulated annealing. Simulated annealing had been extensively introduced in section 2.6.4. Our work studies how simulated annealing will operate when faced with the hedging problem, as defined above (3.2.1). We then leverage this knowledge to improve the time and effectiveness of the simulated annealing method.

Simulated annealing is used as we have assumed that $\mathcal{D}$ is high dimensional and hence covers a huge number of scenarios. Grid search (or similar) approaches are unusable for this class of problem due to the large amount of time they would take to run.

3.3 The Markov Chain

In this section we find an analytic description for the Markov chain that our optimisation process will follow (at least when it is at thermal equilibrium). We use this result in later sections as a foundation for further key results, which, as highlighted in the introduction, allow for very effective control of our optimisation process. We use a combination of (noisy) simulated annealing and underlying data samples to perform the quantile optimisation. In this section we assume that inverse temperature β, and the sample size n used to estimate $V_D(R_T)$, are both fixed.

As discussed in the previous section, we assume that we are optimising the design of a random variable, F_D, at a certain risk (or probability) level, R_T. During the optimisation process we will always search $\mathcal{D}$ using a symmetric proposal density, i.e. $h(D \to D') = h(D' \to D)$. Consequently, *if we could* calculate $V_D(R_T)$ precisely at each step, the relative probability density for each design in a simulated annealing process when using the Metropolis acceptance function at fixed β, would be $\pi_a(D) = e^{\beta V_D(R_T)}$. Note that this is the Boltzmann distribution, as defined in definition 2.2.5, with $E = -V_D$.

We now further assume that F_D is composed from m underlying real valued random variables whose joint distribution is independent of D. In other words, it is only this composition of random variables to form F_D which is dependent on D. These underlying random variables are not required to be independent from each other. We then consider $\mathbf{X}$ to be a sample set of n independent, m dimensional, elements each drawn from the distribution of the m random variables from which

F_D is composed. We consider $\mathcal{X}$ to be the space of all possible choices of $\mathbf{X}$.

Since $V_D(R_T)$ is estimated using sampling it will inherently have some error associated with it. The precise error will depend on the specific sample set used. Due to this we can now instead consider a Markov chain which operates over the joint space of $\mathcal{D}$ and $\mathcal{X}$ (instead of just $\mathcal{D}$) where β and n are fixed.

We denote the estimate of $V_D(R_T)$ using $\mathbf{X}$ as $\hat{V}_D(k, \mathbf{X})$ for some (given) choice of k. We define $\hat{V}_D(k, \mathbf{X})$ to be the kth ranked value of V_D obtained from the n elements contained in the sample set $\mathbf{X}$. We would naively expect an appropriate choice of k to be $\lfloor R_T n \rfloor$, however, as we shall see, this may not always be the best choice. More specifically, $\hat{V}_D(k, \mathbf{X})$ is calculated by composing the n sample elements contained in the sample set $\mathbf{X}$ into n samples drawn from the distribution of F_D using the composition parametrised by D. These samples from the distribution of F_D are then put in ascending order and the kth ordered point selected to be $\hat{V}_D(k, \mathbf{X})$.

We will use a similar Metropolis acceptance function for this new Markov chain. That is, for fixed β and k, the probability of accepting a move is

$$a(D, \mathbf{X} \to D', \mathbf{X}') = \min\left(1, e^{\beta\left(\hat{V}_{D'}(k,\mathbf{X}') - \hat{V}_D(k,\mathbf{X})\right)}\right). \tag{3.1}$$

Furthermore, the proposal densities for both D and $\mathbf{X}$ will be mutually independent of each other

$$h(D, \mathbf{X} \to D', \mathbf{X}') = h(D \to D')h(\mathbf{X} \to \mathbf{X}'), \tag{3.2}$$

and, as before, for D, the proposal density will be symmetric,

$$h(D \to D') = h(D' \to D). \tag{3.3}$$

The proposal density for $\mathbf{X}$ will reflect the underlying densities of the random variables from which F_D is composed. A candidate sample set $\mathbf{X}'$ will be constructed by taking n new randomly drawn samples from these underlying distributions. Alternatively, l, where $0 < l < n$, randomly selected elements of the current sample set $\mathbf{X}$ will be replaced by l newly drawn elements. The method used will be consistent throughout a simulation. Hence, by construction, the proposal density for $\mathbf{X}$ will satisfy

$$\frac{h(\mathbf{X} \to \mathbf{X}')}{h(\mathbf{X}' \to \mathbf{X})} = \frac{H(\mathbf{X}')}{H(\mathbf{X})} \tag{3.4}$$

where $H(\mathbf{X}) = \Pi_{x \in \mathbf{X}} \eta(x)$ and $\eta(x)$ is the joint probability density of all the m random variables which are composed to form F_D. Note that the product is over the n sample elements which are contained in the sample set $\mathbf{X} \in \mathcal{X}$. It should be clear that each sample element, $x \in \mathbf{X}$, will itself be m dimensional.

Proposition 3.3.1. *The relative probability density of the resulting Markov chain in the space of ($\mathcal{D}$,$\mathcal{X}$) will hence be*

$$\pi(D, \mathbf{X}) = H(\mathbf{X}) e^{\beta \hat{V}_D(k, \mathbf{X})}.$$

for a fixed choice of β, n and k.

Proof. [Roberts and Rosenthal, 2004] state that given a factorisation of the proposal densities (equations 3.2, 3.3, 3.4) and acceptance probabilities (equation 3.1) the relative probability density of the Markov chain is as stated in the proposition, provided that the detailed balance condition can be proved. That is, we are required to show that

$$\pi(D, \mathbf{X}) q(D, \mathbf{X} \to D', \mathbf{X}') = \pi(D', \mathbf{X}') q(D', \mathbf{X}' \to D, \mathbf{X}) \tag{3.5}$$

where $q(D, \mathbf{X} \to D', \mathbf{X}') = a(D, \mathbf{X} \to D', \mathbf{X}') h(D, \mathbf{X} \to D', \mathbf{X}')$. Starting from the left hand side and substituting equations 3.2, 3.3, 3.1 and 3.4 leads to

$$\begin{aligned}
&\pi(D, \mathbf{X}) q(D, \mathbf{X} \to D', \mathbf{X}') \\
&= \pi(D, \mathbf{X}) h(D, \mathbf{X} \to D', \mathbf{X}') a(D, \mathbf{X} \to D', \mathbf{X}') \\
&= \pi(D, \mathbf{X}) \min \left(h(D, \mathbf{X} \to D', \mathbf{X}'), h(D', \mathbf{X}' \to D, \mathbf{X}) \frac{H(\mathbf{X}')}{H(\mathbf{X})} e^{\beta \left(\hat{V}_{D'}(k, \mathbf{X}') - \hat{V}_D(k, \mathbf{X}) \right)} \right) \\
&= \min \left(\pi(D, \mathbf{X}) h(D, \mathbf{X} \to D', \mathbf{X}'), \pi(D', \mathbf{X}') h(D', \mathbf{X}' \to D, \mathbf{X}) \right)
\end{aligned}$$

which is symmetric under the interchange of $(D, \mathbf{X})$ and $(D', \mathbf{X}')$ and hence we have shown the required equality (equation 3.5). □

We can now use the inversion method to project the stationary density from $(\mathcal{D}, \mathcal{X})$ to $(\mathcal{D}, [0,1]^n)$. We can then further project the stationary density onto $(\mathcal{D}, [0,1])$. This allows for a more interpretable description of the relative probability density.

To do this, we first note that $\hat{V}_D(k, \mathbf{X})$ must equal $V_D(R_k)$ for corresponding choices of $R_k \in [0,1]$ for all possible choices of k. Here R_k is the kth ranked of n samples of the uniform distribution on the unit interval (by the inversion method).

Such an equivalence must exist by construction of $\hat{V}_D(k, \mathbf{X})$. Proof and further details can be found in [Devroye, 1986].

The projected relative density would then become

$$\pi(D, R_1 \ldots, R_n) = \zeta(R_1, \ldots, R_n) e^{\beta V_D(R_k)}$$

where $\zeta(R_1, \ldots, R_n)$ is the joint density function for all of the $R_k s$. If we can then find the relative probability density of R_k we can further project the density to

$$\pi(D, R_k) = \zeta_k(R_k) e^{\beta V_D(R_k)} \tag{3.6}$$

$$\pi(D) = \int_0^1 dR_k \zeta_k(R_k) e^{\beta V_D(R_k)} \tag{3.7}$$

The first statement is a result of integrating $\pi(D, R_1 \ldots, R_n)$ with respect to R_i $\forall i \neq k$. The second statement is the result of integrating $\pi(D, R_k)$ with respect to R_k. These integrals will always exist when $V_D(.)$ is upper bounded on $[0, 1]$.

The relative density of the R_ks can be easily calculated since these R_k relate to uniform draws from $[0, 1]$ by the inversion method. We can use the binomial distribution to calculate their joint a priori probabilities of occurrence. The probability density of R_k, labelled $\zeta_k(R_k)$, is then given by

$$\begin{aligned} & P\left(k-1 \text{ samples are less than } R_k\right) \\ \times & P\left(n-k \text{ samples are greater than } R_k\right) \\ \times & n \text{ (Possible choices for the } k\text{th sample)} \\ \times & \binom{n-1}{k-1} \text{ (Possible permutations of the remaining samples).} \end{aligned}$$

By substituting the correct binomial probabilities into the above calculation and simplifying, we find that

$$\zeta_k(R_k) = \binom{n}{k} k R_k^{k-1} (1 - R_k)^{n-k}. \tag{3.8}$$

We note that R_k has a Beta distribution with parameters $\alpha = k$ and $\beta = n - k + 1$.

Remark 3.3.2. In order to consider expectations over this chain we will need to assume that the normalisation constant for the above relative densities exist, i.e.

$$N(\beta, n, k) = \int_{\mathcal{D}} dD \int_0^1 dR_k \zeta_k(R_k) e^{\beta V_D(R_k)} < \infty. \tag{3.9}$$

Clearly existence of the normalisation constant is dependent on the choice of $V_D(.)$, which as previously discussed is unlikely to be known in closed form (if at all). However, if we further assume that $\mathcal{D}$ is bounded and that $V_D(.)$ is bounded in the region near R_T then it seems reasonable to expect that the integral will exist for a sufficiently large choice of n and equivalent choice of $k = \lfloor R_T n \rfloor$.

It is worth noting that the two factors in $\pi(D, R_k)$ above will attempt to push $\langle R_k \rangle$ in different directions. The $\zeta_k(R_k)$ term will seek to select values of R_k near to $\frac{k}{n}$, whereas the $e^{\beta V_D(R_k)}$ term will seek to bias the chain towards larger choices of R_k. This is because it will give more weight to overestimates of $V_D(R_T)$ than to underestimates of $V_D(R_T)$. The pressure each term exerts on the choices of R_k will be driven by β and n respectively. This tension will mean that in practice $\langle R_k \rangle$ will almost always exceed $\frac{k}{n}$.

In this section we have found the Markov chain weights for our optimisation process. In later sections we build on the above results to prove some of the key results contained in this chapter.

3.4 Calculating the Risk

In this section we analytically prove several key statistics, which we will use in later sections to accurately control our optimisation process. We use the Markov chain results from the previous section to relate observable statistics with unobservable, but more understandable, statistics. This gives us a way of calculating the level of risk we are actually optimising (on average) given a fixed choice of n and k. Using these results we can significantly reduce the number of samples required to optimise an extreme quantile compared to a more traditional approach. Figure 3.1 demonstrates this point. Accurately estimating the quantile at each step would require a huge number of samples of F_D, the exact number of which is generally unknown and, as discussed at the end of the last section, as β grows a larger number of samples will be required. If we can calculate $\langle R_k \rangle$ during the algorithm we can limit the number of samples used in each iteration to the absolute minimum required, saving both time and computational resources. Without such an estimator we would either be forced to use a much larger choice of n, or not use sufficient n and find that we actually optimised a much higher quantile.

We continue to use the notation that was set up in the previous section. These results are recapitulated and numerically tested in section 3.5.

Initially we find the relationship between $\zeta_k(R)$ and $\zeta_j(R)$. As a reminder, $\zeta_k(R_k)$ is the probability density that a sample of rank k out of n samples from

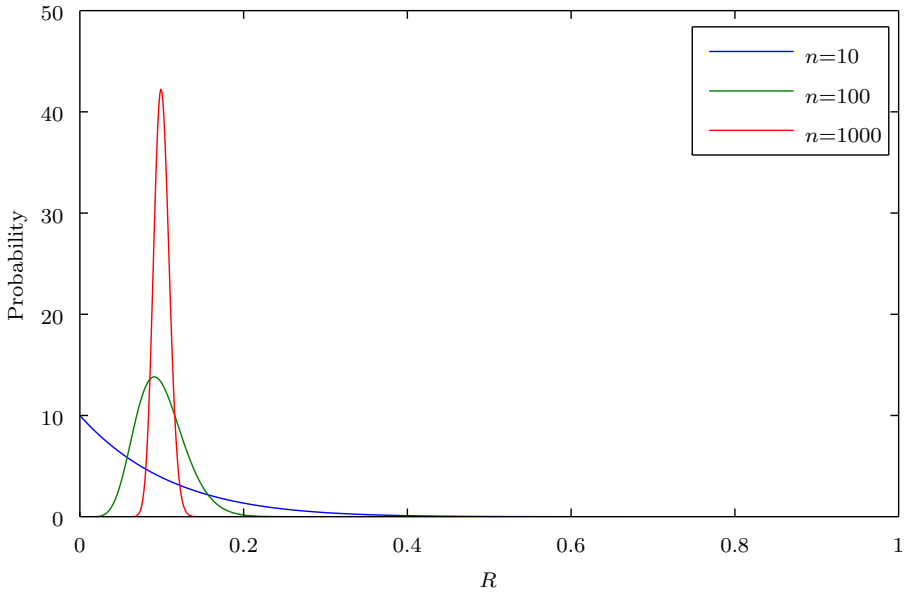


Figure 3.1: The figure above shows how $\zeta_k(R_k)$ changes as n increases, with $k = \lfloor 0.1n \rfloor$ in each case. We can see that as n increases not only does the dominant value rapidly approach 0.1 but also that the relative dominance of the dominant value increases. In traditional simulations one would be forced to use a (very) large choice of n to ensure that the optimisation optimised the correct quantile. We aim to be able to calculate the actual $\langle R_k \rangle$ used during the simulation so that we can fine tune the number of samples that are needed. This enables us to reduce the work required and ensure that we are using a sufficient number of samples for each iteration. An insufficient number of samples would potentially mean that we would actually optimise a higher quantile due to the upward pressure from the $e^{\beta V_D(R_k)}$ term in the weight.

the distribution of F_D corresponds to cumulative probability R. Without loss of generality we will assume that $k < j$ (for the sake of simplicity).

Proposition 3.4.1. *Given fixed k and j, with $k < j$, we find that*

$$\zeta_j(R) = \frac{(n-k)!(k-1)!}{(n-j)!(j-1)!} \left(\frac{R}{1-R} \right)^{j-k} \zeta_k(R).$$

Proof. The proof follows directly from the definition of $\zeta_k(R)$ and $\zeta_j(R)$.

□

We now calculate the relationship between several statistics of the Markov chain at equilibrium for fixed β and n. Notationally, we take $\langle . \rangle_k$ to be the arithmetic

mean over a Markov chain which is searched using point k in the n ordered samples.

The first result allows us to introspect the level of risk the Markov chain is sampling at without assuming any knowledge of $V_D(.)$ (as opposed to the cruder estimation of k/n which is not valid for small n).

Theorem 3.4.2 (Risk Estimation)**.** *Given fixed β, n, k and j we find that*

$$\left\langle e^{\beta(V_D(R_j)-V_D(R_k))}\right\rangle_k = \frac{(n-k)!(k-1)!}{(n-j)!(j-1)!}\left\langle R_j^{j-k}(1-R_k)^{k-j}\right\rangle_k$$

where $N(\beta,n,k)$ exists and the expectations are finite.

Proof. We first expand the definition of $\left\langle e^{\beta(V_D(R_j)-V_D(R_k))}\right\rangle_k$,

$$\left\langle e^{\beta(V_D(R_j)-V_D(R_k))}\right\rangle_k = \frac{\int_{\mathcal{D}} dD \int_0^1 dR_1 \ldots \int_0^1 dR_n e^{\beta(V_D(R_j)-V_D(R_k))}\pi(D,R_1,\ldots,R_n)}{N(\beta,n,k)}.$$

By expanding $\pi(D,R_1,\ldots,R_n)$ using conditional probability, integrating unused terms and substituting in the formula for $\pi(D,R_k)$ (from equation 3.7) we find

$$= \int_{\mathcal{D}} dD \int_0^1 dR_j \int_0^1 dR_k e^{\beta(V_D(R_j)-V_D(R_k))}\zeta_k(R_j|R_k)\frac{\zeta_k(R_k)e^{\beta V_D(R_k)}}{N(\beta,n,k)}$$

where $\zeta_k(R_j|R_k)$ is the relative density for the conditional distribution. Moving the normalisation constant outside the integrand and cancelling terms gives

$$= \frac{1}{N(\beta,n,k)}\int_{\mathcal{D}} dD \int_0^1 dR_j \int_0^1 dR_k e^{\beta V_D(R_j)}\zeta(R_j|R_k)\zeta_k(R_k).$$

Integrating out R_k and using the substitution for $\zeta_j(.)$ found in proposition 3.4.1 results in

$$= \frac{(n-k)!(k-1)!}{(n-j)!(j-1)!}\frac{1}{N(\beta,n,k)}\int_{\mathcal{D}} dD \int_0^1 dR_j R_j^{j-k}(1-R_j)^{k-j}\ e^{\beta V_D(R_j)}\zeta_k(R_j).$$

By observing that R_j is just a variable of integration, we can relabel it to R_k to get

$$= \frac{(n-k)!(k-1)!}{(n-j)!(j-1)!} \frac{1}{N(\beta,n,k)} \int_{\mathcal{D}} dD \int_0^1 dR_k R_k^{j-k} (1-R_k)^{k-j} \; e^{\beta V_D(R_k)} \zeta_k(R_k).$$

However, this is simply an expectation, $\left\langle R_k^{j-k}(1-R_k)^{k-j} \right\rangle_k$, multiplied by a constant, giving the result.

□

Example 3.4.3. In the case when $j = k+1$ we have that

$$\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))} \right\rangle_k = \frac{n-k}{k} \left\langle \frac{R_k}{1-R_k} \right\rangle_k . \tag{3.10}$$

We now have a method for identifying the level of risk the Markov chain is actually optimising, using only the kth and $(k+1)$th ranked sample of F_D. No knowledge of $V_D(.)$ or R_k is required. This crucial result allows us to greatly reduce the number of samples required during the optimisation process.

The second result in this section allows us to calculate an approximation for the derivative of $\frac{d\langle R_k \rangle}{dn}$ and make the above approximation more exact. Estimating this derivative will enable us to adjust the number of samples used to allow us to perform an optimisation at the desired level of risk, whilst using the minimal number of samples.

Theorem 3.4.4. *Given fixed β, n, k and a choice of p such that $0 \leq p \leq n$,*

$$\langle f(R_k) \rangle_{k,n-p} = \frac{\langle f(R_k)(1-R_k)^{-p} \rangle_{k,n}}{\left\langle \left(\frac{R_k}{1-R_k} + 1 \right)^p \right\rangle_{k,n}}$$

where $N(\beta,n,k)$ exists and the expectations are finite and where $\langle . \rangle_{k,n}$ is an average computed over a Markov chain optimising at point k and using n samples at each step.

Proof. Expanding $\langle f(r) \rangle_{k,n-p}$ according to its definition gives

$$\langle f(R_k)\rangle_{n-p} = \frac{1}{N(\beta,n,k)} \int_{\mathcal{D}} dD \int_0^1 dR_k f(R_k)\pi(D,R_k)\Big|_{n=n-p}$$
$$= \frac{\int_{\mathcal{D}} dD \int_0^1 dR_k f(R_k)\binom{n-p}{k} k R_k^{k-1}(1-R_k)^{n-p-k} e^{\beta V_D(R_k)}}{\int_{\mathcal{D}} dD \int_0^1 dR_k \binom{n-p}{k} k R_k^{k-1}(1-R_k)^{n-p-k} e^{\beta V_D(R_k)}}.$$

Multiplying both numerator and denominator by $\frac{\binom{n}{k}}{\binom{n-p}{k}}$ and separating out $(1-R_k)^{-p}$ results in

$$= \frac{\int_{\mathcal{D}} dD \int_0^1 dR_k f(R_k)(1-R_k)^{-p}\pi(D,R_k)}{\int_{\mathcal{D}} dD \int_0^1 dR_k (1-R_k)^{-p}\pi(D,R_k)}.$$

Dividing both the numerator and denominator by $N(\beta,n,k)$ gives

$$= \frac{\langle f(R_k)(1-R_k)^{-p}\rangle_{k,n}}{\langle (1-R_k)^{-p}\rangle_{k,n}}.$$

By substituting $\frac{1}{1-R_k} = \frac{R_k}{1-R_k} + 1$ we reach the final result

$$= \frac{\langle f(R_k)(1-R_k)^{-p}\rangle_{k,n}}{\left\langle \left(\frac{R_k}{1-R_k}+1\right)^p \right\rangle_{k,n}}.$$

□

Corollary 3.4.5. *Given fixed β, n, k, with k small, we have*

$$\left\langle \frac{R_k}{1-R_k} \right\rangle_{k,n} - \left\langle \frac{R_k}{1-R_k} \right\rangle_{k,n-1} = \frac{\left(\frac{k}{n-k}\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}\right)^2 - \frac{k(k+1)}{(n-k)(n-k-1)}\left\langle e^{\beta(V_D(R_{k+2})-V_D(R_k))}\right\rangle_{k,n}}{1+\frac{k}{n-k}\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}}$$

where $N(\beta, n, k)$ exists and the expectations are finite.

Proof. We expand the left hand side by applying Theorem 3.4.4 to $f(R_k) = \frac{R_k}{1-R_k}$, with $p = 1$, to get

$$\begin{aligned}\left\langle \frac{R_k}{1-R_k} \right\rangle_{k,n-1} &= \frac{\left\langle \frac{R_k}{1-R_k}(1-R_k)^{-1}\right\rangle_{k,n}}{\sum_{i=0}^{1}\binom{1}{i}\left\langle \left(\frac{R_k}{1-R_k}\right)^i\right\rangle_{k,n}} \\ &= \frac{\left\langle \frac{R_k}{(1-R_k)^2}\right\rangle_{k,n}}{1+\left\langle \frac{R_k}{1-R_k}\right\rangle_{k,n}}.\end{aligned}$$

By substituting $\frac{1}{1-R_k} = \frac{R_k}{1-R_k} + 1$ into the numerator we get

$$= \frac{\left\langle \left(\frac{R_k}{1-R_k}\right)^2 + \frac{R_k}{1-R_k}\right\rangle_{k,n}}{1+\left\langle \frac{R_k}{1-R_k}\right\rangle_{k,n}}.$$

We are now able to expand $\left\langle \frac{R_k}{1-R_k} \right\rangle_{k,n} - \left\langle \frac{R_k}{1-R_k} \right\rangle_{k,n-1}$ by substituting the above expansion. This results in

$$\begin{aligned}\left\langle \frac{R_k}{1-R_k} \right\rangle_{k,n} - \left\langle \frac{R_k}{1-R_k} \right\rangle_{k,n-1} &= \left\langle \frac{R_k}{1-R_k} \right\rangle_{k,n} - \frac{\left\langle \left(\frac{R_k}{1-R_k}\right)^2 + \frac{R_k}{1-R_k}\right\rangle_{k,n}}{1+\left\langle \frac{R_k}{1-R_k}\right\rangle_{k,n}} \\ &= \frac{\left(\left\langle \frac{R_k}{1-R_k}\right\rangle_{k,n}\right)^2 - \left\langle \left(\frac{R_k}{1-R_k}\right)^2\right\rangle_{k,n}}{1+\left\langle \frac{R_k}{1-R_k}\right\rangle_{k,n}}.\end{aligned}$$

By applying Theorem 3.4.2 we get an (observable) expression in terms of exponential averages, given by

$$= \frac{\left(\frac{k}{n-k}\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}\right)^2 - \frac{k(k+1)}{(n-k)(n-k-1)}\left\langle e^{\beta(V_D(R_{k+2})-V_D(R_k))}\right\rangle_{k,n}}{1+\frac{k}{n-k}\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}}.$$

□

Furthermore, we can approximate $\frac{d\langle R_k\rangle}{dn}$ using finite differences. If we assume small k and large n we have that

$$\frac{d\langle R_k\rangle}{dn} \approx \langle R_k\rangle_{k,n} - \langle R_k\rangle_{k,n-1} \approx \left\langle \frac{R_k}{1-R_k}\right\rangle_{k,n} - \left\langle \frac{R_k}{1-R_k}\right\rangle_{k,n-1}. \tag{3.11}$$

Note that this becomes equality to leading (first) order in R in the limit of large n (as $n \to \infty$). Then by corollary 3.4.5 we have

$$\frac{d\langle R_k\rangle}{dn} \approx \frac{\left(\frac{k}{n-k}\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}\right)^2 - \frac{k(k+1)}{(n-k)(n-k-1)}\left\langle e^{\beta(V_D(R_{k+2})-V_D(R_k))}\right\rangle_{k,n}}{1+\frac{k}{n-k}\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}} \tag{3.12}$$

assuming $N(\beta, n, k)$ exists and the expectations are finite.

It is worth noting that by using similar methods to those above we can actually find a more accurate analytic approximation for $\frac{d\langle R_k\rangle}{dn}$. However, the resulting expression is much more complex and hence is unlikely to converge quickly. This would make it unsuitable for use in an efficient algorithm. These expressions could be found by following a similar logic to that of corollary 3.4.6.

Finally, we note that in theorem 3.4.2 we found an observable estimator for $\left\langle \frac{R_k}{1-R_k}\right\rangle_{k,n}$, however ideally we wish to estimate $\langle R_k\rangle_{k,n}$ as it is a much more clearly interpretable quantity. When R_k is small we would expect both expectations to be approximately equal, however this does not hold for larger R_k. In the following corollary we find an upper bound on $\langle R_k\rangle_{k,n}$ for all R_k, based on $\left\langle \frac{R_k}{1-R_k}\right\rangle_{k,n}$.

Corollary 3.4.6 (Bounding the Risk)**.** *Given fixed β, n, k*

$$\langle R_k\rangle_{k,n} \leq \frac{\frac{k}{n-k}\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}}{1+\frac{k}{n-k}\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}}$$

when $N(\beta, n, k)$ exists and the expectations are finite.

Proof. By applying Theorem 3.4.4 to $f(R_k) = R_k$ with $p = 1$ we get

$$\langle R_k \rangle_{k,n-1} = \frac{\left\langle \frac{R_k}{1-R_k} \right\rangle_{k,n}}{1 + \left\langle \frac{R_k}{1-R_k} \right\rangle_{k,n}}.$$

By applying Theorem 3.4.2 we get an (observable) expression in terms of exponential averages, given by

$$= \frac{\frac{k}{n-k} \left\langle e^{\beta(V_D(R_{k+1}) - V_D(R_k))} \right\rangle_{k,n}}{1 + \frac{k}{n-k} \left\langle e^{\beta(V_D(R_{k+1}) - V_D(R_k))} \right\rangle_{k,n}}.$$

Finally we note that $\langle R_k \rangle_{k,n} \leq \langle R_k \rangle_{k,n-1}$ and thus obtain the final result.

□

The results found in this section give us a powerful set of tools that can be used when designing an optimisation algorithm in later sections (3.6). Critically, since we know the level of risk any given point in our ordered samples is operating at, we can reduce the size of the sample set required in two ways. Firstly, we no longer need to take the (standard) brute force approach of using a fixed size, as we can dynamically calculate the sufficient size of our sample set to optimise at the correct level of risk. Secondly, we can move away from estimating a level of risk, R_T, by the $(R_T n)$ ranked point in our sample set to using, for example, the worst point in the sample set. This again greatly reduces the samples required and is underpinned by the above results which allow us to identify the level of risk of a specific point.

3.5 Numerical Convergence of Estimators

In this section we investigate the numerical behaviour of the estimators found in the previous section (3.4). We know that the estimators must eventually converge over a long time assuming a given system is at equilibrium. However, we wish to investigate the estimator's responsiveness when these assumptions may not be the case.

Our primary purpose in this section is to determine how long the estimators might take to converge in different systems. We will also look for any other observations that could be used to help develop a suitable algorithm based on these results.

In this section we focus on the estimators for

$$\langle R_k \rangle_{k,n} \leq \frac{\frac{k}{n-k}\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}}{1+\frac{k}{n-k}\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}} \text{ (see 3.4.6) and}$$

$$\frac{d\langle R_k\rangle}{dn} \approx \frac{\left(\frac{k}{n-k}\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}\right)^2}{1+\frac{k}{n-k}\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}} - \frac{\frac{k(k+1)}{(n-k)(n-k-1)}\left\langle e^{\beta(V_D(R_{k+2})-V_D(R_k))}\right\rangle_{k,n}}{1+\frac{k}{n-k}\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}} \text{ (as per 3.12).}$$

We evaluate the estimators using a simple, one dimensional, toy model. We choose $V_D(R)$ to be the inverse cumulative distribution function of a standard normal distribution. In this example there is only one possible design. Mathematically, we explore the system which has relative density $\zeta_k(R_k)e^{\beta\Phi^{-1}(R_k)}$. The design element does not appear in these weights as we have fixed it to be the standard normal distribution (so there is only one possible design). In this example, F_D is equal to a random variable which has a standard normal distribution.

We use the methodology as described in section 3.3 to sample the Markov chain. We attempt to replace all the samples contained in $\mathbf{X}$ at each step. Clearly, the design, D, is not perturbed as there is only one possible choice.

As the mathematics exposes, we expect our chain to have two tensions underpinning it. The first is that the randomly drawn samples from the normal distribution select in favour of $\langle R_k \rangle$ close to $\frac{k}{n}$. The second is that $e^{\beta V_D(R_k)} = e^{\beta\Phi^{-1}(R_k)}$ will seek to push $\langle R_k \rangle$ up. This term will bias our chain towards sets of "lucky" samples of X which have an abnormally high kth member when ordered. As we increase n we would expect $\langle R_k \rangle$ to move closer to $\frac{k}{n}$. Conversely, as we increase β we would expect the second factor to have an increased effect and hence would expect to see $\langle R_k \rangle >> \frac{k}{n}$.

Since we know $V_D(R_k)$ in this example we can calculate the true value of $\langle R_k \rangle$ a given chain explores. We compare this exact estimate with our estimator for $\langle R_k \rangle$ from the previous section.

Figure 3.2 shows the convergence of our estimator for a variety of choices of β and n. The graphs show the value of the estimator (against the true $\langle R_k \rangle$) for a chain that has a certain number of accepted moves since starting from a random condition. The second half of each set of accepted moves was used to calculate the estimator.

All of the graphs in figure 3.2 demonstrate good convergence of the estimator (for $\langle R_k \rangle$) over most parameter settings β. However as β increases the estimator

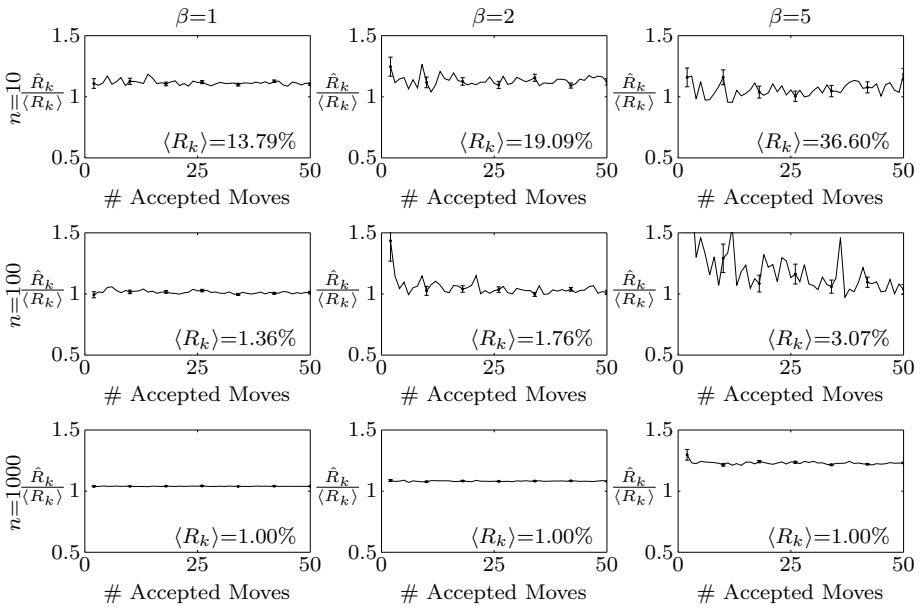


Figure 3.2: The above charts show the convergence of the estimator for $\langle R_k \rangle$ as defined in corollary 3.4.6. $\hat{R}_k$ denotes this estimator. $\langle R_k \rangle$ denotes the true expected value of R_k as calculated explicitly using the known $V_D^{-1}(V) = \Phi(V)$. k was set to $\max(1, 0.01n)$ (targeting 1% risk when $n \geq 100$). The estimator shows excellent convergence for all choices of n and β. The convergence is slower for larger choices of β when coupled with small n, however we should not be operating in this regime as it will also have high expected R_k. The graphs also show that the estimator is indeed an upper bound and is therefore a reliable measure to ensure our chain is operating at the target risk level R_T%.

performs less well. As expected, small n and large β suffer the most. It is worth noting that when $\beta = 5$, $\langle R_k \rangle$ has been pushed up significantly for $n = 10$ and $n = 100$. In these cases $\langle R_k \rangle = 36.6\%$ and $\langle R_k \rangle = 3.1\%$ respectively. In both cases this is approximately three times larger than the level of $\langle R_k \rangle \left(= \frac{k}{n}\right)$ one might expect. We also note that, crucially, our estimator always over estimates the true value of $\langle R_k \rangle$. This means it can be used as a reliable safety check to ensure we have enough samples to search at the target level of risk R_T.

We also note that our estimator for $\langle R_k \rangle$ (as defined in corollary 3.4.6) appears to converge more quickly than when using a known $V_D^{-1}(V) = \Phi(V)$ to calculate the expected risk level. Figure 3.3 demonstrates this for the case when $n = 100$ and $\beta = 1$.

Figure 3.4 shows the result of the estimator for $\frac{d\langle R_k \rangle}{dn}$ versus a pointwise

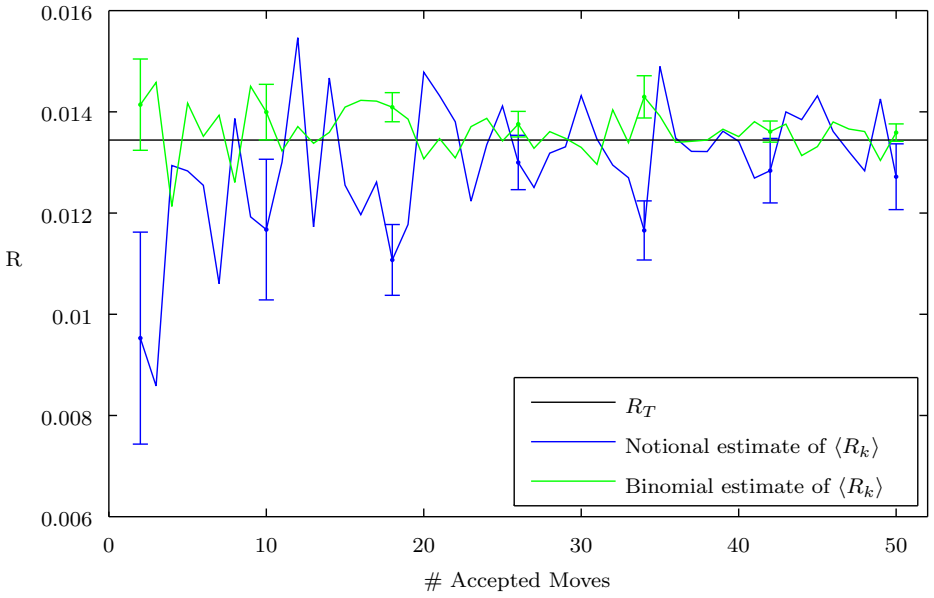


Figure 3.3: The above figure shows the convergence of the estimator for $\langle R_k \rangle$ for a specific choice of $n = 100$ and $\beta = 1$. The notional estimate of $\langle R_k \rangle$ was calculated using $V_D(R_k)$ (which was known in this example). The binomial estimate was calculated as shown in corollary 3.4.6. We see improved convergence using our estimator over the raw calculation.

estimate of $\frac{d\langle R_k \rangle}{dn}$ using the known $V_D(R_k)$. We note that the value of $\frac{\hat{R}'_k}{R'_k} = \left(\frac{\frac{d\langle \hat{R}_k \rangle}{dn}}{\frac{d\langle R_k \rangle}{dn}} \right)$ generally requires more samples to achieve the same relative change in $\langle R_k \rangle$ as β increases (for fixed n). This is to be expected, since as β increases more samples will be required to hold $\langle R_k \rangle$ down and therefore a greater change in n will be required to reduce $\langle R_k \rangle$ by the same (relative) amount. As before the estimator begins to suffer when β is high and n is small, however this is not a regime we intend to operate in.

The above results show that the estimators found in the previous section perform very well and should provide a good building block for our algorithm. In the next section we use the intuition developed both here and in previous sections to design an improved annealing algorithm for our class of problem. We then numerically test this algorithm in the following sections.

Finally, we note that, although the results are not included for the sake of brevity, the convergence for other forms for V_D was tested and found to have similar results as the standard normal case shown above.

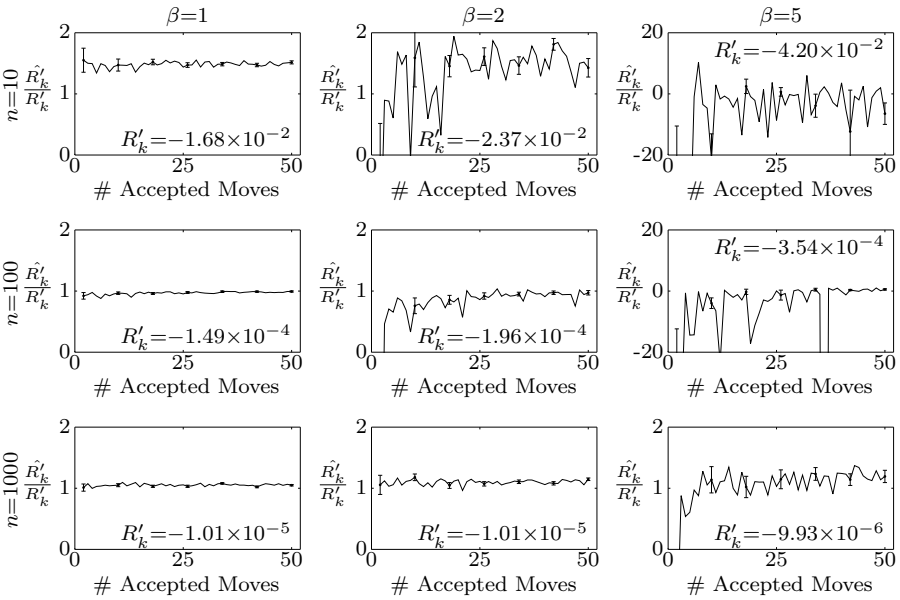


Figure 3.4: The above figures show the convergence of the estimator for $\frac{d\langle R_k\rangle}{dn}$ (as defined in equation 3.12). $\hat{R}'_k$ denotes this estimator. R'_k denotes the true $\frac{d\langle R_k\rangle}{dn}$ which was pointwise estimated using the known $V_D^{-1}(V) = \Phi(V)$ to calculate $\langle R_k\rangle$ at nearby n. As before, k was set to $\max(1, 0.01n)$. The results generally show that the estimator convergences quickly to the correct value. When n is fixed we see that $\frac{d\langle R_k\rangle}{dn}$ increases with increasing with β (at least for small n). This at first seems counter intuitive as we expect it to be harder to push $\langle R_k\rangle$ down as β increases. However, we need to remember that $\langle R_k\rangle$ has also significantly increased with β and hence to decrease $\langle R_k\rangle$ by the same relative amount would require significantly more samples for higher β. This supports our initial argument that for high β we need to fight harder (and perform more sampling) to keep $\langle R_k\rangle$ low and that to obtain the same relative reduction of $\langle R_k\rangle$ requires significantly more samples (as compared to a smaller choice of β). The estimator demonstrates some degradation at low n and high β.

3.6 Algorithm

In this section we build on the intuition built in the previous two sections (3.4 and 3.5) to improve the basic annealing algorithm to allow it to more quickly solve the style of problem discussed in section 3.2. We focus on how we can use the previous results to extend the algorithm. As discussed in the section 2.6.4, the choice of cooling schedule for stochastic annealing is problematic. A wide variety of different methodologies are discussed in the literature [Kolonko, 1999; Bertsimas and Tsitsiklis, 1993; Thompson and Dowsland, 1998], however we stick to a very simple schedule of linearly increasing β every N accepted design changes and instead focus our discussion on optimising the new features we have introduced. All of the annealing algorithms used in this thesis are summarised in appendix A.

In order to define the algorithm we assume we have a predefined cooling schedule which is defined by a sequence of inverse temperatures, $\beta_1, \ldots, \beta_m$, at points $N_1, \ldots, N_m$. As a reminder of notation (from sections 3.2 and 3.3); F_D is a random variable which is parametrised by a choice of $D \in \mathcal{D}$. F_D is composed of m underlying random variables whose individual distributions are independent of D. For example, the underlying random variables could encapsulate possible future asset prices and D the design of a portfolio which contains these assets. $\hat{V}_D(k, \mathbf{X})$ is the estimate of $V_D(R_T)$ based on the kth smallest sample of the resultant distribution of F_D. $\mathbf{X}$ is a samples set where each individual sample (element) is independently drawn from the distributions of the underlying random variables of F_D. Label X as the joint distribution of the distributions of the underlying random variables which are composed to form F_D.

Finally, we define $|\mathbf{X}|$ to be the cardinality (size) of the set $\mathbf{X}$, $\lceil y \rceil$ to represent the smallest integer greater than y and $\lfloor y \rfloor$ to represent the largest integer smaller than y.

We initialise the algorithms with an initial condition for D (D_0) and provide some method for drawing samples from the distributions of the underlying random variables. We wish to optimise the design of the random variable F_D at it's R_T percentile.

Algorithm 3.1 defines a simple brute force version of the proposed algorithm. This algorithm shows how the cooling scheme defined above is used. The algorithm is a standard simulated annealing algorithm and essentially assumes that the estimate of $V_D(R_T)$ is correct and no biasing of the quantile from the $e^{\beta V_D(R_k)}$ term occurs. Each iteration the algorithm draws $n = \left\lfloor R_T^{-2} \right\rfloor$ samples from X, to form a sample set, and then uses the elements of this sample set to estimate $V_{\tilde{D}}(R)$, where $\tilde{D}$ is a

perturbation of D. The new design choice is then accepted with a probability given by the Metropolis acceptance function. This function uses the estimate of $V_{\tilde{D}}(R)$ and the current estimate of $V_D(R_T)$. ($V_D(R_T)$ is not re-estimated each step.) The system is cooled used the cooling schedule defined above.

Algorithm 3.1 Brute Force

1: $D := D_0$, $n := \left\lceil \left(\frac{1}{R_T}\right)^2 \right\rceil$ and $k := \max(1, \lfloor nR_T \rfloor)$
2: $\mathbf{X} :=$ n samples drawn from X
3: $V := \hat{V}_D(k, \mathbf{X})$
4: **for** $1 \leq i \leq m$ **do** ▷ For each cooling step
5: $\quad j := 0$
6: $\quad$ **while** $j \leq N_i$ **do** ▷ For each iteration
7: $\quad\quad \tilde{D} :=$ a perturbation of D
8: $\quad\quad \tilde{\mathbf{X}} :=$ n (new) samples from X
9: $\quad\quad \tilde{V} := \hat{V}_{\tilde{D}}(k, \tilde{\mathbf{X}})$
10: $\quad\quad p := \min(1, e^{-\beta_i(V - \tilde{V})})$ ▷ Metropolis Acceptance Function
11: $\quad\quad$ **if** $p \geq u$ where $u \sim U(0,1)$ **then**
12: $\quad\quad\quad V := \tilde{V}$, $D := \tilde{D}$ and $X := \tilde{\mathbf{X}}$
13: $\quad\quad\quad j := j + 1$
14: Result is D

Algorithm 3.2 defines the algorithm when we wish to change n with β. $c(n)$ is a (functional) parameter which governs how much we perturb the random samples used to estimate $V_D(R_T)$ per attempted perturbation of D. In this algorithm we use the estimators for $\langle R_k \rangle$ and $\frac{d\langle R_k \rangle}{dn}$, as found in the preceding sections, to carefully tune the number of samples, and hence portfolio evaluations, required at each step. This should allow us to gain a significant time saving over the traditional method in which a fixed number of samples would be used at each step to estimate $V_D(R_T)$. This would then be considered an accurate estimate and a standard annealing method could be applied. In our results in the next section we show that this assumption is not always correct.

In this algorithm we use the same $\mathbf{X}$ to evaluate both the original and perturbed design and we then alternate between perturbing D and perturbing $\mathbf{X}$ each iteration. The reason we do this, instead of perturbing them together, is to improve acceptance rates. The disadvantage of using a smaller sample size to estimate $V_D(R_T)$ is that we are more vulnerable to lucky sampling of X giving a large overestimate of $V_D(R_T)$. This would mean although the system is in equilibrium, frequent lucky sets of samples of X could reduce the rate of movement through the system. (In other words, it will take much longer to fairly sample the equilibrium system

Algorithm 3.2 Simulated Annealing on n

$D := D_0$, $n := 100$ and $k := \max(1, \lfloor nR_T \rfloor)$
$\mathbf{X} :=$ n samples drawn from X
$V := \hat{V}_D(k, \mathbf{X})$
for $1 \leq i \leq m$ **do** ▷ For each cooling step
  for $1 \leq b \leq 2$ **do** ▷ Split the iterations in two
    $j := 0$
    while $j \leq N_i/2$ **do** ▷ For half of the iterations
      $\tilde{D} :=$ a perturbation of D
      $\tilde{V} := \hat{V}_{\tilde{D}}(k, \mathbf{X})$
      $p := \min(1, e^{-\beta_i(V-\tilde{V})})$ ▷ Metropolis Acceptance Function
      if $p \geq u$ where $u \sim U(0,1)$ **then**
        $V := \tilde{V}$ and $D := \tilde{D}$
        $j := j + 1$
      $A := \frac{k}{n-k} e^{\beta(V_D(R_{k+1}) - V_D(R_k))}$
      $B := \frac{k(k+1)}{(n-k)(n-k-1)} e^{\beta(V_D(R_{k+2}) - V_D(R_k))}$
      $\hat{R} := \frac{1}{j}\left((j-1)\hat{R} + \frac{A}{1+A}\right)$ ▷ Calculate the estimator for $\langle R_k \rangle$
      $\hat{dR} := \frac{1}{j}\left((j-1)\hat{dR} + \frac{A^2-B}{1+A}\right)$ ▷ Calculate the estimator for $\frac{d\langle R_k \rangle}{dn}$
      for $1 \leq l \leq c(n)$ **do** ▷ Perturb $\mathbf{X}$ $c(n)$ times
        $\tilde{\mathbf{X}} := \mathbf{X}$
        Replace one element of $\tilde{\mathbf{X}}$ with a new random sample from X
        $\tilde{V} := \hat{V}_{\tilde{D}}(k, \mathbf{X})$
        $p := \min(1, e^{-\beta_i(V-\tilde{V})})$ ▷ Metropolis Acceptance Function
        if $p \geq u$ where $u \sim U(0,1)$ **then**
          $V := \tilde{V}$ and $\mathbf{X} := \tilde{\mathbf{X}}$
    if $\hat{R} > R_T$ **then** ▷ Check the level of risk we have been optimising
      $n := \min\left(2000, n + \left\lceil \frac{R_T - \hat{R}}{\hat{dR}} \right\rceil\right)$ ▷ Increase n to push $\langle R_k \rangle$ down
      $k := \max(1, \lfloor nR_T \rfloor)$
      while $|X| < n$ **do** ▷ Add new random samples to $\mathbf{X}$
        Draw a random sample from X and add it to $\mathbf{X}$
      $V := \hat{V}_D(k, \mathbf{X})$.
    $\hat{R} := 0$ ▷ Reset estimators for $\langle R_k \rangle$ and $\frac{d\langle R_k \rangle}{dn}$
    $\hat{dR} := 0$
Result is D

than if we encourage quicker acceptance rates by perturbing $\mathbf{X}$ and D separately.) We perturb $\mathbf{X}$ by considering 100 independent swaps of a randomly selected element of the current sample set with a new sample drawn from X. We perturb each sample in $\mathbf{X}$ independently as it does not increase the number of portfolio evaluations required, but should improve the acceptance rate for these perturbations.

We check our estimators for $\langle R_k \rangle$ when we change the temperature and once in between changing temperatures. We check if $\hat{R}_k > R_T$ and if so we increase n. How we increase n depends on the variant of the algorithm used (see below).

We start with a sample set containing a hundred members as it easily allows us to provide initial risk estimates for target risk levels as low as 1%. Furthermore, as discussed in the previous section, using only a small sample set size is likely to increase the movement of the system as it does not allow for a good estimate of $V_D(R_T)$. As n increases we rescale k to be $\lfloor nR_T \rfloor$ (see line 27). This suggests that we should further increase n as the estimator for $\frac{d\langle R_k \rangle}{dn}$ is for fixed k, however we anticipate this change being minor and therefore expect limited impact from each change in n. The algorithm will self correct over the course of the chain and so any error created by changing k should have limited impact.

We wish to attempt to ensure that each time we perturb $\mathbf{X}$ we have some impact on $V_D(R_k, \mathbf{X})$. For this to be the case (on average) we need to perturb at least $\frac{n}{k}$ elements of the sample set. We take $c(n) = 100$ because for small n (when β is low) we would expect $c(n) > \frac{n}{k}$. However, once n grows large $c(n)$ may be less than $\frac{n}{k}$. This is not a problem as at this point β will be high and the move acceptance rate for D greatly reduced. We would therefore still expect that $V_D(R_k, \mathbf{X})$ would change between accepted design perturbations. We could set $c(n) = n$ but we wish to limit the amount of work the algorithm requires and a higher number of perturbations to $\mathbf{X}$ should not be needed.

Based on the numerical results in section 3.5 the cooling intervals, N_i, should be at least a hundred steps to ensure the estimators used have time to converge.

Aside from the comments above the algorithm should be a natural extension of the Brute Force algorithm (3.1). Clearly, the annealing schedule could be adjusted to work on another metric. For example, the number of accepted moves, as used in the previous section. A simulated tempering algorithm could also be used in place of simulated annealing. Furthermore, the acceptance function could be replaced with a different Metropolis-Hastings acceptance function.

We note that the algorithm as defined above is actually a type of single component Metropolis Hastings algorithm, rather than the simple Metropolis Hastings algorithm discussed in earlier sections (which used a single update step). The chain

will converge to the same stationary distribution and an analogous proof is possible [Gilks et al., 1996].

As previously discussed, if we fixed n, this algorithm would be a deterministic simulated annealing algorithm in the space of $(\mathcal{D}, [0,1])$ and hence with a suitable cooling schedule should find the global maximum of $\zeta_k(R_k)e^{\beta V_D(R_k)}$ [Henderson et al., 2003]. For a choice of $V_D(.)$ that is bounded on $[0,1]$ and continuous in the region around R_T we would further expect that for a sufficiently large choice of n the value of D that maximised $\zeta_k(R_k)e^{\beta V_D(R_k)}$ would also maximise $e^{\beta V_D(R_T)}$ and hence maximise $V_D(R_T)$. It is then plausible that one could prove that, under the conditions above, a version of the above algorithm that used a logarithmic cooling schedule and incremented n in some particular way, would converge (with probability 1) to $\max_D(V_D(R_T))$. However, since we do not intend to cool the temperature logarithmically and our current method of increasing n is not analytically dependant on $V_D(.)$ it is highly unlikely we could prove the algorithm (as is) will converge to a global maximum of $V_D(R_T)$. That said, by virtue of the above argument, we would reasonably expect (as a heuristic) that algorithm 3.2 will find, at worst, a (relatively) good local maximum of $V_D(R_T)$.

In the next section we will apply the above algorithm to a test problem from finance before concluding the chapter in section 3.8.

3.7 Results

In this section we apply the algorithm outlined in the previous section (3.6) to a simple portfolio selection problem from finance.

In order to study the algorithm several variants were compared. These are outlined below. As before R_T is the target risk level and n is the number of samples used at each step to estimate $V_D(R_T)$. The algorithms are:

BF A standard brute force algorithm using $n = \lfloor R_T^{-2} \rfloor$ samples to estimate $V_D(R)$. This is the algorithm as described in 3.1.

SBF A standard brute force algorithm using $n = \lfloor R_T^{-1} \rfloor$ samples to estimate $V_D(R)$. This is the algorithm as described in 3.1, except on line 1 set $n \leftarrow \left\lceil \frac{1}{R_T} \right\rceil$.

SA1 In this algorithm n is increased by a fixed amount every N_i accepted D moves, irrespective of the estimated value of $\langle R_k \rangle$. This is as algorithm 3.2, except on line 28 set the condition to always true and on line 26 set $n \leftarrow n + 50$.

SA2 In this algorithm we check if $\hat{R}_k > R_T$ every $\frac{N_i}{2}$ accepted D moves. If $\hat{R}_k$ is larger than the target value (R_T) then we increase n by a fixed amount. This is as algorithm 3.2, except on line 26 set $n \leftarrow n + 50$.

SA3 In this algorithm we check if $\hat{R}_k > R_T$ every $\frac{N_i}{2}$ accepted D moves. If $\hat{R}_k$ is larger than the target value (R_T) then we increase n by an amount dictated by $\frac{d\langle \hat{R}_k \rangle}{dn}$. This is algorithm 3.2.

All of the annealing algorithms used in this thesis are summarised in appendix A.

SBF and BF were both considered so that the effect of the bias, induced by the $e^{\beta V_D(R_k)}$ term, could be considered for different fixed sample sizes. This comparison should allow us to highlight the difficulty in selecting an appropriate sample size. The three different SA algorithms will enable us to compare the impact of each of the estimators on the number of portfolio evaluations required and the final result. Clearly, if all else is equal, SA3 is the most useful algorithm as it requires the least prior knowledge.

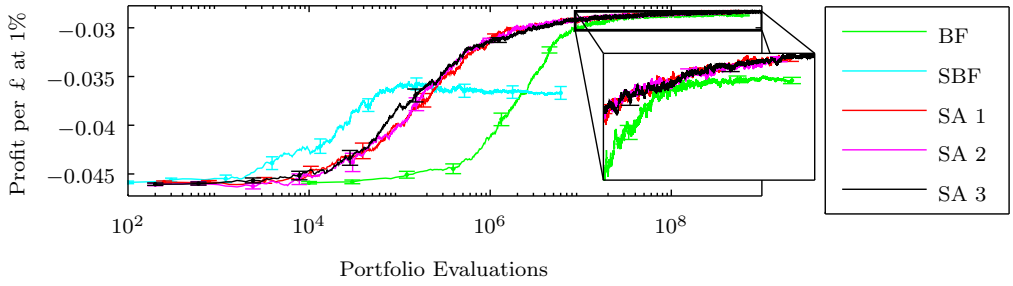
The same annealing schedule we used across all the algorithms. The algorithms started at $\beta = 250$ and increased beta by 250 every hundred accepted perturbations of D. The algorithms were allowed a maximum of $60,000$ attempted moves in D.

D was perturbed consistently across the algorithms. Each time D was perturbed each element was changed by a uniform random number from the region $[-0.025, 0.025]$, subject to the constraints on each dimensions. The perturbation to D was then renormalised and a further check made to ensure the individual constraints held, if they did not the process was restarted.

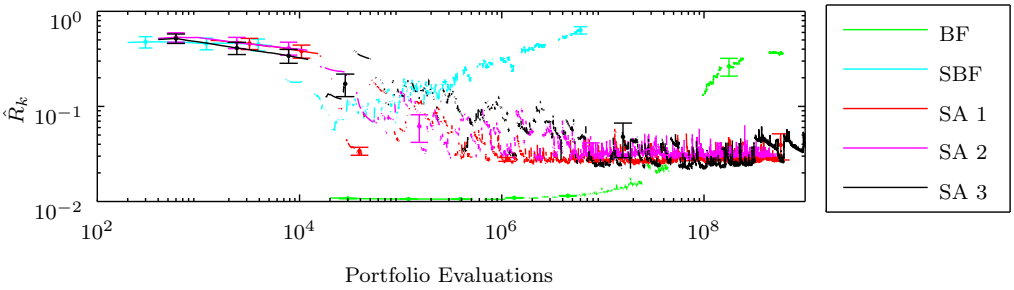
A portfolio of ten stocks was optimised at the R_T percentile. The algorithm was tested for R_T equal to 1%, 5% and 10% (equivalent to the 99%, 95% and 90% Value at Risk respectively).

We later discuss the specifics of how the individual stocks were modelled and the interpretation of the design space. Mathematically, the design space was taken to be $\mathcal{D} = \left\{ D \in [-5, 5]^{10} : \sum_{d_i \in D} = 1 \right\}$.

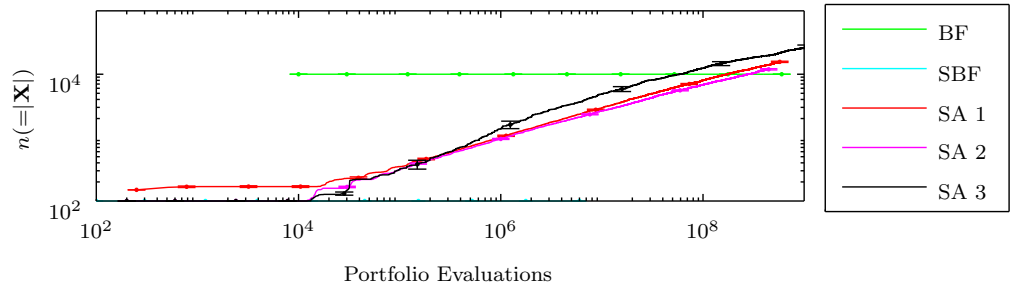
The code was developed in MATLAB and then run on a grid computing platform at the University of Warwick. Each algorithm was run 48 times for each choice of R_T to find both the expected solution and standard error.



(a) The above figure shows the convergence of the different algorithms. We can see that the new SA algorithms all find a better solution than BF. Furthermore, the SA algorithms are always at a better solution given a fixed number of portfolio evaluations (although it is close around $10^{7.2}$ evaluations). Finally, we note that the SA algorithms are able to get within 10% of the final brute force solution with half a magnitude less samples than BF. This is a saving of $10^{6.5}$ portfolio evaluations. SBF generally does very poorly as expected. Note the true value of $V_D(R_T)$ is plotted rather than the value estimated during the simulation

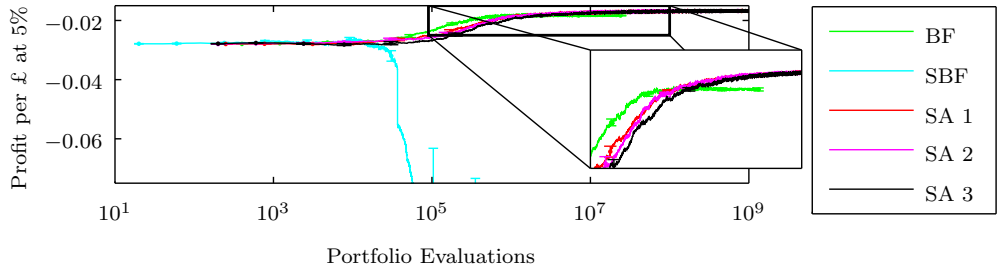


(b) In the figure above we see how the value of $\hat{R}_k$ at which the algorithms are operating changes as the anneal progresses. SBF never gets close to 1% and is driven even higher as the algorithm progresses. The SA algorithms all start high, but quickly adjust to a low value of $\hat{R}_k$ and remain at that value. BF starts low, but struggles to maintain a low level of $\hat{R}_k$ towards the end of the anneal. The SA algorithms do not appear to quite reach 1%. This is likely due to the fact we increase k (and sometimes β) at the same time as increasing n meaning the system can never quite get n high enough.

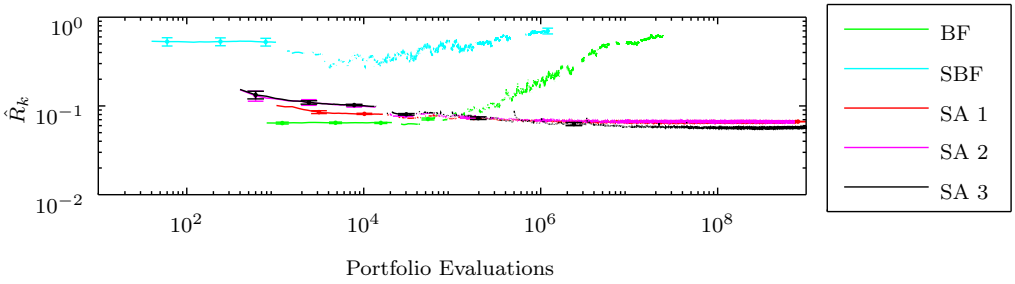


(c) In this figure we can see the number of samples each algorithm used to estimate $V_D(R_T)$ as it progressed. SBF and BF remain fixed at their initial values and SA rises steadily (to above 10^4 samples).

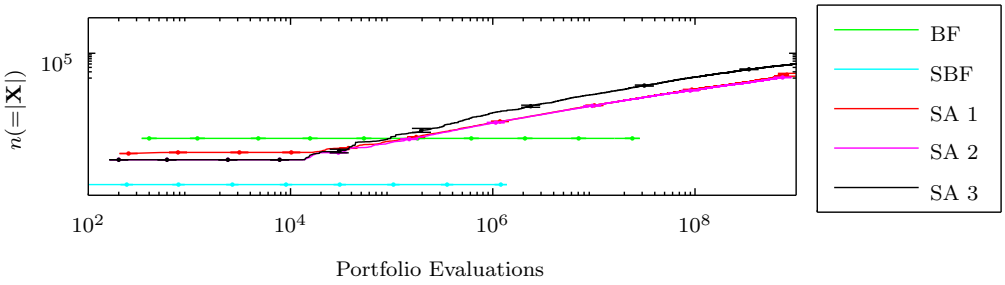
Figure 3.5: The algorithms have all optimised a portfolio at 1%, or equivalently, the 99% Value at Risk.



(a) The above figure shows the convergence of the different algorithms. We can see that the SA algorithms all find a significantly improved solution when compared to BF. However, in this case the SA algorithms are not always at a better solution for any choice of portfolio evaluations. They are generally at a slightly better solution for the initial anneal and then surpassed by BF until the later stage of the anneals shown above. That said there is nothing like the magnitude of difference seen in the 1% case. SBF does very poorly as expected.



(b) The above figure demonstrates how the value of $\hat{R}_k$ at which the algorithms are operating changes as the anneal progresses. SBF never gets close to 5% and is driven even higher as the algorithm progresses. The SA algorithms all start high, but quickly adjust to a low value of $\hat{R}_k$ and remain at that value. BF starts low, but struggles to maintain a low level of $\hat{R}_k$ from about halfway through the anneal.



(c) In this figure we can see the number of samples each algorithm used to estimate $V_D(R_T)$ as it progressed. SBF and BF remain fixed at their initial values and SA rises steadily (to above 10^4 samples). It is surprising that this increase in $|\mathbf{X}|$ matches that of 1% (figure 3.5c). This suggests that β has a much stronger impact on the size of $|\mathbf{X}|$ than R_T.

Figure 3.6: The algorithms have each optimised a portfolio at 5%, or equivalently, the 95% Value at Risk.

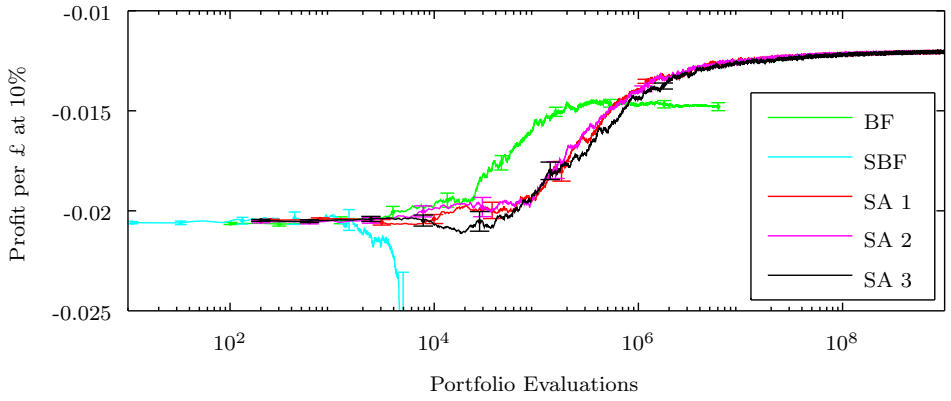


Figure 3.7: The above figure shows the result for $R_T = 10\%$ (equivalent to 90% Value at Risk). We can see a similar performance as compared to the case when $R_T = 5\%$. BF is initially quicker, but is significantly outperformed by the final result of the SA algorithms.

Figure 3.5 contains the key results when $R_T = 1\%$. Firstly, we can see from figure 3.5a that the SA algorithms all perform well. However, as expected, SBF fails to reach a good solution. This is because SBF simply doesn't use enough samples of X to counteract the pull of $e^{\beta V_D(R_T)}$. The poor solution is unlikely to be caused by the annealing schedule as the annealing schedule is reasonably adaptive to how the optimisation is progressing. We also see that the new algorithm variants (SA1-SA3) actually find a significantly (outside error bars) better final result than BF. This is a highly encouraging result.

Figure 3.5b plots $\hat{R}_k$ for each algorithm. As we would expect, BF maintains 1% constantly for most of the simulation. More surprisingly, BF's 10^4 sample size is not sufficient to stop $\hat{R}_k$ increasing significantly towards the end of the anneal. This potentially explains why SA1 - SA3 are able to reach a better solution than BF. We also note that SBF has an ever increasing $\hat{R}_k$ which reaches values as high as 60%. As discussed previously, this is likely to have lead to the sub-optimal solution found by SBF. The SA algorithms do not get $\hat{R}_k$ as low as we would like. This is likely due to the fact that whenever we increase n we also increase k which will negate part of the increase in n. Additionally, we often increase β at the same time as increasing n meaning the system is often trying to catch up with the current value of β.

Figure 3.5c shows how $n = (|\mathbf{X}|)$ changes during the simulation. It appears we made a fortuitous choice of how to increase the samples in SA1 as it broadly tracks SA2 and SA3. As highlighted above, this also demonstrates the effectiveness

of both SA2 and SA3 as they were both able to self tune. This self tuning ability is ideal in any algorithm. SA3 was able to select both when to increment n and the desired magnitude of such an increment. In contrast, SA2 only selected when to increment n. If we assume that the increment of n available to SA2 is always sufficient to maintain a low value of $\hat{R}_k$, we would expect SA2 to be able to match the final result of SA3, although SA2 might take longer to find it. This is because for SA2 it is likely that n will be increased by a relatively large amount whenever $\hat{R}_k$ gets too high. In contrast, SA3 will be able to choose a smaller increment when $\hat{R}_k$ gets high, but n will need to be increased more regularly. This means that SA3 should be able to track the optimal curve for n more accurately than SA2 (given a choice of β). In contrast, we would expect SA2 to use a choice of n that is higher than the optimal value. This improved tracking of the optimal choice of n should lead to a faster performance by SA3.

We also note that the new algorithms require significantly less samples from X in their early stages. This could have a significant advantage when it is either very difficult or very expensive to draw samples from X. This is relevant in financial problems when there is limited historical data available for the underlying assets and it is not possible to construct a model to generate further sample points.

Overall, our new algorithms have performed very well at 1% and all the metrics/estimators have performed as predicted. This is a very strong and encouraging result.

Figures 3.6 and 3.7 show the results for $R_T = 5\%$ and $R_T = 10\%$ respectively. The indicators for $R_T = 10\%$ (not included) demonstrate the same patterns of those for $R_T = 5\%$.

For the higher values of R_T we see a slightly different behaviour compared to the 1% case. Whilst the SA algorithms continue to show a better final result versus BF, they take (marginally) longer to do it. This means the SA algorithms are not as efficient at finding a reasonable solution, given a strongly constrained number of portfolio evaluations. SBF consistently performs very badly at these risk levels and will not be discussed further.

Figure 3.6b shows how $\hat{R}_k$ varies over the anneal. The results are largely similar to the 1% case. However, we do see $\hat{R}_k$ rising much more quickly in the BF case, which would corroborate with its lower final performance.

The most surprising result is in figure 3.6c which shows the number of samples of X required as the algorithm progresses. The number of samples required at each point roughly matches that of the 1% case. This implies that the number of samples required to correctly evaluate $V_D(R_T)$ is much more reliant on the size of β than

the notional choice of R_T.

As discussed above, the results for $R_T = 10\%$ closely mirror those of the 5% case. We see that whilst the performance of the final result is further improved, the SA algorithms reach a less adequate result for a constrained number of portfolio evaluations.

3.7.1 Example Specific Comments

We now discuss the specific financial model used and some of the surrounding details. We wish to find a distribution that can represent the tail risk of the individual stocks, and hence the tail risk of the whole portfolio. We would like such a model to include correlations as we expect there to be potential hedges in the portfolio. One option would be to use historical data. This is, in some sense, the most representative model of likely future returns. The problem with this is that if you are using daily returns as your historical data you need to decide how far back in time you believe to be relevant. Assuming there are approximately 250 trading days per year, 2 years of data would give you 500 historical samples and 5 years 1250 samples. These are unlikely to be a sufficient number of samples for the algorithm to use so we must fall back to a fitted model of some form. We describe one such model below.

We generate a model of X by fitting the daily (percentage) returns of the ten stocks (ABF.L, ADN.L, ADM.L, AGK.L, AMEC.L, AAL.L, ANTO.L, ARM.L, AZN.L, AV.L) from 2012 to a ten dimensional multivariate skewed Student-t distribution. This model is gaining credence in the literature, as highlighted in the introduction. See [Enerlein and Keller, 1995; Venter and de Jongh, 2002; Schmidt et al., 2006; Eberlein and Prause, 1998] for further information. This model captures both heavy tails (if they exist) and the correlations between the stocks. The data was fitted to the distribution using a Multi Cycle, Expectation, Conditional Expectation (MCECM) algorithm as implemented in the R package ghyp [Breymann and Luthi, 2013]. More information on MCECM can be found in [McNiel et al., 2005].

As described above, the design space is mathematically defined as $\mathcal{D} = \left\{D \in [-5, 5]^{10} : \sum_{d_i \in D} = 1\right\}$. Such a design space states that we allow short selling of stock and that we require we spend exactly £1 on our portfolio. It is possible to pick a portfolio that sells £2 of one stock and buys £3 of another. For the sake of simplicity, it was assumed that pounds were infinitely divisible.

The results have been largely described above; however, we come back to the question of using historical data to perform the optimisation. We previously discussed that the SA algorithms require significantly fewer samples in the early stages of the optimisation process. Unfortunately, to get even an approximate result using

these algorithms still required a sample set containing more than 1250 elements, which equates to over 5 years of historical data. In later chapters we attempt to reduce this requirement on the sample set size.

In this section we have explored a range of numerical results allowing us to make several key observations. The results have consistently been strong and highly encouraging.

3.8 Discussion

In this chapter we have studied the implications of using simulated annealing to optimise a simple risk problem. We initially analysed the problem analytically to find the expected Markov Chain the optimisation would follow (see section 3.3). Using this knowledge we were able to find two key indicators of the process in section 3.4. We tested these indicators in section 3.5 before adapting the annealing algorithm to take advantage of them in section 3.6. We studied a simple financial example in section 3.7 that demonstrated strong results.

The discovery of the indicators and their adaptation into the annealing algorithm has proved to be an effective combination. We have consistently seen improved final results and in certain cases seen these results reached more quickly than traditional methods.

The challenge remains to find an optimisation method that can significantly reduce the number of samples of X (the joint distribution of the random variables that are composed to form F_D) required. Ideally, this would be an effective method for $n < 500$, as this would allow market data from the previous two years to be used as a representative sample. We pursue this challenge further in the next chapter, where we will study the system from a physical standpoint and some of the intuition that flows from this will allow us to reduce k (as suggested at the end of section 3.4).

The most surprising result of the chapter came in the discovery that n ($= |\mathbf{X}|$) appears to be very closely linked to β. This appears to be the case almost to the complete exclusion of the choice of R_T. We will delve into this question more deeply in the next chapter (4) where we discuss the scaling relationship between some of the parameters.

It is also worth noting some of the current limitations of the method. As of yet we cannot apply the method to risk-reward style problems which would be more typical of portfolio optimisation problems. In these cases, each extra unit of risk must be supported by an increased expected return. The ratio between the expected

return and risk level is set by the user. We probe into some of these problems in chapter 5.

It is also possible to use our method to solve some Service Level Agreement (SLA) style problems. On the surface, the problem described in section 3.2 appears very similar to that of a SLA problem. However, in a regular SLA problem we would expect the user to set a function describing both the implementation cost of a given design, for example $c(D)$, and a penalty function if a certain threshold was hit, for example $p(D)$. An example of this would be a requirement that 95% of calls to a call centre should be answered within 5 minutes. We would then seek to optimise $\arg\min_D (c(D) - \langle p(D) \rangle)$ in order to maximise profits. In this case calculating $\langle p(D) \rangle$ would rely on an estimate of the quantile at which call time exceeds 5 minutes and not the value of the quantile at a specific level of risk. We cannot currently tackle this particular class of SLA problem, although it may be possible to define an alternative Markov Chain that would. We can however relate the two problems if we assume that the choice of $p(D)$ is such that the cost of constructing the system is significantly less than the penalty of failure, that the failures are independent and that the risk of failure is proportional to the cost of the design. Under these assumptions, we could rewrite the problem as $\arg\min_D (c(D) - \langle p(D) \rangle) \approx \arg\min_D (c(D))|_{V_D(R_T)<TOL}$ for some choice of R_T (which would be given by the penalty clause) and $V_D(R_T)$ which measures the indicator (at a specified level of risk) which must not exceed some tolerance, TOL. It is likely that this would be equivalent to $\arg\min_D (c(D))|_{V_D(R_T)=TOL}$, which with some work could be mapped on to the class of problem the algorithm currently works on. To do this we would require some of the ideas we will develop in chapter 5, when we consider multi point optimisation problems (in this case we wish both low $c(D)$ and $V_D(R_T)$).

Finally, we note that the combination of the new algorithm with the intuition gained from the analytic derivations has resulted in a very successful new method. The method has proved very effective at providing improved solutions over that of traditional approaches. This has been coupled with the ability to gain deep insight into how such an optimisation process would work.

This strong foundation will be built on in the following chapters to further investigate, improve and tease out greater intuition into the proposed algorithm. In chapter 4 we will approach the problem from a physicist's perspective and in chapter 5 we will show how this method could be extended to other problems.

Chapter 4

Optimising Quantile Risk: Interpreting the Physics

4.1 Introduction

In this chapter we continue to discuss and develop ideas on how to optimise a system at a given level of risk. Each of chapters 3 - 5 examines the same fundamental system. The first two chapters, 3 and 4, consider the construction of the method and it's asymptotic behaviour. Chapter 5 then draws on both of these initial chapters to further extend the work.

In this chapter we examine the various physical properties of the newly created ensemble. We consider several simplifications to observe asymptotic behaviour and also look at how the latent energy in the system changes.

The most intriguing result of this chapter is that when we frame our problem in thermodynamic terms we can directly calculate the entropy of the ensemble without resorting to a myriad of relatively complex techniques.

As previously discussed we wish to optimise a criterion on a random variable which is parametrised by a set of parameters (encapsulated by D). Specifically, we wish to find the choice of D which leads to the highest value of the $R_T\%$ quantile of the random variable. Since we do not have access to an inverse cumulative distribution function for the random variable, we are forced to estimate this optimisation quantity using sampling. Motivation for such a problem can be found in 3.1 and 1.2.

We start off by simplifying the Markov weights found in the previous chapter into a more manageable form by using a Poisson approximation.

Using the Poisson approximation we find a clear and fruitful interpretation of

the consequence of the optimising using sampled estimates of the quantile desired: that our system finds the solution that has the best projected worst case scenario. Additionally, we find that the targeted risk-level of the system and the measured risk-level can be combined to get a better risk estimate. This means that we do not need to work our system quite so hard to achieve the same level of optimisation.

In order to exploit the above approximations in our algorithm, we will need to change which of the ordered samples we use for the optimisation. In the previous chapter we always used the $k = \lfloor nR_T \rfloor$th ordered sample. However, the approximations we will make in this chapter will encourage us to reduce k to one and hence simply use the lowest valued sample. Unfortunately, the estimators we originally developed for $\langle R_k \rangle$ and $\frac{d\langle R_k \rangle}{dn}$ may not work so well in this context. Due to this, we will construct alternative estimators for these expectations in section 4.4.

For both of these approximations we show the viability of the estimate using a model (parametrised) choice of inverse cumulative distribution function (which is heavy tailed) for our random variable. We also provide some derivations (where possible) to further support these approximations.

We then use both of these approximations to further improve and enhance the algorithm proposed in chapter 3. We numerically test these improvements.

At the end of the chapter we briefly consider the scaling properties of the system. The result has limited impact on our algorithm but proves interesting none the less.

We finally conclude the chapter in section 4.8 and in context with the rest of the work in chapter 8.

4.2 Problem Definition

We briefly summarise the problem definition. The full problem definition can be found in section 3.2.

Our eventual aim is to maximise the quantile of a random variable at a given level of risk, $R_T \in [0, 1]$. This random variable is parametrised by a choice of design, $D \in \mathcal{D}$, and it's distribution has an associated inverse cumulative distribution function, $V_D(.)$. We assume $V_D(.)$ cannot be found analytically and can only be estimated by sampling. We wish to maximise $V_D(R_T)$, by changing D, for a fixed choice of R_T. Simulated annealing is used to perform the optimisation.

This style of problem has applications in both finance and vehicle routing. We could imagine that D encapsulates the design of a portfolio and $V_D(.)$ its risk curve. We wish to maximise the value of the portfolio at a specific level of (tail)

risk.

In this chapter we wish to study the thermal system that is created by this optimisation (when the temperature is fixed). We first consider the Markov (or thermal) weights of this system and then various approximations of these weights. These approximations allow us to gain insight into how the system operates. Furthermore, as we consider each approximation, we also discuss if there are any algorithmic implications on the original simulated annealing algorithm.

4.3 Markov Chain

In this section we describe the Markov weights of the simulated annealing process at each step (where the inverse temperature, β, is fixed). A brief (intuitive) argument is contained in this section. A full discussion of how these weights can be found is contained in chapter 3.

We wish to focus on the case where we expect the risk level we are optimising at, R_T, to be small. As previously discussed, we assume that $V_D(.)$ (the inverse cumulative distribution function) cannot be calculated analytically and so must be estimated via sampling. We use n to denote the size of the sample set used to estimate $V_D(R_T)$ and we further assume that n is large ($n \geq 100$). We calculate the Markov weights for the resulting chain at fixed temperature β^{-1}.

We consider an ensemble where the weights are based on stochastic estimates of $V_D(R_T)$. These estimates are found by drawing samples from the distribution of the random variable F_D, ordering the samples and selecting the kth ranked sample. We can then parametrise this kth ranked sample in terms of the true quantile that its value corresponds to, denoted R_k. The power of this step is that R_k itself has a simple probability density $\zeta_k(R_k)$ which we know exactly. The resulting weights for a given design are then given by

$$W_B(D|k, n, \beta) = \int_0^1 e^{\beta V_D(R_k)} \zeta_k(R_k) dR_k.$$

As a reminder, we note that X denotes the joint distribution of the distributions of the underlying random variables which are composed to form F_D.

As one would expect, $\zeta_k(R_k)$ is dominated by $R_k = \frac{k}{n}$. However, we would expect the chain weight to be dominated by $R_k \geq \frac{k}{n}$, for fixed k, because the thermal term will seek to drive R_k up, as an increased R_k would result in a perceived improvement of $V_D(R_k)$. It should be clear at this point that as β increases so must n if order to hold R_k down.

We can approximate $\zeta_k(R_k)$ by considering the number (or rate) of samples that will have a probability of occurrence less than R_k. We would expect (on average) this number, or rate, to be $R_T n$, assuming we are taking n samples. Given that $\zeta_k(R_k)$ is the relative probability density that the kth sample (of n) has probability of occurrence R_T, we require that $(k-1)$ of the samples have probability of occurrence less than R_k. This construction leads to a Poisson distribution. This implies $\zeta_k(R) \propto (R_k n)^{k-1} e^{-R_k n}$. This conclusion (and result) is reached more formally in section 3.3.

By combining the above results we have an expected Markov weight for a given design of

$$W_P(D|k,n,\beta) = \int_0^\infty (R_k n)^{k-1} e^{\beta V_D(R_k) - R_k n} dR_k \tag{4.1}$$

for some choice of k. We also note that the Poisson approximation above effectively removes the upper bound on R_k, and because of this we integrate over the full range of $R \in [0, \infty]$. This approximation induces an error of at most $O\left(e^{-n}\right)$.

We appear to have no direct estimator of $\langle R_k \rangle$ in this system, without which we would expect $\langle R_k \rangle$ to be very difficult to calculate. This is similar to the case of entropy in a regular thermal ensemble. The calculation of entropy usually requires running a sequence of related Markov chains and then (effectively) normalising the results from each chain. However, although we optimise the system using just the kth ordered sample, we can use the remaining samples to gain further information on what is happening.

The above idea makes it possible to find results which allow us to simply estimate $\langle R_k \rangle$ and $\frac{d\langle R_k \rangle}{dn}$. By considering the expectation $\left\langle e^{\beta V_D(R_{k+1}) - \beta V_D(R_k)} \right\rangle$ and the ratio between the probability densities $\zeta_k(R)$ and $\zeta_{k+1}(R)$ we can expand and rearrange the expectation to find that

$$\left\langle e^{\beta V_D(R_{k+1}) - \beta V_D(R_k)} \right\rangle = \langle R_k n \rangle .$$

Full detail of this calculation can be found in section 3.4.

In this section we have constructed the Markov weights of our optimisation from a very intuitive (and physical) perspective. We further develop the approximate solution found here in later sections in order to derive several highly interesting and helpful results.

4.4 New Risk Estimators

As discussed in the introduction (4.1), the approximations we will make in the next two sections (4.5 and 4.6) will cause us to optimise the lowest valued sample ($k = 1$). This is in contrast to the previous chapter where we optimised the $k = \lfloor nR_T \rfloor$th ranked sample.

Unfortunately, the estimators we developed for $\langle R_k \rangle$ and $\frac{d\langle R_k \rangle}{dn}$ in the previous chapter (3.4) (and briefly discussed in section 4.3) may not work so well when $k = 1$. This is because the estimator in this case becomes $\left\langle e^{\beta V_D(R_2) - \beta V_D(R_1)} \right\rangle = n \langle R_1 \rangle$. This estimator has the unfortunate property that such an estimator is (exponentially) dominated by small values of $V_D(R_1)$, which we expect to occur very rarely when β is high due to the strong upward bias from $e^{\beta V_D(R_1)}$. This is likely to result in the estimator underestimating $\langle R_1 \rangle$ for high β as it will struggle to sample any small values of $V_D(R_1)$. This was not such a concern when k was high, as the same choice of β would have implicitly required a larger choice of n, and this would in turn have reduced the gap between $V_D(R_{k+1})$ and $V_D(R_k)$ and hence made it much easier for the estimators to converge, despite the $e^{\beta V_D(R_k)}$ term.

In light of the above comments we now consider alternative estimators for $\langle R_k \rangle$ and $\frac{d\langle R_k \rangle}{dn}$. Assume we take a sample of n' draws of X, which we label $\mathbf{Y} = \{x_i \sim X : 1 \leq i \leq n'\}$. We label the value of R induced by each member of this sample set to be $R_i^Y \left(= V_D^{-1}(f_D(x_i))\right)$. By considering how many of these have $V_D(R_i^Y) < V_D(R_k)$, for a fixed choice of k, D and $\mathbf{X}$ (which implicitly fixes R_k), we can estimate $R_k = \frac{1}{n'} \sum_{i=1}^{n'} I\left(V_D(R_i^Y) < V_D(R_k)\right)$ where $I()$ is the indicator function. (As a reminder, $\mathbf{X}$ is the set of samples used to estimate $V_D(R_T)$). We can extend this to calculate an estimate of $\langle R_k \rangle$ over l steps of the chain, which gives

$$\hat{R}'_k = \frac{1}{ln'} \sum_{j=1}^{l} \sum_{i=1}^{n'} I\left(V_D(R_i^Y) < V_D(R_{k,j})\right), \tag{4.2}$$

where $R_{k,j}$ is the value of R_k at step j. We use $\hat{R}'_k$ to denote the result of this new estimator.

We can also find an estimator for $\left\langle R_k^2 \right\rangle$. We know (from binomial statistics) that $\left\langle h^2 \right\rangle - \langle h \rangle^2 = n'(R_k(1-R_k))$, where $h = \sum_{i=1}^{n'} I\left(V_D(R_i^Y) < V_D(R_k)\right)$, for fixed k, D and $\mathbf{X}$. We can rearrange this to see that $\left\langle h^2 \right\rangle = n'(R_k(1-R_k)) + \langle h \rangle^2$. We can then substitute $\langle h \rangle = n' R_k$ and further rearrange to find that $R_k^2 = \frac{\left\langle h^2 \right\rangle - \langle h \rangle}{n'(n'-1)}$. This can again be extended to calculate an estimate of $\left\langle R_k^2 \right\rangle$ over l steps of the chain.

This results in

$$\widehat{R_k'^2} = \frac{1}{ln'^2(n'-1)} \sum_{j=1}^{l} \left(H_j^2 - H_j\right), \tag{4.3}$$

where $H_j = \sum_{i=1}^{n'} I\left(V_D(R_i^Y) < V_D(R_{k,j})\right)$.

Finally, we note that $\frac{d\langle R_k' \rangle}{dn} \approx \hat{R_k'}^2 - \widehat{R'^2_k}$. This can be seen by applying the same arguments as in theorem 3.4.4 and used to find equation 3.12 to the Poisson approximation of the Markov chain induced by the approximation. Combining this result with the above results means that we now have alternative estimators for both $\langle R_k \rangle$ and $\frac{d\langle R_k \rangle}{dn}$.

These estimators should be effective and converge quickly as they are very closely based on binomial statistics. They do however provide limitations on how the algorithm is constructed. Nominally, it would appear we need to perform an additional n' portfolio evaluations per iteration, which could quickly prove expensive. However, if we assume that we only ever perturb $\mathbf{X}$ using new independent samples from X, then we can reduce the impact of the extra evaluations by reusing the elements of the sample set used to calculate the estimators as candidate elements for the attempted perturbations of $\mathbf{X}$. Algorithm 3.2 allows us to do this but we cannot then make any further improvements to how we perturb $\mathbf{X}$.

We will utilise the above estimators for $\langle R_k \rangle$ and $\frac{d\langle R_k \rangle}{dn}$ when we develop adapted optimisation algorithms in sections 4.5 and 4.6.

4.5 Bias to Zero Risk

4.5.1 The Approximation

In this extended section we explore some of the unexpected properties of the weights found in the previous section. We investigate where these properties apply through various analytical and numerical results. Finally we discuss algorithmic implications and provide numerical results.

We apply a steepest descent argument to the weight in equation 4.1 for the case when $k = 1$. If we assume the integral with respect to R_k is dominated by a particular value, say $\hat{R}$, then we have $\beta V_D'(\hat{R}) - n = 0$. This is found by maximising the exponent of e. Our steepest descent assumption implies this relation can be solved for a (single) global maximum. This gives an equation for $\hat{R}$. Substituting

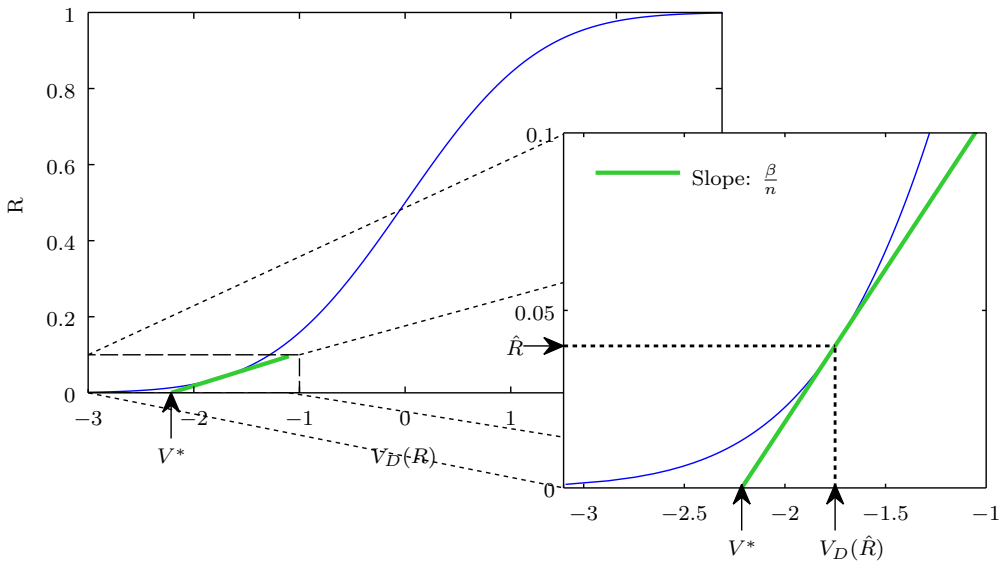


Figure 4.1: The figure above shows the implicit bias to zero risk that occurs when we use the worst of n samples to estimate $V_D(R_T)$. The plot shows a simple choice of $V_D(R_T)$ which we are attempting to optimise at the 5% quantile (95% Value at Risk). The result in equation 4.4 conjectures that we actually bias our chain towards choices of D where the system's (tangential) view of 0% risk is maximised, marked as V^*, instead of $V_D(.05)$.

this back into the weight gives

$$W_P(D|k,n,\beta) \doteqdot e^{\beta\left(V_D(\hat{R})-\hat{R}V'_D(\hat{R})\right)} = e^{\beta V^*_D(\hat{R})} \tag{4.4}$$

where $V^*_D(\hat{R}) = V_D(\hat{R}) - \hat{R}V'_D(\hat{R})$. We define $W_S(D|k,n,\beta) := e^{\beta V^*_D(\hat{R})}$.

This result crucially implies that our system is actually seeking to minimise the random variable, F_D, at the estimated (tangential) 0% risk. In other words, our system is finding the solution that has the best projected worst case scenario. Furthermore, since β and n control the depth of sampling, these parameters control the perception the chain has of the worst case scenario for a given random variable F_D.

Figure 4.1 shows this pictorially. We can see how this bias operates and the impact that β and n have on the depth of sampling. If we were to increase n at fixed β we would drive down R_1 and consequently be able to better estimate the zero risk point. Conversely, if we were to increase β whilst fixing n, we would drive

up R_1 and reduce the quality of our estimate of the zero risk point. These comments assume that $V_D''(R_1) < 0$, but this is highly likely to be the case in most real world problems in the risk tail.

This result is particularly interesting because we can argue that this is what you would naturally wish to do if you had no computation constraints. The decision to optimise at the 95% or 99% Value at Risk (the 5% or 1% quantile respectively) is often an arbitrary choice. The choice is based on of what is thought to be a reasonable, yet still calculable, representation of zero risk. In the above description, we have simply disregarded any notion of the quantile and focus solely on the amount of work to be invested in the problem.

The critical question we have yet to discuss is under which conditions such an approximation might be valid. In the next two subsections we provide various analytic and heuristic arguments that demonstrate this bias is taking place and attempt to isolate the conditions under which the steepest descent approximation behind it might be valid. In sections 4.5.4 and 4.5.5 we consider the algorithmic implications of this result and apply the resulting algorithm to a test problem.

4.5.2 An Example $V_D(R)$

In this section we use a plausible example of $V_D(R)$ in order to further investigate when this bias to zero risk, as uncovered in section 4.5.1, might occur. We assume F_D is distributed with an exponential distribution of the form $\frac{dR}{dV} = V_0^{-1} e^{\frac{V}{\epsilon}}$, where V_0 is a parameter. We can then integrate $\frac{dR}{dV}$ with respect to V and rearrange to find that $V_D(R) = \epsilon \log\left(\frac{R}{R_0}\right)$, where $R_0 = \epsilon^{-1} V_0$. For this particular choice of $V_D(R)$, a given choice of the design variable, D, is encapsulated by both $\epsilon(D)$ and $R_0(D)$.

Given this choice for $V_D(R)$, we can analytically integrate the Markov weights with respect to R_1. Below we calculate these for each of the Markov weights found thus far. We consider the Binomial weights found in section 3.3, the Poisson weights found in section 4.3 and finally the steepest descent weights found in section 4.5.1.

We are also able to find analytic expressions for each of the moments of R_1 in the Binomial and Poisson cases. For the steepest descent case we can calculate the value of R_1 we expect to be dominant.

We then use these weights to evaluate how the relative (and implicitly the absolute) error of the various approximations changes. We use the variance and expectation of R_1 to support this discussion. The expectation of R_1 also proves useful in the next section when we study a numerical example.

Given a choice of $V_D(R) = \epsilon \log\left(\frac{R}{R_0}\right)$ and $k > 0$ we then find Markov weights

of

$$W_B(D|k,n,\beta) \propto R_0^{-\beta\epsilon}\frac{\Gamma(\beta\epsilon+k)}{\Gamma(n+\beta\epsilon+1)} \tag{4.5}$$

$$W_P(D|k,n,\beta) \propto (nR_0)^{-\beta\epsilon}\Gamma(\beta\epsilon+k) \tag{4.6}$$

$$W_S(D|k,n,\beta) \propto \left(\frac{\beta\epsilon+k-1}{nR_0}\right)^{\beta\epsilon}(\beta\epsilon+k-1)^{k-1}\,e^{-(\beta\epsilon+k-1)}\sqrt{\beta\epsilon+k-1} \tag{4.7}$$

where $W_B(D|k,n,\beta)$ is the Markov weight calculated using the Binomial weights (3.3), $W_P(D|k,n,\beta)$ the Poisson weights (4.3) and $W_S(D|k,n,\beta)$ the steepest descent weights (4.5).

Furthermore, the moments in R, for $j \geq 1$ and fixed ϵ, k and n, are given by

$$\left\langle R_k^j \right\rangle_B = \frac{\Gamma(\beta\epsilon+j+k)\Gamma(n+\beta\epsilon+1)}{\Gamma(\beta\epsilon+k)\Gamma(n+j+\beta\epsilon+1)} \tag{4.8}$$

$$\left\langle R_k^j \right\rangle_P = \frac{\Gamma(\beta\epsilon+j+k)}{\Gamma(\beta\epsilon+k)}n^{-j} \tag{4.9}$$

where $\left\langle R_k^j \right\rangle_B$ is the expectation of R_k^j with respect to the Binomial weights (3.3) and $\left\langle R_k^j \right\rangle_P$ the Poisson weights (4.3).

Finally, we expect the steepest descent case to be dominated by

$$\hat{R} = \frac{\beta\epsilon+k-1}{n}. \tag{4.10}$$

We outline the evaluation for the Binomial weights as follows. We consider

$$W_B(D|k,j,n,\beta) = \int_0^1 R_k^j\binom{n}{k}kR_k^{k-1}(1-R)^{n-k}e^{\beta V_D(R_k)}dR_k,$$

for some choice of $j \geq 0$. Substituting $V_D(R_k) = \epsilon\log\left(\frac{R_k}{R_0}\right)$ and rearranging gives

$$= \int_0^1 R_k^j\binom{n}{k}kR_k^{k-1}(1-R_k)^{n-k}\left(\frac{R_k}{R_0}\right)^{\beta\epsilon}dR_k.$$

By collating R_k terms and moving constants outside the integral, we see that this

is equivalent to

$$= \binom{n}{k} k R_0^{-\beta\epsilon} \int_0^1 (1 - R_k)^{n-k} R_k^{\beta\epsilon+j+k-1} dR_k.$$

This integral is now in the form of the Beta function, with arguments $\beta\epsilon + j + k$ and $n - k + 1$, assuming $\beta\epsilon + j + k > 0$. We can rewrite the Beta function in terms of Γ functions to get (see definition 2.2.5)

$$= \binom{n}{k} k R_0^{-\beta\epsilon} \frac{\Gamma(\beta\epsilon + j + k)\Gamma(n - k + 1)}{\Gamma(n + j + \beta\epsilon + 1)}.$$

Since we wish to calculate our results with respect to both the design (ϵ and R_0) and j, we can disregard all terms not containing these variables. This gives that the above is proportional to

$$\propto R_0^{-\beta\epsilon} \frac{\Gamma(\beta\epsilon + j + k)}{\Gamma(n + j + \beta\epsilon + 1)}.$$

When $j = 0$ this gives us the Markov weight of

$$W_B(D|k, n, \beta) \propto R_0^{-\beta\epsilon} \frac{\Gamma(\beta\epsilon + k)}{\Gamma(n + \beta\epsilon + 1)}.$$

Finally, we can find the moments of R by selecting different values of j. This gives that

$$\left\langle R_k^j \right\rangle_B = \frac{R_0^{-\beta\epsilon} \frac{\Gamma(\beta\epsilon+j+k)}{\Gamma(n+j+\beta\epsilon+1)}}{W_B(D)} = \frac{\Gamma(\beta\epsilon + j + k)\Gamma(n + \beta\epsilon + 1)}{\Gamma(\beta\epsilon + k)\Gamma(n + j + \beta\epsilon + 1)}.$$

The results for the other weights follow analogously. In the Poisson case the integration leads directly to a Γ function for the moments.

In the steepest descent case we use the weight given by the second order Taylor's expansion ($W_S(D|k, n, \beta) = e^{\beta V_D^*(\hat{R})} \left(-\beta V_D''(\hat{R})\right)^{-\frac{1}{2}}$ for $k = 1$). To find the quantile that dominates the steepest descent case, when $k = 1$, we substitute $V_D'(R) = \frac{\epsilon}{R}$ into $\beta V_D'(\hat{R}) - n = 0$ and solve for $\hat{R}$. (The second equation is derived at the start of section 4.5). This means for $k = 1$ we find that $\hat{R} = \frac{\beta\epsilon}{n}$. For $k > 1$ the proof is analogous.

Using the weights calculated above we are able to calculate how the relative (and consequently absolute) errors change with respect to n, β, ϵ and R_0. This is done in the corollary below.

The expressions for $W_P(D|k,n,\beta)$ and $W_B(D|k,n,\beta)$ lead directly to

$$\frac{W_P(D|k,n,\beta)}{W_B(D|k,n,\beta)} \propto n^{-\beta\epsilon}\Gamma\left(n+\beta\epsilon+1\right)$$

Similarly we obtain

$$\frac{W_S(D|k,n,\beta)}{W_P(D|k,n,\beta)} \propto (\beta\epsilon+k-1)^{\beta\epsilon}\,\frac{(\beta\epsilon+k-1)^{k-1}\,e^{-(\beta\epsilon+k-1)}}{\Gamma(\beta\epsilon+k)}\sqrt{\beta\epsilon+k-1}.$$

We can approximate $\Gamma(\beta\epsilon+k)$ using Stirling's Approximation (see proposition 2.2.4) to get (for large $n+\beta\epsilon+1$)

$$= (\beta\epsilon+k-1)^{\beta\epsilon}\,\frac{(\beta\epsilon+k-1)^{k-1}\,e^{-(\beta\epsilon+k-1)}}{(\beta\epsilon+k)^{\beta\epsilon+k-\frac{1}{2}}\,e^{-(\beta\epsilon+k)}\sqrt{2\pi}}\sqrt{\beta\epsilon+k-1}.$$

By rearranging the above equation we see that

$$\begin{aligned}
&= \frac{(\beta\epsilon+k-1)^{\beta\epsilon+k-1}\,e^{1}\sqrt{\beta\epsilon+k-1}}{(\beta\epsilon+k)^{\beta\epsilon+k-1}\,(\beta\epsilon+k)^{\frac{1}{2}}\,\sqrt{2\pi}\gamma(\beta\epsilon+k)} \\
&= \left(1+\frac{-1}{1+\beta\epsilon+k-1}\right)^{\beta\epsilon+k-1}\frac{e^{1}\sqrt{\beta\epsilon+k-1}}{(\beta\epsilon+k)^{\frac{1}{2}}\,\sqrt{2\pi}}.
\end{aligned}$$

For $\beta\epsilon >> 1$ this simplifies to

$$\approx e^{-1}\frac{e^{1}}{\sqrt{2\pi}}.$$

Finally, we note that we can remove the $\sqrt{2\pi}$ as we are only finding the result up to proportionality. This gives

$$\frac{W_S(D|k,n,\beta)}{W_P(D|k,n,\beta)} \propto 1$$

for large $\beta\epsilon$.

It then trivially also follows that

$$\frac{W_S(D|k,n,\beta)}{W_B(D|k,n,\beta)} \propto n^{-\beta\epsilon}\Gamma\left(n+\beta\epsilon+1\right)$$

for large $\beta\epsilon$.

The results above allow us to explore when the steepest descent (and Poisson)

approximations are valid. The first key observation is that R_0 has no impact on any of the errors, although it does shape the distribution. This means that the (normalised) weights from either approximation will be correct for a choice of design space which holds ϵ fixed and only varies R_0.

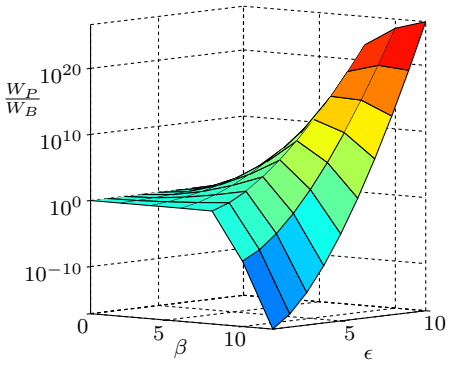
We further consider the error induced by the Poisson approximation. If we apply Stirling's approximation we get

$$\frac{W_P(D|k,n,\beta)}{W_B(D|k,n,\beta)} \approx n^{-\beta\epsilon}\,(n+\beta\epsilon+1)^{\left(n+\beta\epsilon+\frac{1}{2}\right)}\,e^{-(n+\beta\epsilon+1)}.$$

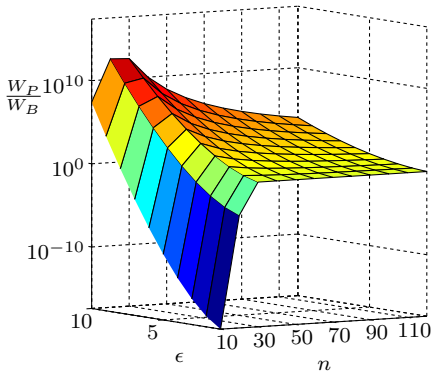
For $n >> \beta\epsilon$ we can further approximate this to $\frac{W_P}{W_B} \propto e^{-\beta\epsilon}$ and hence expect the Poisson weights to bias against large ϵ. If we instead assume that $n << \beta\epsilon$ then our approximation becomes $\frac{W_P}{W_B} \propto n^{-\beta\epsilon}\,(\beta\epsilon)^{\beta\epsilon}\,e^{-\beta\epsilon}$ and we now expect the Poisson weights to bias against small ϵ. We expect n to have limited direct impact under either regime. The error in the weights will increase disproportionately for higher values of ϵ and hence the normalised distribution (across $\mathcal{D}$) will become increasingly distorted as β increases, compared to the true normalised distribution. (This assumes that the design space, $\mathcal{D}$, contains different values of ϵ.) Figures 4.2a and 4.2b show this visually.

Finally, the additional bias induced by the steepest descent approximation is minimal. We can easily see that the error is entirely independent of n and reduces as β increases. Figures 4.3a and 4.3b show this numerically. It is important to note that as β grows in our system the steepest descent weights are largely driven by $\sqrt{\beta\epsilon}$ rather than $e^{\beta V_D^*(\hat{R})}$. If we were to assume that the weights were solely driven by $e^{\beta V_D^*(\hat{R})}$ then the error with respect to the Poisson distribution would increase with β. This view is further supported by considering the moments found in proposition 4.8. We can easily see that the variance of R_1 will reduce as n increases, but that the variance will increase as β increases. As the variance of R_1 increases it becomes more unlikely that the original Poisson integral is dominated by a single value.

In this section we have discussed the parameter regimes where we expect the steepest descent (and Poisson) approximations to hold. However, we have not discussed when this bias to zero risk will actually have an impact! It is entirely plausible that during an optimisation such a bias would still lead to the same choice of design. Mathematically, we may have that $\arg\max_D V_D(R_T) = \arg\max_D V_D^*(R_T)$ for a given choice of R. Figure 4.4 shows this visually. We discuss this question further in the next section (4.5.3).

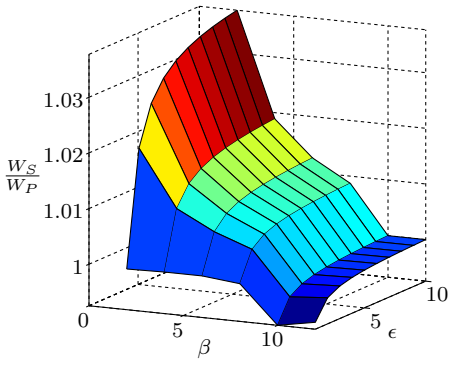


(a) The ratio increases with β. There is a transition in the rate of the ratio's growth at $\beta = 8$. This is when $\beta\epsilon \approx ne$ (taking $\epsilon = 10$ as the dominant value).

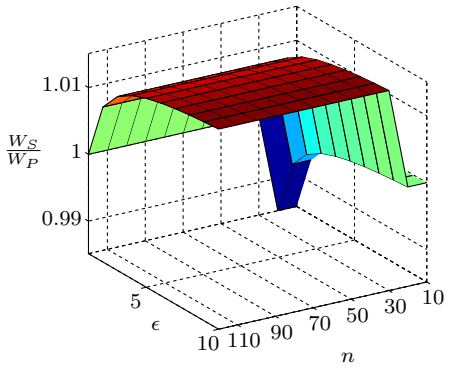


(b) The ratio initially reduces with n but as n grows the impact is greatly reduced.

Figure 4.2: The above figures show how the ratio between the Binomial and Poisson weights changes over time with respect to β and n. $R_0 = 1$. In figure 4.2a $n = 40$ and in figure 4.2b $\beta = 5$. The weights are numerically integrated over $\epsilon \in [1, 10]$ (for fixed β and n).



(a) The ratio reduces as β increases. However, it is worth noting that this convergence is significantly influenced by the $\sqrt{\beta\epsilon}$ term in the steepest descent weight, rather than by the bias to zero risk term $\left(e^{\beta V_D^*(R)}\right)$.



(b) The relative ratio induced by the steepest descent approximation is almost entirely unaffected by n.

Figure 4.3: The above figures show how the ratio between the Poisson and the bias to zero risk weights changes over time with respect to β and n. $R_0 = 1$. In figure 4.3a $n = 40$ and in figure 4.3b $\beta = 5$. The weights are numerically integrated over $\epsilon \in [1, 10]$ (for fixed β and n).

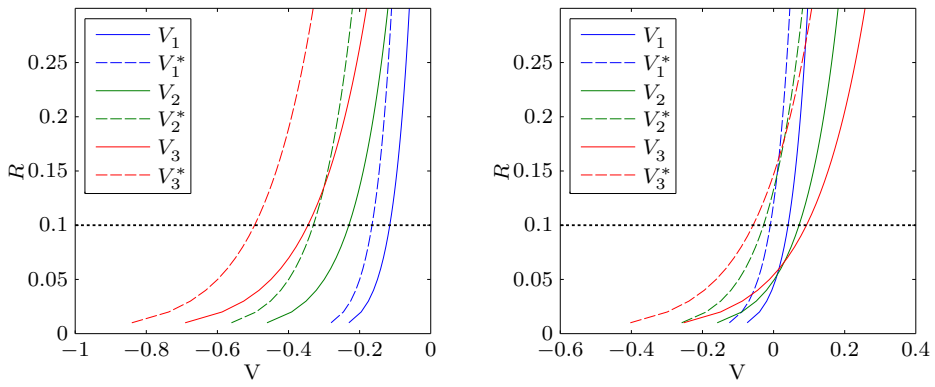


Figure 4.4: An implicit bias to zero risk may not always have an impact on the optimisation process. The above figures show two different systems, each with 3 different possible design choices. Both $V_D(R)$ and $V_D^*(R)$ are plotted. We can see in the problem to the left that $\arg\max_D V_D(0.1) = \arg\max_D V_D^*(0.1)$, whereas in the right hand problem $\arg\max_D V_D(0.1) \neq \arg\max_D V_D^*(0.1)$. We do not necessarily require that both maximisations lead to separate solutions, but it is an important consideration if we wish to be able to discuss the relevance of this bias to zero risk and observe it during an optimisation process.

4.5.3 Numerical Results

In this section we use our example $V_D(R)$ to find and numerically demonstrate this bias to zero risk at work. As a reminder, we have chosen $V_D(R) = \epsilon \log\left(\frac{R}{R_0}\right)$ as this is representative of a parametrised inverse cumulative distribution function for a class of exponential distributions. In this example the design is encapsulated by $\epsilon(D)$ and $R_0(D)$.

As touched on at the end of the previous section (4.5.2), in order to observe this bias at work we need to identify a class of $V_D(R)$ such that $\arg\max_D V_D(R_T) \neq \arg\max_D V_D^*(R_T)$ for a given choice of R_T. If the maximisations lead to the same solution then the bias should have no observable effect. Figure 4.4 shows how one solution might be true under either maximisation.

Given that $V_D(R) = \epsilon \log\left(\frac{R}{R_0}\right)$, we can find that $V_D^*(R) = \epsilon \log\left(\frac{R}{R_0}\right) - \epsilon$. We can rearrange this to find that $V_D^*(R) = V_D\left(\frac{R}{e}\right)$.

This suggests that for our choice of $V_D(R)$ we require that $\arg\max_D V_D(R)$ changes for different choices of R. To do this we must make R_0 a function of ϵ (or vice versa) as otherwise we would always require ϵ to be as small as possible for a given choice of R_T and $R_0 > 1$. We assume that $R_0(\epsilon) = e^{\epsilon a + b}$. This leads to $V_D(R) = \epsilon\left(\log(R) - b - \epsilon a\right)$. This suggests that ϵ is driven high when $R_T > e^b$ and low when $R_T < e^b$. In the case of $V_D^*(R)$ this change is moved to $R_T = e^{b+1}$. The exact value of $\arg\max_D V_D(R_T)$ is $\frac{\log(R_T) - b}{2a}$ (a controls how quickly the value of $\arg\max_D V_D(R_T)$ changes with respect to R_T).

We assume that $\epsilon \in [0, 1]$ and pick $b = \log(0.03)$ and $a = 0.5$. Using these parameter choices and the expectation for R_1 as found in equation 4.8 we find configurations of β and n where $R_T = 0.05$. The weights under this configuration should favour low ϵ for a system optimising $V_D(R_T)$ which uses sampling to estimate $V_D(R_T)$ (which in effect optimises $V_D^*(R_T)$), and high ϵ for a system directly optimising $V_D(R_T)$ (without any sampling bias). Figure 4.5 shows the selection of suitable n and β.

Figure 4.6 shows how this bias will effect our choice of $V_D(.)$. We can clearly see that the chain without any sampling bias is dominated by a larger choice of ϵ when compared to the chain which does have sampling bias. Increasing n (and β) appears to increase the dominance of this bias on the system.

We have successfully shown this bias to zero risk taking place in a toy system. In the previous sections we also explored the parameter regimes where we expect this bias to take place. In the next section, we use the intuition developed in these subsections to exploit this bias and improve our optimisation results.

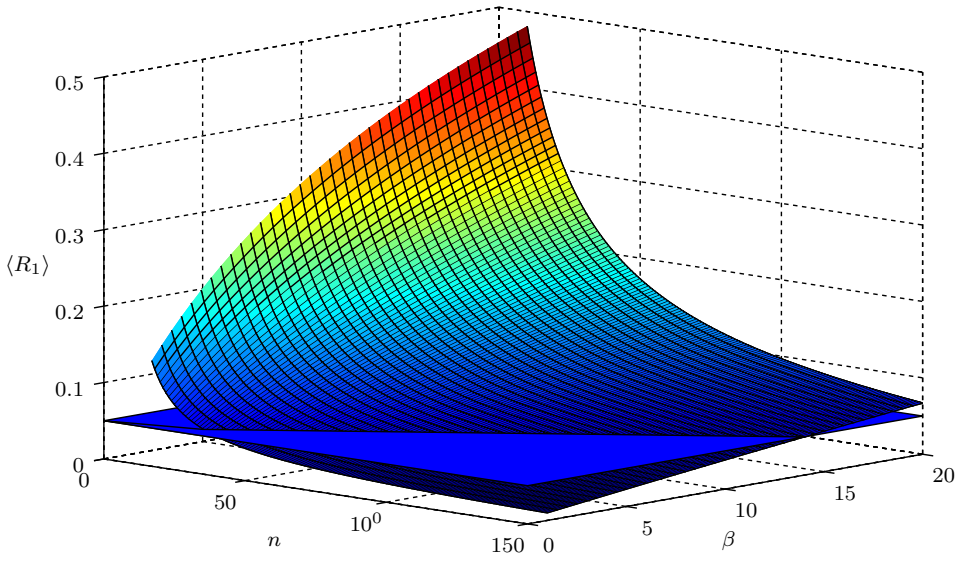
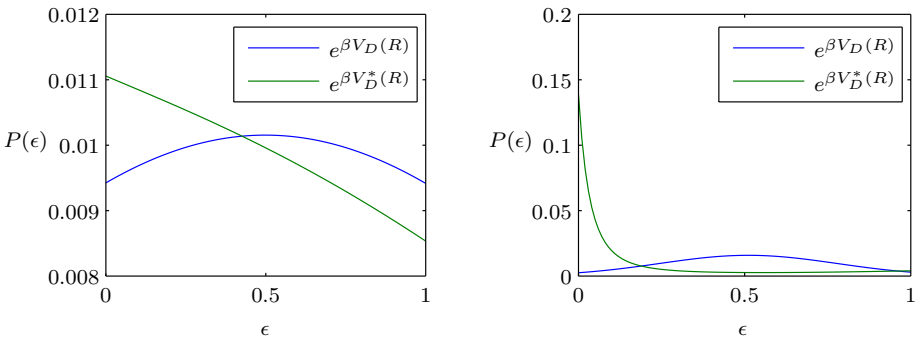


Figure 4.5: The above figure shows how $\langle R_1 \rangle$ varies with n and β for a simple system. The expectation was calculated using proposition 4.8 with numerical integration over ϵ. The blue plane contains the points where $\langle R_1 \rangle = 0.05$. The points of intersection between the two surfaces give the values of β and n which select a value of $\langle R_1 \rangle = 0.05$. Note how $\langle R_1 \rangle$ is pushed up by β and down by n ($k = 1$).



(a) $n = 25$ and $\beta = 0.6$ (b) $n = 150$ and $\beta = 14$

Figure 4.6: The above figures show a system that is biased towards zero risk. In both graphs $V_D(R) = \epsilon\,(\log(R) - b - a\epsilon)$. The figures show the probability of a given value of ϵ occurring. From this it can be seen that maximising $e^{\beta V_D^*(R_T)}$ pushes ϵ down whereas maximising $e^{\beta V_D(R_T)}$ (no sampling bias) pushes ϵ up. When we increase n (and consequently β to hold $\langle R_1 \rangle$ fixed) we see this bias has a more pronounced effect.

4.5.4 Algorithmic Implications

In this section we consider the algorithmic implications of this bias to zero risk effect. Fundamentally, we can vary the size of n at each temperature change to ensure we are optimising $V_D(R_T)$ at the correct (given) level of risk. This can be done in the algorithm using the results found at the end of section 4.3. We take the same approach in the previous chapter (see section 3.6).

Naively we would assume that if we wanted to optimise $V_D(R_T)$ at $R_T\%$ risk we would use the $k = \lfloor R_T n \rfloor th$ ordered point to estimate $V_D(R_T)$. However, since we fully expect the rank $k = \lfloor R_T n \rfloor$ ordered point to have a level of risk much greater than R_T for small n, we could simply set $k = 1$. We would then expect the number of samples (n) needed to be lower than in the case when $k = \lfloor nR_T \rfloor$. By using the above results we could ensure that the worst point in the sample is at $R_T\%$ risk, rather than the kth point. This should mean that our optimisation has this bias to zero risk, and that we are able to operate in this paradigm were we are really specifying a depth of sampling, or alternatively an amount of work to be done to best estimate $V_D(0)$.

This can be done by adapting the algorithm described in section 3.6 to hold $k = 1$ rather than increasing k with n. This would mean that on line 1 and line 27 of algorithm 3.2 we set $k = 1$ instead of $k = \max(1, \lfloor nR_T \rfloor)$. We label these adapted algorithms as

WSA1 n is increased by a fixed percentage every N_i accepted D moves, irrespective of the estimated value of $\langle R_1 \rangle$. This is the algorithm as described above, except that on line 28 we set the condition to always true and on line 26 we set $n := 1.2n$.

WSA2 In this algorithm we check if $\hat{R}_1 > R_T$ every $\frac{N_i}{2}$ accepted D moves. If $\hat{R}'_1$ is larger than the target value (R_T) then we increase n by a fixed percentage. This is the algorithm as described above, except that on line 26 we set $n := 1.2n$.

WSA3 In this algorithm we check if $\hat{R}_1 > R_T$ every $\frac{N_i}{2}$ accepted D moves. If $\hat{R}'_1$ is larger than the target value (R_T) then we increase n by an amount dictated by the estimator for $\frac{d\langle R_1 \rangle}{dn}$ equal to $\hat{R_k}^2 - \hat{R_k^2}$. This is the algorithm as described above with no further modifications.

The above algorithms all use the estimators originally developed in chapter 3 and discussed in section 4.3. All of the annealing algorithms used in this thesis are summarised in appendix A.

Algorithm 4.1 Simulated Annealing on n

$D := D_0$, $n := 100$
$\mathbf{X} :=$ n samples drawn from X
$V := \hat{V}_D(1, \mathbf{X})$
for $1 \leq i \leq m$ **do** ▷ For each cooling step
  for $1 \leq l \leq 2$ **do** ▷ Split the iterations in two
    $j := 0$
    while $j \leq N_i/2$ **do** ▷ For half of the iterations
      $\tilde{D} :=$ a perturbation of D
      $\tilde{V} := \hat{V}_{\tilde{D}}(1, \mathbf{X})$
      $p := \min(1, e^{-\beta_i(V-\tilde{V})})$ ▷ Metropolis Acceptance Function
      if $p \geq u$ where $u \sim U(0,1)$ **then**
        $V := \tilde{V}$ and $D := \tilde{D}$
        $j := j + 1$
      $\{y_a\}_{a=0}^n :=$ n samples drawn from X
      $V_a := f_D(y_a)$ for all y_a
      for $1 \leq a \leq n$ **do** ▷ Perturb $\mathbf{X}$ n times
        $h := \sum_{i=1}^n I\left(V_a < V_D(R_1)\right)$
        $A := A + \frac{1}{n}h$ ▷ Calculate the estimator for $\langle R_1' \rangle$
        $B := B + \frac{1}{n}\left(h^2 - h\right)$ ▷ Calculate the estimator for $\frac{d\langle R_1' \rangle}{dn}$
        $\tilde{\mathbf{X}} := \mathbf{X}$
        Replace one element of $\tilde{\mathbf{X}}$ with y_a
        $\tilde{V} := \hat{V}_{\tilde{D}}(1, \mathbf{X})$ ▷ Can use V_a to avoid a portfolio evaluation
        $p := \min(1, e^{-\beta_i(V-\tilde{V})})$ ▷ Metropolis Acceptance Function
        if $p \geq u$ where $u \sim U(0,1)$ **then**
          $V := \tilde{V}$ and $\mathbf{X} := \tilde{\mathbf{X}}$
    $\hat{R} := \frac{2}{nN_i}A$
    $d\hat{R} := \hat{R}^2 - \frac{2}{n^2(1-n)N_i}B$
    if $\hat{R} > R_T$ **then** ▷ Check the level of risk we have been optimising
      $n := \min\left(2000, n + \left\lceil \frac{R_T - \hat{R}}{d\hat{R}} \right\rceil\right)$ ▷ Increase n to push $\langle R_1' \rangle$ down
      while $|X| < n$ **do** ▷ Add new random samples to $\mathbf{X}$
        Draw a random sample from X and add it to $\mathbf{X}$
      $V := \hat{V}_D(1, \mathbf{X})$.
    $A := 0$ ▷ Reset estimators for $\langle R_1 \rangle$ and $\frac{d\langle R_1 \rangle}{dn}$
    $B := 0$
Result is D

We now consider how we could use the new estimators for $\langle R_k \rangle$ and $\frac{d\langle R_k \rangle}{dn}$ developed in section 4.4. Algorithm 4.1 describes how such an algorithm would work. We use the same notation as in algorithm 3.2. As a reminder, we define $\hat{V}_D(1, \mathbf{X})$ to be the estimate of $V_D(R_T)$ obtained by calculating $\{F_D(x) : x \in \mathbf{X}\}$, and then picking the worst valued point. $\mathbf{X}$ is a set of samples drawn from X, i.e. $\forall x \in \mathbf{X}, x \sim X$. The exact value of R_1 is not required by this estimate. $n = |\mathbf{X}|$ is the cardinality (size) of the set $\mathbf{X}$.

The key change from algorithm 3.2 (apart from setting $k = 1$) is to use the new estimators. As discussed in section 4.4, we can evaluate these new estimators very cheaply. This is because we can use the same portfolio calculations that we use to perturb $\mathbf{X}$ to calculate the estimators at each step. In this updated algorithm we attempt a greater number of perturbations of $\mathbf{X}$ between each design perturbation, which does slightly increase the work at each step. We perform more perturbations of $\mathbf{X}$ to ensure that we have a sufficient number of samples with which to evaluate the (new) estimators.

In summary, the resulting algorithm mostly comprises of a simple anneal. The twist is that we additionally measure estimates for $\langle R' \rangle$ and $\left\langle R'^2 \right\rangle$ during the optimisation. These expectations allow us to accurately control the number of samples to use at each step and hence to ensure the anneal has a reliable estimate of $V_D(R_T)$ to optimise with at each step.

We consider two different variants of this new algorithm. These are

qWSA2 In this algorithm we use the new estimators from section 4.4. We then check if $\hat{R}'_1 > R_T$ every N_i accepted D moves. If $\hat{R}'_1$ is larger than the target value then we increase n by a fixed percentage. This is algorithm 4.1, except that on line 29 we set $n := 1.2n$.

qWSA3 In this algorithm we check if $\hat{R}'_1 > R_T$ every N_i accepted D moves. If $\hat{R}'_1$ is larger than the target value then we increase n by an amount dictated by $\frac{d\langle R_1 \rangle}{dn}$. This is algorithm 4.1 with no modifications.)

In the next section we apply both of these algorithms to a sample problem, and then consider some numerical results.

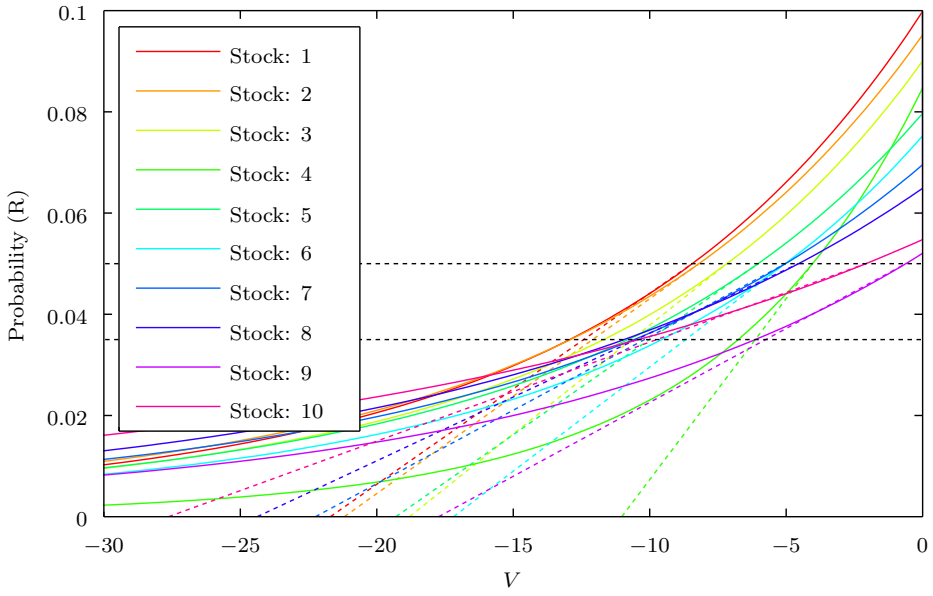


Figure 4.7: The above figure shows the implicit bias to zero risk that occurs for each (single) option our optimisation process could select. The optimisation process could also select fractional amounts of each option, as long as it sells a net total of £1 worth. It should be clear that, at least for the ten individual cases plotted, the best option is to select option 9 if optimising $V_D(0.05)$ and option 4 if optimising $V^*_D(0.05)$. There could potentially be a better solution to both problems which combines different stocks. The two dotted lines highlight $R = 5\%$ and $R = 3.5\%$.

4.5.5 Results

We consider a user who wishes to sell £1 of call options spread over ten stocks (the same stocks as the previous chapter). We assume that these options can be infinitely divided and that there is an open market on which they can be traded. As in the previous chapters the stocks are modelled using a multivariate skewed-t distribution (the same model as in chapter 3). The (one day) option prices were calculated using Black Scholes [Glasserman, 2004]. Figure 4.7 shows the tail of the cumulative distribution function ($V_D(0.05)$) of the different portfolios, where only a single option is sold. The system could decide to sell fractional units of multiple options, making the possible design space nine dimensional, $\mathcal{D} = [0, 1]^9$. Table 4.1 shows the stocks and the strike prices used.

As previously discussed, in this work we focus on the additional complications introduced by estimating $V_D(R)$ through sampling. Finding the optimal cooling schedule for a given simulated annealing problem is a well known and discussed

Stock	Current Price (p)	Strike Price (p)
ABF.L	1550.0	1568.9
ADN.L	381.6	390.7
ADM.L	1198.0	1228.4
AGK.L	1749.0	1789.6
AMEC.L	1035.0	1060.5
AAL.L	2000.5	2060.5
ANTO.L	1322.0	1366.6
ARM.L	799.5	821.7
AZN.L	2985.0	3025.5
AV.L	388.4	401.9

Table 4.1: The above table defines the stock and strike prices used for the call options to be sold. (The current price is the price of the stocks on Monday, 7 January 2013.) As before, the returns model used was fitted to daily returns from 2012.

problem. During our numerical tests we assume a very simple choice of cooling schedule. We increase β by 0.125 after every 200 accepted perturbations of D. β is initially set to 1. D was perturbed in the same way as in previous examples (3.7).

The system is optimised at $R_T = 0.05$. 48 identical optimisations were run for each method, the results were then averaged, and the standard error calculated. We expect the (q)WSA algorithms will in fact optimise $V_D^*(0.05)$.

Figures 4.8 to 4.12 show the results and demonstrate the bias to zero risk in action, when using qWSA2 and qWSA3.

Figure 4.8 shows the performance of the algorithms against $V_D(0.05)$. We would expect a brute force algorithm to optimise $V_D(0.05)$ and the (q)WSA algorithms to optimise $V_D^*(0.05)$. We see that BF10000 (a brute force algorithm using 10, 000 samples) finds the best solution to $\arg\max_D V_D(0.05)$. In contrast, BF400 fails to find a good solution, again highlighting the difficulty of choosing the correct number of samples to estimate a quantile during an optimisation process. We see that none of the (q)WSA algorithms perform as well as BF10000 at optimising $V_D(0.05)$, but most beat BF400. This is to be expected as we do not expect these algorithms to optimise $V_D(0.05)$.

Figure 4.9 shows the performance of the algorithms against $V_D^*(0.05)$. Whilst both qWSA algorithms do find a relatively better result than BF10000, the final result has not changed much from the starting point. This suggests that while a bias is taking place, it is not a strong one. The WSA algorithms both perform poorly when compared in terms of $V_D^*(0.05)$.

Figure 4.10 shows how both estimators for $\langle R_1 \rangle$ evolve during the optimisa-

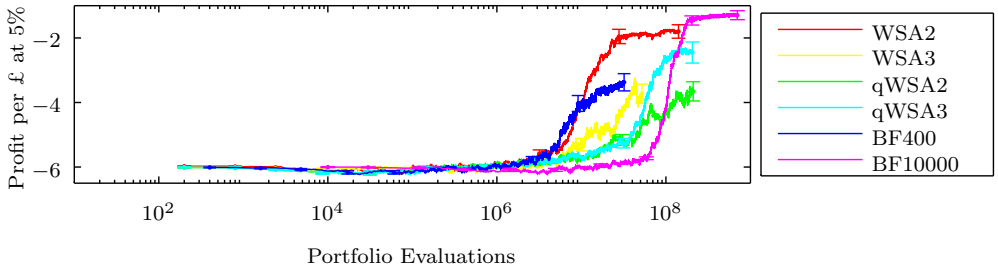


Figure 4.8: The above figure shows the average result for each algorithm. We see that qWSA2, BF400 and WSA3 all perform poorly. The other algorithms all obtain similar results. It is surprising that BF400 does not do well, but this underlines the challenge of naively selecting a suitable number of samples when optimising at a particular risk level. We would expect all of the (q)WSA algorithms to perform badly as compared to $V_D(0.05)$ as we hypothesised that they would in fact optimise $V_D^*(0.05)$. Consequently, it is more surprising that WSA2 and qWSA3 performed (relatively) well. We will look at this further in the later results.

tions. We see that, as predicted, the original estimator (from chapter 3 and section 4.3) appears to be significantly more noisy than the new estimator for $\hat{R}_1'$. We can see that the WSA algorithms actually optimise the system at a risk point significantly higher than 5%, leading to their bad results. In contrast we can see that both qWSA algorithms retain tight control of $\hat{R}_1'$ in that $\hat{R}_1'$ does not become too high or too low. It is important to note that in all cases $\hat{R}_1'$ remains above 3.5% and hence the algorithms did not inadvertently optimise $V_D^*(0.05)$ by optimising $V_D(R_T')$, where $R_T' < 0.03$.

Figures 4.11 and 4.12 dig deeper in to the performance of the (q)WSA algorithms in their optimisation of $V_D^*(0.05)$. We note that the results of the optimisations are actually bimodal. One mode represents the optimisation of $V_D^*(0.05)$ and the other $V_D(0.05)$. The majority of the optimisation runs actually optimise $V_D^*(0.05)$ (see figure 4.11). We see a low average for $V_D^*(0.05)$ in figure 4.9 because the downward change in $V_D^*(0.05)$ when $V_D(0.05)$ is optimised is significantly larger than the upward change when $V_D^*(0.05)$ is optimised (as one would expect). This means that when we take an average over $V_D^*(0.05)$ in figure 4.9 we misrepresent the majority of the samples.

We have successfully observed the bias to zero risk in a real world problem. The bias to zero risk approximation states that the (q)WSA algorithms, which aim to optimise $V_D(R_T)$, will in fact attempt to optimise $V_D^*(\hat{R}) = V_D(\hat{R}) - \hat{R}V_D'(\hat{R})$. We have also demonstrated the use of an alternative algorithm which allows for a much

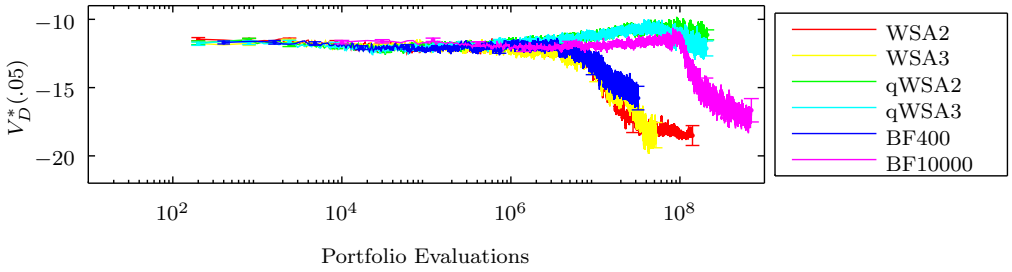
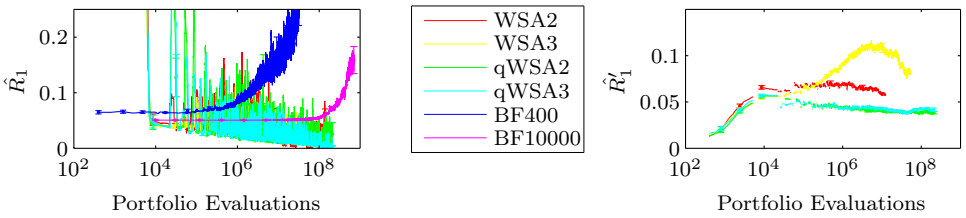


Figure 4.9: The above plot shows the average result for $V_D^*(R_T)$ for each algorithm. We see that both qWSA2 and qWSA3 perform well (as compared to brute force). However, we might have hoped to see more improvement in $V_D^*(0.05)$ as the chain progressed, rather than just a better performance when compared to brute force.

smaller number of samples to be used. Finally, we have developed new indicators for $\langle R_1 \rangle$ and $\frac{d\langle R_1 \rangle}{dn}$ and shown them to be more effective than the previous estimators when $k = 1$. They do however constrain how we perturb the samples ($\mathcal{X}$) used to estimate $V_D(R_T)$. In the next section we will look at an alternative approximation, which covers a larger variety of situations. This new approximation will lead to similar conclusions, but will not prove as elegant.



(a) We see that both brute force algorithms struggle to hold $\langle R_1 \rangle$ low. This would explain the poor performance of BF400. We further note that the estimators for $\langle R_1 \rangle$ are very noisy (as compared to $\langle R'_1 \rangle$). This is as predicted in section 4.5.4 and would explain the relatively poor performance of both WSA2 and WSA3 (with respect to $V_D^*(R_T)$).

(b) The figure shows that both WSA2 and WSA3 optimise the system at risk points well above 5%, potentially, explaining their bad performance. In contrast, qWSA2 and qWSA3 both maintain $\hat{R}'_1$ well. We additionally note that $\hat{R}'_1$ does not drop below 3.5% for either algorithm. This ensures that we did not perform comparatively well at optimising $V_D^*(0.05)$ simply by (unintentionally) optimising at a lower value of risk.

Figure 4.10: The above figures show how both the estimators of $\langle R_1 \rangle$ change over the course of the optimisations. As a reminder, $\langle R_1 \rangle$ is calculated using the results in chapter 3 and section 4.3, whereas $\langle R'_1 \rangle$ is calculated using the method found in section 4.5.4.

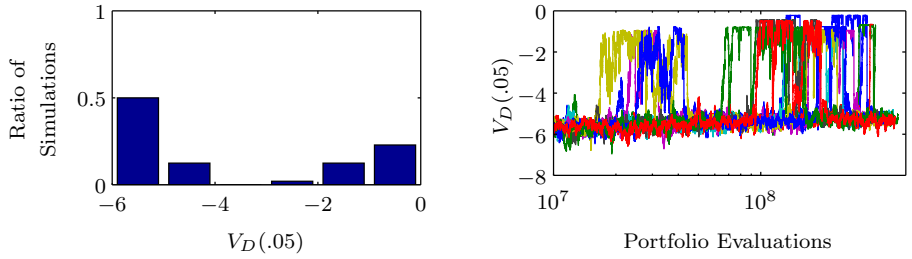


Figure 4.11: The above figures show that the final solution for $V_D(0.05)$ from the qWSA2 algorithm actually represents a bimodal distribution. The histogram (left) shows the final value of $V_D(0.05)$, over all 48 simulations. We see a mode at around -1 and -5. The plot on the right shows how $V_D(0.05)$ evolves over the chain for 10 independent runs. Several of the chains can be seen jumping between the two modes.

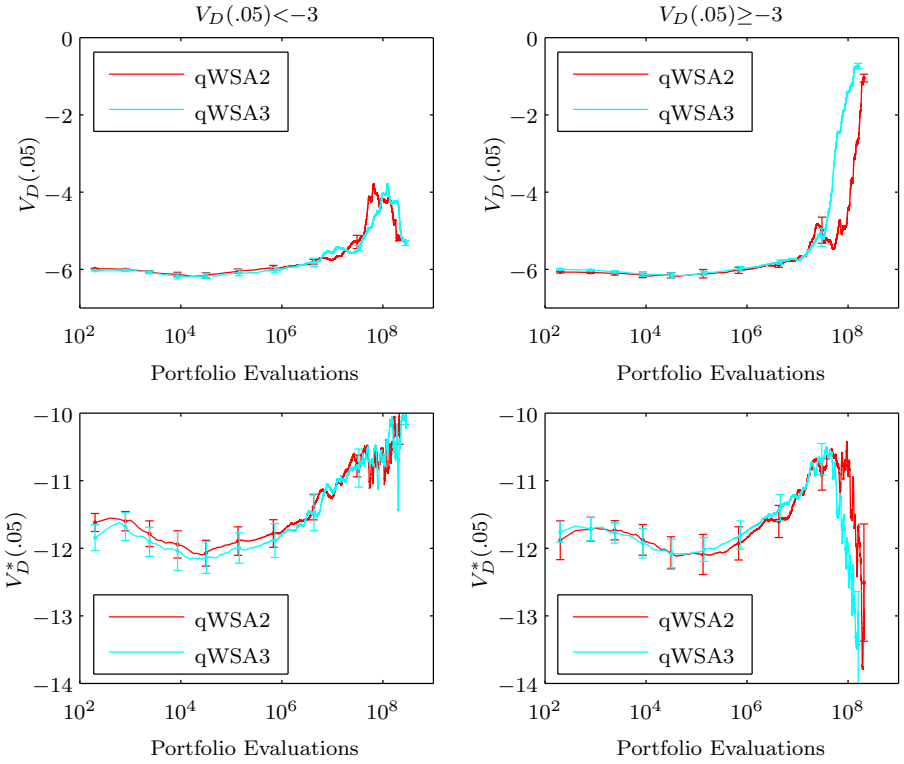


Figure 4.12: The above graphs show the comparative results of qWSA2 and qWSA3 for each mode, in terms of $V_D(0.05)$ and $V_D^*(0.05)$. From the left hand figures, we can see that over the course of the simulation qWSA2 shows significant improvement with respect to $V_D^*(0.05)$ when the final $V_D(0.05) < -3$. This is the dominant mode of the results. qWSA3 also shows improvement. The graphs on the right show how the second mode, where the final $V_D(0.05) \geq -3$, optimises $V_D(0.05)$ and consequently has significantly worse performance with respect to $V_D^*(0.05)$. There are significantly more results which find the solution where $V_D(0.05) < -3$ than where $V_D(0.05) \geq -3$.

4.6 Adapting k

4.6.1 The α Approximation

In this section we wish to move to a different paradigm from that of the previous section. In the previous section we deliberately held $k = 1$ in order to exploit a bias to zero risk and whilst this proved effective, it is still plausible that we really do wish to optimise a system at a specific percentile and not at a particular depth of sampling.

As previously discussed, in an unbiased system, we should set $k = \lfloor R_T n \rfloor$. We now consider a new variable, $s := \frac{k-1}{n}$. This is the level of risk implied by k and n. When there is no induced bias in the system $s = R_T$.

Using this variable we can find an alternative approximation to the chain. Given that the Poisson weights are

$$W_P(D|k, n, \beta) \propto \int_0^\infty \frac{1}{(k-1)!}(R_k n)^{k-1} e^{\beta V_D(R_k) - R_k n} dR_k,$$

We substitute $k = sn+1$ and apply Stirling's Approximation (2.2.4) on the factorial (which is really only a change in normalisation as independent of D) to get

$$\begin{aligned} W_P(D|k, n, \beta) &\propto \int_0^\infty e^{\beta V_D(R_k) - nR_k + sn\log(nR_k) - sn\log(sn) + sn} dR_k \\ &\propto \int_0^\infty e^{\beta V_D(R_k) + sn\left(1 - \frac{R_k}{s} + \log\left(\frac{R_k}{s}\right)\right)} dR_k. \end{aligned}$$

If we Taylor expand $1 - \frac{R_k}{s} + \log\left(\frac{R_k}{s}\right)$ around $\frac{R_k}{s} = 1$ we can see that $1 - \frac{R_k}{s} + \log\left(\frac{R_k}{s}\right)$ is approximately $-\frac{1}{2}\left(\frac{R_k}{s} - 1\right)^2$. Substituting back into the weight gives that

$$W_P(D|k, n, \beta) \propto \int_0^\infty e^{\beta V_D(R_k) - \frac{sn}{2}\left(\frac{R_k}{s} - 1\right)^2} dR_k.$$

If we now consider a steepest descent argument, we see that $n\left(\frac{R''}{s} - 1\right) = \beta V_D'(R'')$ for some R''. By substituting this back into the weight, and taking a linear approx-

imation we get that

$$W_{AS}(D|k,n,\beta) := e^{\beta\left(V_D\left(\frac{R''+s}{2}\right)\right)}. \tag{4.11}$$

This approximation suggests that our upward biasing by β may not have as significant an impact as we originally thought (at least to some approximation). We can use this result to better tune the size of sample we use; either directly, as an improved test for whether we should increase the sample size used to estimate $V_D(R_T)$, or indirectly by using this result to reduce k without changing the sampling level, to improve our estimate of $V_D(R_T)$.

4.6.2 An Example $V_D(R)$

We can consider how the error of this approximation changes for an example choice of $V_D(R)$. As before, we take $V_D(R)$ to be $\epsilon \log\left(\frac{R}{R_0}\right)$ (as discussed in section 4.5.2).

In this example we can find an expression for the weights in terms of s, n and β. We can then compare this result with the Poisson weight found in equation 4.6, to see how the error changes with respect to both β and n.

We need to find the value of R'' which the steepest descent argument specifies when $V_D(R) = \epsilon \log\left(\frac{R}{R_0}\right)$. Substituting this choice of $V_D(R)$ into $n\left(\frac{R''}{s} - 1\right) = \beta V_D'(R'')$ results in a quadratic expression in R''. We can then take the positive root as, by definition, $R'' \geq 0$. This gives $R'' = \frac{s+\sqrt{s^2+\frac{4\beta\epsilon s}{n}}}{2}$. Substituting this value of R'' back into equation 4.11 gives

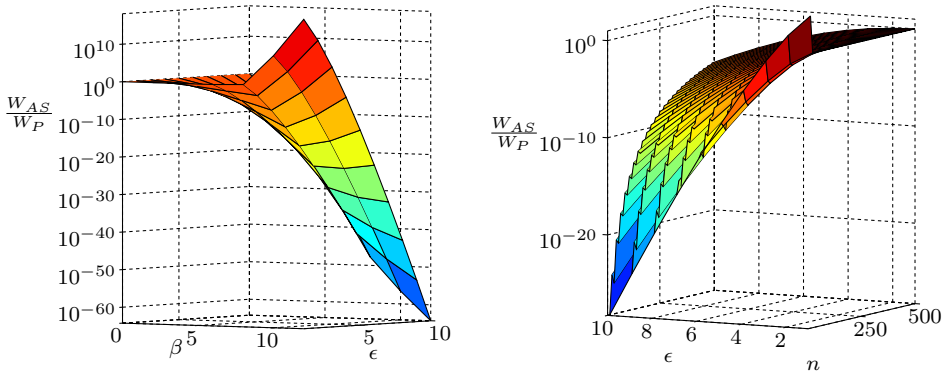
$$W_{AS}(D|k,n,\beta) \propto \left(\frac{s}{4R_0}\left(3+\sqrt{1+\frac{4\beta\epsilon}{ns}}\right)\right)^{\beta\epsilon}$$

where $W_{AS}(D,n,\beta)$ is the Markov weight for this chain calculated according to the weight defined in equation 4.11.

By applying a similar argument to earlier we can see that

$$\frac{W_{AS}}{W_P} = \left(\frac{sn\left(3+\sqrt{1+4\frac{\beta\epsilon}{n}}\right)}{4(\beta\epsilon+sn)}\right)^{\beta\epsilon}.$$

It should be clear from this expression that changing n has very little impact on the ratio between the different approximations of the weight. However, as with the bias to zero risk approximation, increasing β will increasingly bias large values of



(a) The ratio substantially increases as β increases. However, it is worth noting that this error is in the opposition direction to that of $\frac{W_P}{W_B}$ so there may be some cancelling.

(b) The ratio is almost entirely unaffected by n. Substantially increasing n is not sufficient to recover from the biasing effects of $e^{\beta V_D(R_k)}$.

Figure 4.13: The above figures show how the ratio between the Poisson and floating k weights change over time with respect to n and β. $R_0 = 1$. In figure 4.13a $n = 40$ and in figure 4.13b $\beta = 5$. The weights are numerically integrated over $\epsilon \in [1, 10]$ (for fixed n and β).

ϵ. If the design space being considered includes various different choices of ϵ then the normalised distribution across $\mathcal{D}$, as calculated using this approximation, will become increasingly distorted. Figures 4.13a and 4.13b show this visually.

4.6.3 Algorithmic Implications

As discussed at the end of section 4.6.1, the above approximation suggests we could relax the rate at which we increase n, choose some $k < R_T n$, or do a combination of both to help ensure we actually optimise a given system at R_T. We use the algorithms proposed in sections 3.6 and 4.5.4, with the only refinement being that we increase n when $\frac{1}{2}\left(\hat{R}_k + \frac{k}{n}\right) > R_T$. Clearly if $k = R_T n$, this reduces to the previous case where n increases if $\hat{R}_k > R_T$. In the algorithm variants that use $\frac{d\langle \hat{R}_k \rangle}{dn}$ to adjust n, we also only increase n when $\frac{1}{2}\left(\hat{R}_k + \frac{k}{n}\right) > R_T$ instead of increasing n when $\hat{R}_k > R_T$.

To avoid the degradation when $k = R_T n$ we again hold $k = 1$. If we assume n is large, then our condition for increasing n becomes $\hat{R}_k > 2R_T$. This suggests that we will actually optimise the system at $\frac{R_k}{2}$, when $k = 1$. It is worth noting that in this example we can no longer expect $\left(\frac{R_T}{s} - 1\right)$ to be small (an assumption made in

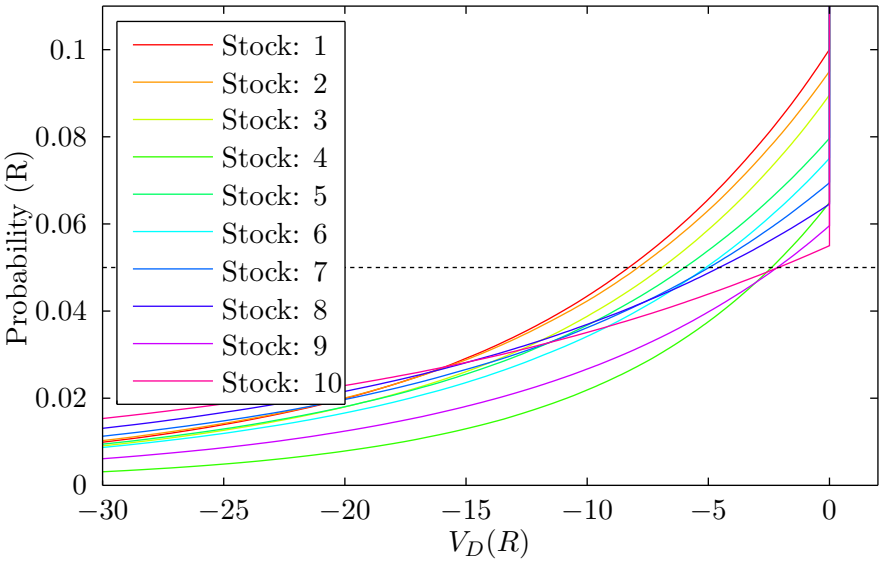


Figure 4.14: The above figure shows the tail of the c.d.f. for each (single) option our optimisation process could select. The optimisation process could also select fractional amounts of each stock, as long as it sells exactly £1 worth. It should be clear in this context that the bias to zero risk has minimal explanatory power. For the ten individual cases plotted, the best option is to select any stock if optimising $V_D(0.1)$ and stock 10 if optimising $V_D(0.05)$ $\left(= V_D\left(\frac{1}{2} \times 0.1\right)\right)$. There could potentially be a better solution to both problems which combines different stocks.

the approximation) as we are explicitly expecting $s > R_T$. However this expression is still of $O(1)$ and hence we would still expect to see the same qualitative behaviour that our approximation dictates. We will discuss this further in the discussion at the end of the chapter (section 4.8).

4.6.4 Results

In this section, we wish to evaluate the above approximation when $k = 1$, using a problem in which the bias to zero risk approximation does not hold. We continue to consider a user who wishes to sell £1 of call options. We assume that these options can be infinitely divided and that there is an open market on which they can be traded. The options prices were calculated using Black Scholes [Glasserman, 2004]. As in the previous chapter the stocks are modelled using a multivariate skewed-t distribution. Table 4.2 shows the stocks and the strike prices used.

We wish to find a portfolio containing some combination of the stocks in figure 4.14 that optimises the risk at 5%. We will do this by optimising $V_D(0.1)$

Stock	Current Price (p)	Strike Price (p)
ABF.L	1550.0	1569.0
ADN.L	381.6	390.7
ADM.L	1198.0	1228.3
AGK.L	1749.0	1785.2
AMEC.L	1035.0	1060.5
AAL.L	2000.5	2060.5
ANTO.L	1322.0	1366.6
ARM.L	799.5	821.6
AZN.L	2985.0	3027.7
AV.L	388.4	401.9

Table 4.2: The above table defines the stock and strike prices used for the call options to be sold. (The current price is the price of the stocks on Monday, 7 January 2013.) As before, the returns model used was fitted to daily returns from 2012.

using algorithms 3.2 and 4.1. By the above approximation, this should be equivalent to optimising at 5% risk. As can be seen from the figure, the descriptive power of the bias to zero risk approximation is limited in this context, as a system optimised using $V_D^*(0.1)$ would just result in an almost random set of stocks (which excluded stock 1). This is unsurprising given that the bias to zero risk approximation implicitly requires $V_D'(0.1) > 0$, which is not always the case in this problem.

We use the same simple style of cooling schedule as in the previous results sections. We increase β by 0.125 after every 200 accepted perturbations of D. β is initially set to 1. The design space is $\mathcal{D} = [0, 1]^9$. 48 identical optimisations were run for each method, the results were then averaged, and the standard error calculated. D was perturbed in the same way as in previous examples (3.7).

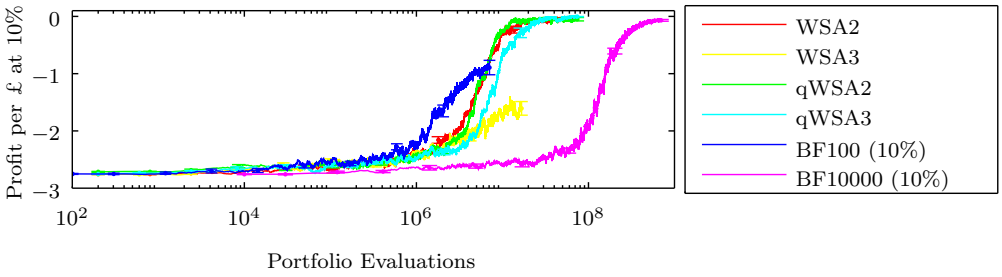


Figure 4.15: The above figure shows the convergence of the different algorithms, versus their value at the 10% risk point. This is the point the algorithms nominally optimised. The (q)WSA algorithms generally find a better solution for a given fixed number of portfolio evaluations. We note that the brute force algorithm that one would nominally use for such a problem is BF100, which in this scenario appears to significantly under perform the other algorithms. WSA3 also fails to find a good solution. This is due to the estimator for $\frac{d\langle R_1 \rangle}{dn}$ not converging correctly.

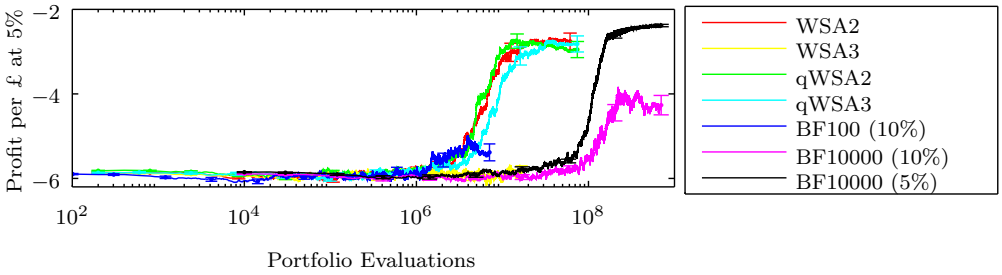


Figure 4.16: The above figure shows the convergence of the different algorithms, in terms of their value at the 5% risk point. This is the point we expected the (q)WSA algorithms to actually optimise. The (q)WSA algorithms generally find a better solution for a given fixed number of portfolio evaluations, but do not manage to reach the same final solution as a brute force algorithm which seeks to directly optimise the 5% risk point (BR10000 (5%)). The (q)WSA algorithms final solutions (except for WSA3) are better than either of the brute force algorithms that optimised the 10% risk point. This demonstrates the above bias is occurring and that the approximation is valid.

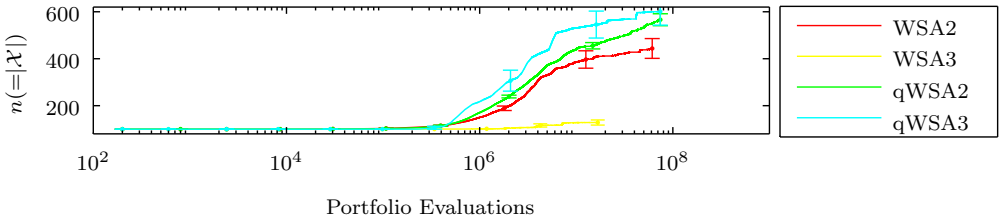


Figure 4.17: In this figure we can see the number of samples each algorithm used to estimate $V_D(R)$ as it progressed. Since BF100 and BF10000 remain fixed at their initial values we do not show them here. As described above, we can see that WSA3 fails to significantly increase n. This is because the estimator for $\frac{d\langle R_1\rangle}{dn}$ is not converging correctly. All the other algorithms show comparable results.

Figures 4.15, 4.16 and 4.17 show the results and demonstrate the expected bias occurring. The new algorithms generally all perform well, except for WSA3, in which the indicator for $\frac{d\langle R_1\rangle}{dn}$ struggles to converge. Figure 4.15 shows that the remaining (q)WSA algorithms all perform very well at the 10% risk point and that they reach the same solution as BF10000 (a brute force algorithm which uses 10, 000 samples per step). The figure also shows that the naive choice for brute force in this case, BF100, fails to converge to a good solution. Figure 4.16 shows how the same algorithms performed at the 5% risk point (even though they were nominally optimising at 10%). We see that all of the (q)WSA algorithms (except for WSA3) outperformed BR10000 (10%) at this point (both in terms of work and final result). This strongly corroborates that the above bias exists and that the resulting approximation is a reasonable one. That said, a brute force algorithm directly optimising at the 5% risk point does obtain a better final solution (BF10000 (5%)), although it takes almost two orders of magnitude of additional moves to achieve this.

Figure 4.17 shows the number of samples required to achieve the optimisations. It is worth noting that the number of samples used by the (q)WSA is an order of magnitude less than is required for BF10000. This in itself might make the algorithms highly desirable in certain circumstances, either when it is very expensive, or simply impossible, to obtain large samples of X. We previously discussed this at the end of section 3.7 and will return to this topic in the final discussion in chapter 8.

In this section we have observed that a system which has $k < nR_T$ may not be as affected by the upward pressure induced by $e^{\beta V_D(R_T)}$ as when $k = nR_T$. We found that $W(D) \propto e^{\beta\left(V_D\left(\frac{R''+s}{2}\right)\right)}$ (equation 4.11) and then (successfully) numerically

tested it for the case when $k = 1$. We also note that, as in section 4.5, the new indicators appear to provide better estimates of $\langle R_k \rangle$ and $\frac{d\langle R_k \rangle}{dn}$ (at least when $k = 1$) and hence allow for more robust algorithms in this case.

4.7 System Scaling

In this section we study the expected scaling relation between the system parameters β and n. Previously we have calculated several key results to allow us to correctly increase n as β increases (as part of the annealing process). However, beyond these estimators and empirical observations we have not investigated any possible scaling relations. We use the results in this section to confirm our previous intuition.

In order to study this scaling relation we use a class of $V_D(R)$ which encompasses our current test case. We assume that $\frac{dR}{dV} \propto e^{-|V|^p}$. By application of corollary 2.4.3 we see that

$$R \propto \frac{e^{-|V|^p}}{|V|^{p-1}} \tag{4.12}$$

for $|V| >> 1$. Furthermore, we note that in this example $V_D'(R) \propto e^{|V|^p}$ (found by inverting $\frac{dR}{dV}$).

In section 3.3 we found the weight of the chain, for fixed β and n, to be

$$W(D) = \int_0^1 dR_k \binom{n}{k} k R_k^{k-1} (1 - R_k)^{n-k} e^{\beta V_D(R_k)}.$$

By applying a steepest descent argument to this (akin to the argument made in section 4.5) we find

$$V_D'(R_k) = \frac{1}{\beta} \left(\frac{(n-1)R_k - k + 1}{R_k(1 - R_k)} \right).$$

If we substitute our assumption for $V_D'(R_k)$ (from above) and take the log of both sides we get

$$|V|^p \propto \ln \left(\frac{1}{\beta} \left(\frac{(n-1)R_k - k + 1}{R_k(1 - R_k)} \right) \right).$$

Substituting the above result into equation 4.12 results in

$$R_k \propto \beta \left(\frac{(n-1)R_k - k + 1}{R_k(1-R_k)} \right)^{-1} \left(\ln \left(\frac{1}{\beta} \frac{(n-1)R_k - k + 1}{R_k(1-R_k)} \right) \right)^{-\frac{p-1}{p}}.$$

This can be rearranged to find

$$\beta \propto \left(\frac{(n-1)R_k - k + 1}{(1-R_k)} \right) \left(\ln \left(\frac{1}{\beta} \frac{(n-1)R_k - k + 1}{R_k(1-R_k)} \right) \right)^{\frac{p-1}{p}}.$$

From this relation we can see that any increase in β would need to be supported by an increase in n or an increase in R_k (as we expect nR_k to be much larger than k) since the impact of the logarithm term will be very limited. Equivalently this relation suggests that an increase in β that is not supported by a suitable increase in n will drive up R_k. The choice of k does not have a major impact on the scaling relation.

This increase in R_k will mean we are no longer optimising our random variable at the correct level of risk. Furthermore, it will increase the bias in the sampling of $\mathcal{X}$ towards the current choice of D. This in turn could have a potential impact on the acceptance rates of D, slowing the optimisation. This reduction in acceptance rates is because it is likely that it will become harder for a perturbation of D to be accepted as $\mathbf{X}$ becomes more tailored to the current (specific) choice of D.

These analytical observations match our previous empirical results and intuitive understanding. Unfortunately they do not provide any significant further understanding beyond this particular choice of $V_D(R_k)$, and so it is unclear how they could be used to improve the algorithm as it stands. The effects described above could have significantly more impact if both D and $\mathbf{X}$ were perturbed simultaneously.

4.8 Discussion

In this chapter we have considered two different physical approximations and various properties of the (thermal) system created when simulated annealing is used to solve a simple risk problem. The two approximation we considered for the chain were $e^{\beta V_D^*(\hat{R})}$ where $V_D^*(\hat{R}) = V_D(\hat{R}) - \hat{R}V_D'(\hat{R})$ and $\hat{R}$ was the dominant quantile explored (section 4.5); and $e^{\beta V_D\left(\frac{R_k+s}{2}\right)}$ where $s = \frac{k}{n}$ (section 4.6).

We initially analysed the problem from a physical perspective to calculate an approximation to the resulting Markov chain. We then layered two other approximations on top of this to gain insight into how the system operated and then

used these insights to adapt the simulated annealing algorithm. For both of these approximations we considered the error induced by them (for an example choice of $V_D(R)$) and successfully demonstrated the effect of the algorithmic changes. In the final section we considered some of the scaling properties of the system which supported our results.

It should be clear, especially since the two approximations discussed describe the same system, that their interpretations are similar. The first approximation, which showed a bias to zero risk, was based on a steepest descent argument and revealed that an optimisation process would be biased towards solutions that had the smallest (at least as perceived by the chain) zero risk point. The second approximation built on this argument by applying a linear approximation to $V_D(.)$ (and expanding the argument more clearly to larger choices of k). This suggests that the optimisation process would equivalently optimise the system at $V_D\left(\frac{R_k}{2}\right)$. Whilst the second approximation is very sensitive to the actual $V_D(.)$ in its interpretation, it should be clear that optimising $V_D^*(\hat{R})$ is likely to be equivalent to optimising $V_D(R')$, for some choice of $0 < R' < R_T$. This was explicitly seen to be the case for our choice of $V_D(.)$ in section 4.5.2, where $V_D^*(\hat{R}) = V_D\left(\frac{R_T}{e}\right) \approx V_D\left(\frac{R_k}{2}\right)$.

In section 4.4 we also found new estimators for $\langle R_k \rangle$ and $\frac{d\langle R_k \rangle}{dn}$. The various numerical results demonstrated that these new estimators appeared to give more reliable results than the estimators found in the previous chapter. However, this improvement came at a cost: to use them efficiently we had to constrain how we perturbed $\mathbf{X}$. This meant we were unable to use them to measure the performance of the brute force algorithms. It may also slow down the optimisation process in a highly complex problem, where it becomes important to change the sample set $\mathbf{X}$ by basing new elements of $\mathbf{X}$ on current elements of $\mathbf{X}$, rather than just taking new draws from X.

During the course of this chapter we have used an example choice of $V_D(.)$ which is akin to an exponential distribution to evaluate the various approximations. It would have been nice to consider other choices of $V_D(.)$ (similar to when we considered how the system might scale in section 4.7). Unfortunately we would not have been able to perform the various integrations and error calculations as easily if we had chosen a $V_D(.)$ which represented a normal distribution. We could have calculated such a $V_D(.)$ in two ways. If we were to assume $\frac{dR}{dV}$ was normally distributed then we would have had $\frac{dR}{dV} \propto V_0^{-1} e^{-\frac{1}{2}V^2}$. By applying a similar argument to that of proposition 2.4.3, we could integrate this to find that $R \approx (2V_0 V)^{-1} e^{-\frac{1}{2}V^2}$. Rearranging this expression for V gives
$V_D(R) = \sqrt{2}\sqrt{-\log(2V_0 R) - \log(V_D(R))}$. We could now iteratively solve for $V_D(R)$,

however the resulting approximation would not be malleable enough to calculate analytically the various weights involved. A second approach would be to directly approximate the inverse error function. This could (very approximately) lead to a choice of $V_D(R)$ such that $V_D(R) = \epsilon \log\left(\frac{R-R^2}{R_0}\right)$. This approximation would have allowed us to calculate some, but not all, of the weights we have considered.

We also note that we could formally have proved that as n increases $\langle R_k \rangle \to \frac{k}{n} = R_T$ (at least for a sensible choice of $V_D(.)$). However, such a proof is unlikely to have yielded any further information on the rate of convergence of the approximations unless the exact form of $V_D(.)$ was known. At most, such a proof would show that the various approximations (for certain classes of $V_D(.)$) do indeed get arbitrarily close to the true weights, for a sufficiently large choice of n, rather than coinciding only at the limit. It should be possible to see that this is at least approximately true by observation.

We could have further considered an array of physical properties of the system, including both the energy and the intensivity and extensivity of the various variables. However it is unclear, beyond mere curiosity, the impact such results would have. If we consider the system in terms of entropy, it should be clear that increasing either n or β reduces the entropy, and hence the energy, of the implied thermal system.

In the course of this chapter we have shown two different approximations of the annealed system. The first, which showed a bias to zero risk, was shown to have a direct impact on a real world numerical problem and represented a significant paradigm shift. The second, which showed we could assume that the attempted level of risk, R_T, still had some impact on the risk, allowed us to further relax our annealing algorithm to reduce work, but still achieve solid results.

We will discuss the ideas from both this and the preceding chapter further in the final discussion.

Chapter 5

Further Extensions: Optimising Quantile Risk

5.1 Introduction

In this short chapter we build on the fundamental ideas developed in chapters 3 and 4. These ideas were used to develop an algorithm to optimise a system at a given level of risk. In this chapter we draw ideas from both of the previous chapters. In chapters 3 and 4 we defined the Markov chain (and ensemble) we expect our optimisation method to follow. We now push these ideas further so that we can extend the algorithm to cover a wider variety of problems.

We consider two possible extensions to the algorithm to allow it to tackle further use cases. The first extension allows for only a risk reward optimisation and the second for optimisation at multiple quantiles.

The risk reward optimisation allows for a Markovitz style optimisation. In this scenario the user specifies how much extra expected gain they require before accepting further predicted risk in the solution. This quantity is generally a linear ratio and is known as the risk reward ratio. We show that analogous proofs to those in chapters 3 and 4 can be found along with some basic numerical results.

The multi quantile optimisation method allows the user to specify two quantiles to optimise at and the weighting to give each one. This could be the median and some specified risk level to give a similar optimisation as to the previous method. Alternatively, it could be used to tightly control the shape of the risk tail of the resulting random variable.

We conclude the chapter in section 5.5 and in context with the rest of the work in chapter 8.

5.2 Problem Definition

In this chapter we wish to consider the optimisation of
$\arg\max_D \int_0^1 V_D(R)w(R)dR$, for some choice of $w(R)$. If we take $w(R) = 1$ when $R = R_T$ and 0 elsewhere then we recover the case where we wish to optimise a single quantile. $w(R)$ could represent a wide range of choices including the mean of F_D, the interquartile range of F_D, or even allow for the detailed specification of part (or all) of a desired risk curve. Ideally, we would like to consider any choice of $w(R)$, however, practicalities will require us to only consider the optimisation of the mean alongside some quantile, or just two separate quantiles.

Mathematically, we consider the following two choices of $w(R)$;

$$w_1(R) = \frac{1}{n} + \alpha\delta\left(R_T - R\right) \text{ and}$$
$$w_2(R) = \delta\left(R - r_1\right) + \alpha\delta\left(R - r_2\right),$$

where α, R_T, r_1 and r_2 are parameters and $\delta(x)$ is the Dirac delta function. Thus a system using $w_1(R)$ or $w_2(R)$ will find $\arg\max_D \mu_D + \alpha V_D(R_T)$, where μ_D is the mean of F_D, and $\arg\max_D V_D(r_1) + \alpha V_D(r_2)$ respectively.

5.3 Adding a Mean

In this section we add the mean to our optimisation. This addition turns out to be relatively simple. As discussed in the previous section (5.2), we wish to solve $\arg\max_D \mu_D + \alpha V_D(R_T)$ for a specified choice of R_T and α. If we assume that μ_D can be accurately calculated from the same samples used to estimate $V_D(R)$, without any bias, then the calculations from sections 3.3 and 3.4 largely hold without any significant adjustments. While this assumption is not strictly true, it should be provide a reasonable approximation during the annealing process. Indeed, this assumption is often implicitly used whenever a mean value (which is somehow estimated at each step) is optimised.

The relative density for a given design, D, extends to include the mean, giving

$$\pi_m(D, R_k) = \binom{n}{k} k R_k^{k-1}(1 - R_k)^{n-k} e^{\beta(\mu_D + \alpha V_D(R_k))} \text{ and}$$
$$\pi_m(D) = \int_0^1 dR_k \binom{n}{k} k R_k^{k-1}(1 - R_k)^{n-k} e^{\beta(\mu_D + \alpha V_D(R_k))}.$$

The results to estimate $\langle R_k \rangle$ and $\frac{d\langle R_k \rangle}{dn}$ then become

$$\langle R_k \rangle_{k,n} \leq \frac{\frac{k}{n-k}\left\langle e^{\beta\alpha(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}}{1+\frac{k}{n-k}\left\langle e^{\beta\alpha(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}} \text{ (see 3.4.6) and}$$

$$\frac{d\langle R_k \rangle}{dn} \approx \frac{\left(\frac{k}{n-k}\left\langle e^{\beta\alpha(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}\right)^2}{1+\frac{k}{n-k}\left\langle e^{\beta\alpha(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}} - \frac{\frac{k(k+1)}{(n-k)(n-k-1)}\left\langle e^{\beta\alpha(V_D(R_{k+2})-V_D(R_k))}\right\rangle_{k,n}}{1+\frac{k}{n-k}\left\langle e^{\beta\alpha(V_D(R_{k+1})-V_D(R_k))}\right\rangle_{k,n}} \text{ (see 3.12).}$$

5.3.1 Algorithmic Implications

Since the estimators do not include $e^{\beta\mu_D}$ we can largely reuse algorithm 3.2 for the optimisation. The key changes we need to make are to adjust the acceptance probability and the estimators for $\langle R_k \rangle$. The acceptance probability, p, becomes

$$p = \min\left(1, e^{\beta_i(\tilde{\mu}_D - \mu_D + \alpha(\tilde{V}-V))}\right)$$

where μ_D is the mean calculated using $\mathbf{X}$ and D and $\tilde{\mu}_D$ is the mean calculated from $\tilde{\mathbf{X}}$ and $\tilde{D}$. The estimators for $\langle R_k \rangle$ and $\frac{d\langle R_k \rangle}{dn}$ in the algorithm are replaced with those given above (α added).

We also note that we can use an adaptation of algorithm 4.1. The estimator for $\langle R_k \rangle$ in this algorithm continues to hold, regardless of the optimisation target. We do however adjust the algorithm slightly to use different values of k (rather than just $k = 1$). We adjust k in the same way as in algorithm 3.2. We also now perform just 100 perturbations of $\mathbf{X}$ per design perturbation. (This is the same as algorithm 4.1, however, in algorithm 4.1 we perturbed $\mathbf{X}$ n times as k was fixed at one.) We label this adaptation qSA2 (where we just use the estimator for $\langle R \rangle$ to control n). We cannot use the estimator for $\frac{d\langle R_k \rangle}{dn}$ in algorithm 4.1 in this context as it will no longer be correct.

A table summarising the basic algorithms used for one quantile (before the above adaptation) can be found in appendix A.

5.3.2 Results

We test the above algorithms and related assumptions using a simple portfolio. The particular stocks and models used are the same as those in section 3.7. We require the optimisation to select the best combination of ten stocks to buy. We consider the best portfolio to be one which maximises $\mu_D + \alpha V_D(0.05)$. We consider when

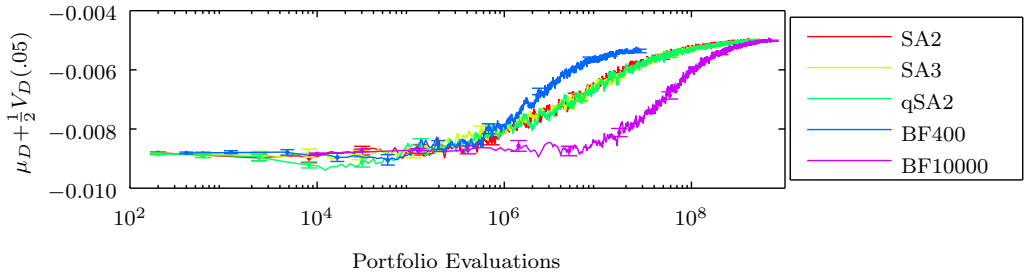


Figure 5.1: The above figure shows the optimisation results when $\alpha = 0.5$. We see very similar results to chapters 3 and 4. The (q)SA algorithms all find a better result as compared to the naive choice of brute force algorithm (BF400), but do take slightly longer to run. They appear to find the most optimal solution (which is likely to be found by BF10000).

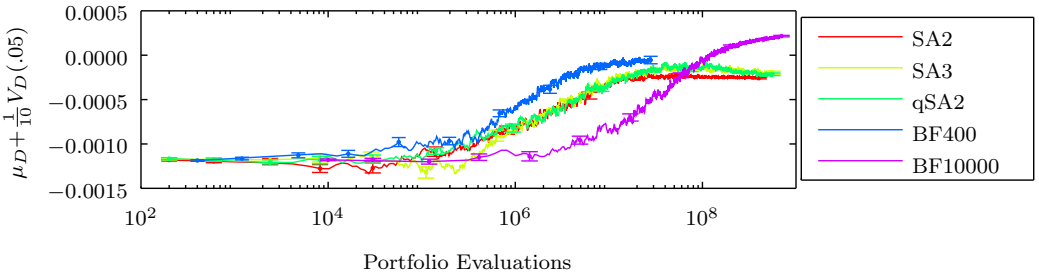


Figure 5.2: The above figure shows the optimisation results when $\alpha = 0.1$. The (q)SA algorithms fail to find a good solution as compared to both the naive choice of brute force algorithm (BF400) and the long run brute force algorithm (BF10000).

$\alpha = 0.5$ and $\alpha = 0.1$. We test these two values of α because we suspect that when α becomes small, our optimisation methods will become more sensitive to the error induced by the estimation of μ_D. When $\alpha = 0.5$ the optimisation will be dominated by $V_D(0.05)$. At $\alpha = 0.1$ both terms in the optimisation will have broadly equal magnitude (due to the relative magnitude of μ_D and $V_D(0.05)$).

As in previous chapters the stocks are modelled using a multivariate skewed-t distribution. We also use the same simple style of cooling schedule as in the previous results sections. When $\alpha = 0.5$, we increase β by 250 after every 200 accepted perturbations of D. β is initially set to 250. When we reduce α to 0.1, we increment β by 500 and start at $2,000$. As in previous cases, the design space is $\mathcal{D} = [0, 1]^9$. 48 identical optimisations were run for each method, the results were then averaged, and the standard error calculated. D is perturbed as in section 3.7.

Figures 5.1 and 5.2 show the results. These results are largely as one might expect. We do well when the $V_D(R_T)$ is dominant in the optimisation (when $\alpha = 0.5$) but when α gets smaller we struggle to find a good solution. This is probably due to a bias induced by the optimisation in the estimation of μ_D.

There is no difference in performance (in either case) between qSA2 and SA2.

In the next section we attempt to solve this estimation bias of μ_D by optimising using the median.

5.4 Two Quantile Optimisation

In this section we aim to rectify the issue from the last section that the estimation of μ_D is likely to be biased. We consider the case when we wish to optimise two risk points. We will mainly consider the median (representing an average) and some other risk point. Mathematically, we wish to optimise $\arg\max_D V_D(r_1) + \alpha V_D(r_2)$ for parameters $r_1, r_2 \in [0, 1]$ and $\alpha \in \mathbb{R}$. We will normally take $r_1 = \frac{1}{2}$.

5.4.1 Markov Chain

In this section we find the Markov chain the above optimisation process will follow (at least when β and n are fixed). The argument used is an adaptation of that found in sections 3.3 and 3.4 and hence is only briefly covered here.

In this new chain we need to consider the joint probability density of the data points which we use to estimate $V_D(r_1)$ and $V_D(r_2)$. This is because, by using a similar argument to that of section 3.3, we find for $j < k$ (and fixed β, n) that the relative probability density of a specific design, D, is

$$\pi_2(D) = \int_0^1 \int_0^1 \zeta_{j,k}(R_j, R_k) e^{\beta(V_D(R_k) + \alpha V_D(R_j))} dR_j dR_k \tag{5.1}$$

where $\zeta_{j,k}(R_j, R_k)$ is the relative joint probability density of the points in positions j and k of n ordered samples having values R_j and R_k respectively.

In order to calculate this joint density, we note that $\zeta_{j,k}(R_j, R_k) = \zeta_{j,k}(R_j|R_k)\zeta_k(R_k)$. We know that $\zeta_k(R_k) = \binom{n}{k} k R_k^{k-1}(1 - R_k)^{n-k}$ (see equation 3.8). By definition $\zeta_{j,k}(R_j|R_k)$ is the density of the j^{th} sample (of $k - 1$ samples) having value $R_j \in [0, R_k)$ (we still assume $j < k$). This means that the original argument used in section 3.3 still holds, except that in this case n should be replaced with $k - 1$, R rescaled to $\frac{R_j}{R_k}$ and the expression renormalised. This results in $\zeta_{j,k}(R_j|R_k) = \binom{k-1}{j} j \left(\frac{R_j}{R_k}\right)^{j-1} \left(1 - \frac{R_j}{R_k}\right)^{k-1-j} \frac{1}{R_k}$. Combining both

of these expressions gives that

$$\zeta_{j,k}(R_j, R_k) = \binom{k-1}{j} j \left(\frac{R_j}{R_k}\right)^{j-1} \left(1 - \frac{R_j}{R_k}\right)^{k-1-j} \frac{1}{R_k} \binom{n}{k} k R_k^{k-1} (1 - R_k)^{n-k}.$$

The above expression can then be arranged and the assumption that $j < k$ explicitly added to find that

$$\zeta_{j,k}(R_j, R_k) = \begin{cases} \binom{k-1}{j}\binom{n}{k} jk R_j^{j-1} (R_k - R_j)^{k-1-j} (1 - R_k)^{n-k} & j < k \\ 0 & otherwise \end{cases}. \quad (5.2)$$

Substituting this expression back into equation 5.1 would give a complete expression for the weight of a given design.

Similarly to chapter 3, we can find expressions which allow us to estimate the values of $\langle R_k \rangle$ and $\langle R_j \rangle$ which the chain is actually exploring. These are helpful because they will allow us to monitor the optimisation process and ensure that we are optimising the correct risk points. This is a concern because we expect the $e^{\beta(V_D(R_k)+\alpha V_D(R_j))}$ term in the density (weight) to induce a bias towards higher valued samples for high β and α and hence push up $\langle R_k \rangle$ and $\langle R_j \rangle$ if n is not sufficient.

A similar relation as to the single quantities for these densities can also be found (see 5.2); these are

$$\zeta_{j,k}(R_j, R_k) = \frac{j}{k-1-j} \frac{R_k - R_j}{R_j} \zeta_{j+1,k}(R_j, R_k) \text{ and}$$
$$\zeta_{j,k}(R_j, R_k) = \frac{k-j}{n-k} \frac{1 - R_k}{R_k - R_j} \zeta_{j,k+1}(R_j, R_k).$$

Furthermore, we can find additional expectation relationship for the resulting chain. These are given by Given fixed β, α, n, k and j we have that

$$\left\langle e^{\beta\alpha(V_D(R_{j+1})-V_D(R_j))} \right\rangle = \frac{k-1-j}{j} \left\langle \frac{R_j}{R_k - R_j} \right\rangle \text{ and}$$
$$\left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))} \right\rangle = \frac{n-k}{k-j} \left\langle \frac{R_k - R_j}{1 - R_k} \right\rangle.$$

The expectations are taken over an ensemble with weights as given by equation 5.1. The conditions and proof are akin to that of theorem 3.4.2.

Unfortunately, the above expressions are not immediately helpful. To make these expressions more usable, we assume that we can separate the expectations into terms of $\langle R_k \rangle$ and $\langle R_j \rangle$. This allows us to solve the two equations to find estimators

for $\langle R_k \rangle$ and $\langle R_j \rangle$. This assumption is hard to justify mathematically, however we shall see in the results that the resulting estimators do prove to be effective. It is plausible that an argument could be made stating that the random variables of concern become independent in the limit of $n \to \infty$ and hence that this assumption holds in the limit of large n. Making this assumption gives, for fixed β, α, n, k and j, that

$$\langle R_k \rangle \approx \frac{B(1+A)}{B(1+A)+1} \text{ and} \tag{5.3}$$

$$\langle R_j \rangle \approx \frac{A \langle R_k \rangle}{1+A} \tag{5.4}$$

where

$$A = \frac{j}{k-1-j} \left\langle e^{\beta\alpha(V_D(R_{j+1})-V_D(R_j))} \right\rangle \text{ and}$$
$$B = \frac{k-j}{n-k} \left\langle e^{\beta(V_D(R_{k+1})-V_D(R_k))} \right\rangle .$$

It has not been possible to find estimators for $\frac{d\langle R_k \rangle}{dn}$ or $\frac{d\langle R_j \rangle}{dn}$ (unlike in chapter 3). This means we will have to increase n more arbitrarily when R_k or R_j drop too low. The consequence of this is that we are less able to track the minimum required n for each choice of β. We will instead use (marginally) more samples than needed for the majority of choices of β. That said, we would still expect to see significant improvement over a brute force algorithm.

We will numerically test these estimators in the next section (5.4.2) before using them to develop a new algorithm in the following sections (5.4.3 and 5.4.4).

5.4.2 Numerical Convergence of Estimators

In this section we investigate the convergence and accuracy of the estimators defined in the previous section. For the estimators to be useful in a potential algorithm they must converge quickly and not over a long time. We will also take note of any features that might be relevant when constructing an algorithm.

We evaluate the estimators using a simple, one dimensional, model. As in section 3.5 we take $V_D(.)$ to be the inverse cumulative distribution function of the normal distribution. This implicitly means that there is only one possible choice of design.

We expect to exhibit the same two tensions as in the case when we only optimise on one quantile. The $e^{\beta(V_D(R_k)+\alpha V_D(R_j))}$ term will seek to drive up R_k and R_j, whereas the samples will seek to keep R_k and R_j at the correct values.

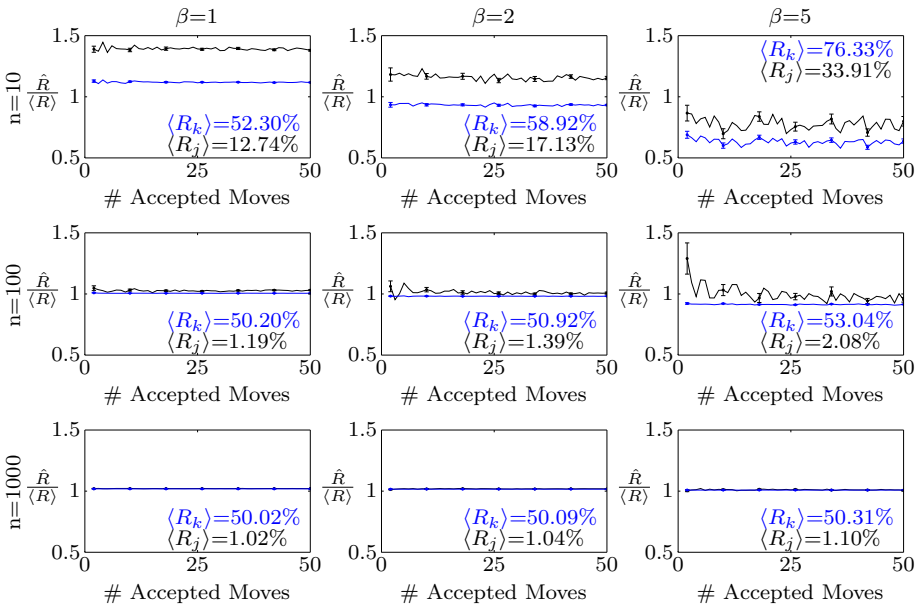


Figure 5.3: The above charts show the accuracy and convergence of the estimators for $\langle R_k \rangle$ and $\langle R_j \rangle$ as defined in equations 5.3 and 5.4. $\hat{R}_k$ and $\hat{R}_j$ denote the estimators. $\langle R_k \rangle$ and $\langle R_j \rangle$ denote the true expected value of R_k and R_j as calculated explicitly using the known $V_D^{-1}(V) = \Phi(V)$. j was set to $\max(1, 0.01n)$ (targeting 1% risk when $n \geq 100$) and $k = \lfloor 0.5n \rfloor$. In all cases $\alpha = 0.5$. The estimator shows good convergence for most choices of n and β. The estimators appear to lose accuracy when the system becomes too dominated by the exponential term (for example, when $n = 10$ and $\beta = 5$), however we should not be operating in this regime as it will also have high expected R_k and/or R_j. The graphs also (unfortunately) show that these estimators are not necessarily an upper bound on $\langle R_k \rangle$ and$\langle R_j \rangle$.

Figure 5.3 shows the convergence of the various estimators. The estimators appear to converge quickly in all cases, however their accuracy suffers when β is high and n is low. Fortunately, we do not intend to operate in this regime and hence our estimators should provide a helpful and reliable tool to allow the algorithm to self tune.

In the next section we use these new estimators to design an improved algorithm which solves optimisation problems over two quantiles.

5.4.3 Algorithm

We can largely use the previously discussed algorithms (3.2 and 4.1), with similar adaptations to those introduced in section 5.3.1. In both of the main algorithms

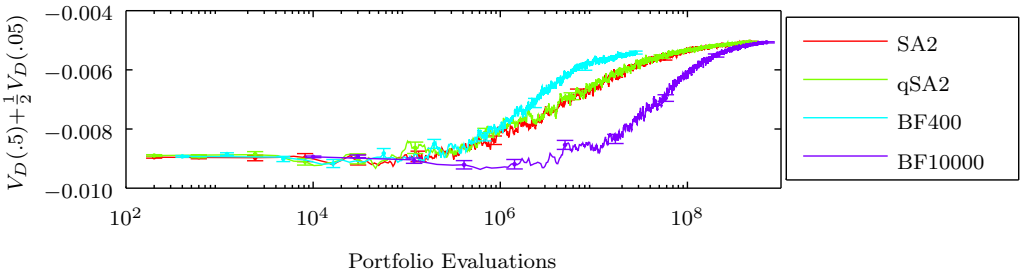
we simply need to calculate the estimators for both $\langle R_k \rangle$ and $\langle R_j \rangle$ rather than just $\langle R_k \rangle$. In algorithm 4.1 the estimators for both $\langle R_k \rangle$ and $\langle R_j \rangle$ can be calculated in each case by using the same technique as in the original algorithm.

The major change is in how we manage to keep both quantiles at the correct level of risk. For this, we use the same method of increasing n (the number of samples used for the estimations) to control $\langle R_j \rangle$. To control $\langle R_k \rangle$ we simply adjust the value of k (if needed) at the same time as we check whether we need to increase n. Specifically, we set $k \leftarrow k + \left(r_1 - \hat{R}_k\right) \frac{n}{2}$. This effectively approximates $\frac{d\langle R_k \rangle}{dk}$ to be $\frac{2}{n}$. We choose this value (a likely overestimate) as we want to limit the fluctuations of k in case the estimators do not converge correctly. The optimisation process should have enough opportunities to adjust k (and n) to ensure that we are operating at the correct levels of risk.

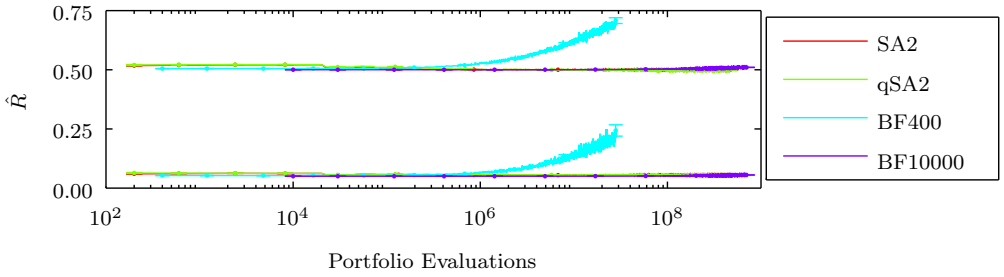
A table summarising the basic algorithms used for one quantile (before the above adaptations) can be found in appendix A.

5.4.4 Results

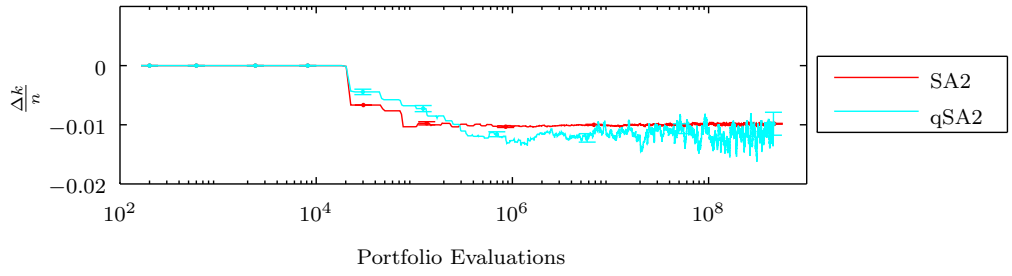
We use the same problem and parameter choices as in section 5.3.2 to test these new algorithms.



(a) The above figure shows the convergence of the algorithms when $\alpha = 0.5$. As in the case when the mean was used we see good results. We beat BF400 and appear to find the optimal solution.

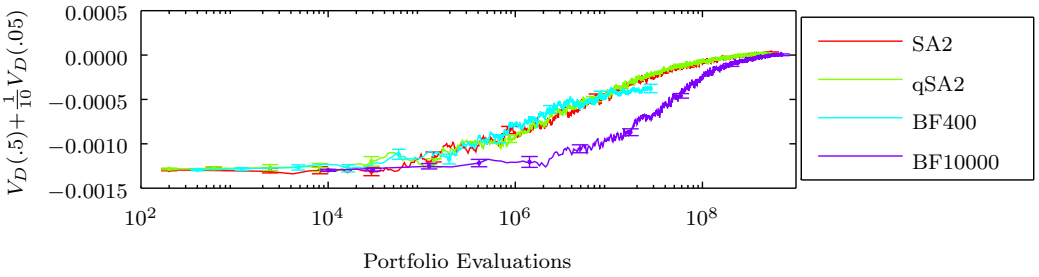


(b) The above graph shows the estimator for $\langle R \rangle$ over the course of the optimisation when $\alpha = 0.5$. We display the estimators developed in section 5.4.1, however both estimation methods give the same results. We see we manage both values of $\hat{R}$ well. BF400 struggles to hold both $\hat{R}$s low, which explains its (relatively) bad convergence.

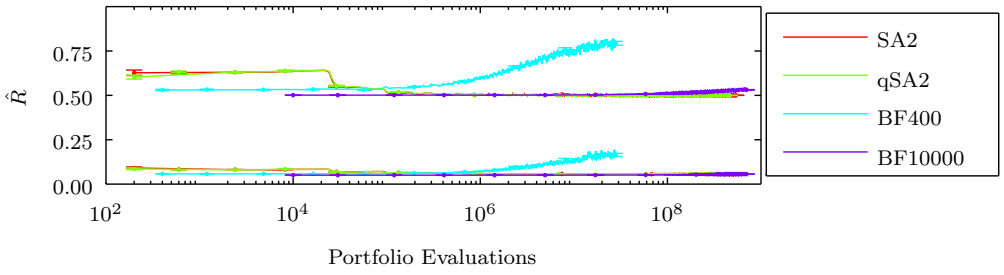


(c) The above plot shows how we adjusted k over the course of the algorithm. It can be seen that for the majority of the algorithm we used a choice of k that was equivalent to 49%.

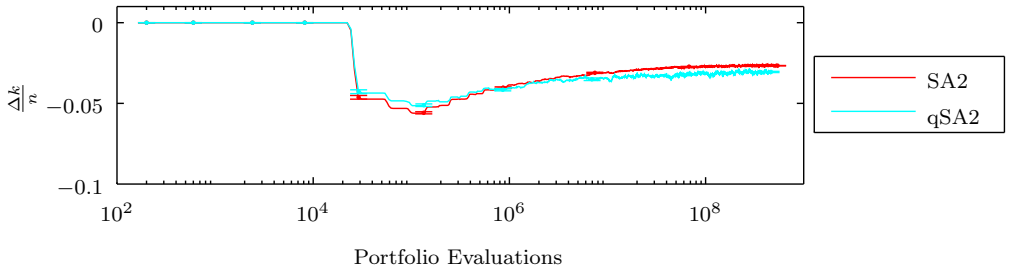
Figure 5.4: Results for the case when $\alpha = 0.5$.



(a) The above figure shows the convergence of the algorithms when $\alpha = 0.1$. In contrast to the case when the mean was used we see good results. We substantially beat BF400 and perform slightly better than BF10000.



(b) The above graph shows the estimator for $\langle R \rangle$ over the course of the optimisation when $\alpha = 0.1$. Both values of $\hat{R}$ are controlled well. Both BF400 (and to some extent BF10000) struggle to hold $\hat{R}$ low, which explains their (relatively) bad convergence.



(c) The above plot shows how we adjusted k over the course of the algorithm. It can be seen that for the majority of the algorithm we used a choice of k that was equivalent to 47%. This is lower than when $\alpha = 0.5$. This change is due to the increased dependence on $V_D(0.5)$ when $\alpha = 0.1$. This is likely to explain the particularly poor performance of BF400 in this case as $V_D(0.5)$ is more susceptible to biasing, for the relatively larger choice of alpha.

Figure 5.5: Results for the case when $\alpha = 0.1$.

Figures 5.4 and 5.5 show the results. When $\alpha = 0.5$ we see results similar to those in section 5.3 and the previous chapters. We do see that we need to change k

slightly to completely control $\langle R_k \rangle$, but the resulting bias is not huge.

When $\alpha = 0.1$ we see substantial improvement over the previous section, where we used the mean instead of the median. We can see from figure 5.5c that we have to work significantly harder to counteract the bias in $\hat{R}_k$ to control the optimisation (than when $\alpha = 0.5$). This effort pays off in the convergence results (figure 5.5a) where we outperform both brute force algorithms and in the case of BF10000 take less time to do it.

Thus we have (albeit briefly) shown that we can successfully control an optimisation that requires two quantiles. We have shown that this control is often critical as the naive choice of brute force algorithm (which uses $\frac{1}{r_2^2}$ samples) rarely finds the best solution.

5.5 Discussion

This chapter contains some interesting insights in how to extend the work from the two preceding chapters (3 and 4) to cover a wider variety of use cases. We have shown that the various estimators we have developed can be (reasonably) simply extended to further use cases. We have also shown that whilst we could assume that the statistics over all the samples are unbiased, this does not appear to be true (section 5.3), especially towards the end of the optimisations.

This chapter helps to open the door to many aspects of possible future work. As discussed in section 3.8, the work developed in this chapter should enable us to solve SLA style questions. We also note that whilst we would struggle to extend the estimators we original developed in chapter 3 to more than two quantiles, the estimators developed in chapter 4 should allow us to consider every single risk point should we so wish. This would mean we were able to optimise any given choice of $w(R)$ (section 5.2).

It is also possible that by using the estimators from chapter 4 we could find new estimators for both $\frac{d\langle R_k \rangle}{dk}$ and $\frac{d\langle R_k \rangle}{dn}$ to provide even greater algorithmic control.

Finally, in this chapter we operated by adjusting both n and k to control the quantiles we wished to optimise. One could imagine, for a choice of $w(R)$ which uses a large number of quantiles (for example the mean), using a form of importance sampling for the estimation. The importance sampling would be based on the estimators from chapter 4, as these provide access to the entire risk curve.

The notable achievement of this chapter was to take the work previously developed and expand it to two quantiles. This required some mathematical work to adapt the estimators previously used to the Markov chains induced by the new

optimisation problem. The numerical results were highly encouraging and showed a very strong performance for the new algorithm (section 5.4). This supported the assumption, made when constructing equations 5.3 and 5.4, that the expectations for $\langle R_j \rangle$ and $\langle R_k \rangle$ could be separated.

The hope of this chapter was to open the door to many avenues of future research. Some of these are described above and some of the overarching themes and extensions will be discussed in the final discussion in chapter 8.

Chapter 6

Estimating Extreme Risk

6.1 Introduction

In this (and the following) chapter we delve into the estimation and understanding of extreme risk.

There are a vast variety of problems in which the estimation of extreme risk plays a critical role, from reducing the loss of network packets to finance. We provide a new method for tackling this style of problem, with examples based in finance and reverse stress testing (1.3).

We refer to, and then extend, ideas pre-existing in the physical, statistical, computational and financial literatures. We aim to contrast, improve and transfer these ideas amongst the fields involved.

In this chapter we consider the ideas of [Chan and Kroese, 2011] and expand them to the Normal Inverse Gaussian distribution. This is a particularly relevant strategy when there is an understanding of the key (underlying) stochastic variable(s) driving a problem. To do this, we must first calculate the conditional probabilities of the NIG distribution. Using these, we define an algorithm to calculate the probabilities of extreme events quickly.

The central concept behind these ideas is that by understanding the asymptotic behaviour of a distribution, we can design an efficient sampling scheme. [Chan and Kroese, 2011] apply this intuition to a bridge network and the normal copula model used in finance for estimating credit risk.

This algorithm is applied to two distinct classes of problems, both of which are taken directly from a financial context. Firstly, we calculate the probabilities of extreme events in an example one dimensional NIG distribution. We then use the same ideas to calculate the probability of an extreme return in a single stock. This

is then expanded to calculating the likelihood of an extreme return for an entire portfolio of stocks.

The chapter is concluded in section 6.6.

In the following chapter (7) we consider the challenges involved when there may be no such understanding of the (underlying) stochastic variable(s) which drive a given problem. With the developed tool kit we are able to investigate both exponentially and algebraically tailed risk problems.

In the conclusion (chapter 8) we consider the impact of changing our focus. We consider moving away from the estimation of the risk at a particular point in state space to a more holistic approach. We discuss the implications of a method that attempts to understand all the causes and drivers of the risk in a particular problem.

6.2 Methodology

In this section we build on the ideas of [Chan and Kroese, 2011], as covered in section 2.5.3, in order to estimate the probabilities of extreme events for random variables which are Normal Inverse Gaussian (NIG) distributed. This is most directly applied to estimating the probability of a large loss of a financial stock.

By considering the functional form of the Normal Inverse Gaussian distribution (2.3.6), we note that when $\gamma < 0$, (sufficiently) large negative values will overwhelmingly be driven by extreme draws of $\mathbf{W}$, rather than $\mathbf{Z}$. This is because for $\mathbf{X}$ to have an extreme value, we require either large $\mathbf{W}$, or, alternatively, extreme values of both $\mathbf{W}$ and $\mathbf{Z}$. However, these scenarios do not have the same chance of occurrence. The heavy tailed nature of the Inverse Gaussian distribution ($\mathbf{W}$) implies it has a much higher likelihood of taking a high value than the Normal distribution ($\mathbf{Z}$), during an extreme event. Furthermore, the heavy tail makes it significantly more likely that only $\mathbf{W}$ is large, as opposed to both $\mathbf{W}$ and $\mathbf{Z}$ being large, due to the rate of 'decay' in the tail versus the growth in magnitude required.

We use this intuition to calculate the probabilities for rare events of random variables which are NIG distributed (predominantly when $\gamma < 0$).

To allow us to do this we need to calculate the conditional probabilities associated with the NIG distribution. These are $P(\mathbf{X} < x|z)$, where (fixed) $z \sim \mathbf{Z}$ and $P(\mathbf{X} < x|w)$, where (fixed) $w \sim \mathbf{W}$.

6.2.1 Conditional Probabilities

Proposition 6.2.1. *If* $\mathbf{X} \sim GH_1(\lambda, \chi, \psi, \mu, \sigma^2, \gamma)$ *(a one dimensional GH distribution), then*

$$P(\mathbf{X} < x|w) = \Phi\left(\frac{x - \mu - w\gamma}{\sqrt{w}\sigma}\right).$$

Proof. This result is a trivial application of 2.3.7. □

Proposition 6.2.2. *If* $\mathbf{X} \sim GH_1(\lambda, \chi, \psi, \mu, \sigma^2, \gamma)$ *(a one dimensional GH distribution), then when* $\gamma < 0$*,*

$$P(\mathbf{X} < x|z) = \begin{cases} 1 - F_W\left(w^-\right) & \text{if } \mu \geq x \\ F_W\left(w^+\right) + 1 - F_W\left(w^-\right) & \text{if } \mu < x, (\sigma z)^2 > 4\gamma(\mu - x), z > 0 \\ \mathbf{1}_{\mu<x} & \text{otherwise} \end{cases}$$

when $\gamma = 0$*,*

$$P(\mathbf{X} < x|z) = \begin{cases} F_W\left(\left(\frac{x-\mu}{\sigma z}\right)^2\right) & \text{if } \frac{x-\mu}{\sigma z} \geq 0, z > 0 \\ 1 - F_W\left(\left(\frac{x-\mu}{\sigma z}\right)^2\right) & \text{if } \frac{x-\mu}{\sigma z} \geq 0, z < 0 \\ \mathbf{1}_{\mu<x} & \text{otherwise} \end{cases}$$

and when $\gamma > 0$*,*

$$P(\mathbf{X} < x|z) = \begin{cases} F_W\left(w^+\right) & \text{if } \mu \leq x \\ F_W\left(w^+\right) - F_W\left(w^-\right) & \text{if } \mu > x, (\sigma z)^2 > 4\gamma(\mu - x), z < 0 \\ \mathbf{1}_{\mu<x} & \text{otherwise} \end{cases}$$

where F_W *is the cumulative distribution function of* W *and*

$$w^+ = \left(\frac{-\sigma z + \sqrt{(\sigma z)^2 - 4\gamma(\mu - x)}}{2\gamma}\right)^2 \text{ and}$$
$$w^- = \left(\frac{-\sigma z - \sqrt{(\sigma z)^2 - 4\gamma(\mu - x)}}{2\gamma}\right)^2.$$

Proof. We first consider the case when $\gamma < 0$.

Consider $x = \mu + \gamma w + \sigma\sqrt{w}z$. By rearranging and consequently using the

quadratic formulae we can solve for $\sqrt{w}$,

$$\begin{aligned} & x = \mu + \gamma w + \sigma\sqrt{w}z \\ \implies & (\mu - x) + \gamma w + \sigma\sqrt{w}z = 0 \\ \implies & \sqrt{w} = \frac{-\sigma z \pm \sqrt{(\sigma z)^2 - 4\gamma(\mu - x)}}{2\gamma}. \end{aligned} \tag{6.1}$$

For convenience, we define w^+ and w^- to be the two roots of 6.1, squared, assuming they exist, as defined above (in the proposition).

Since we only want real solutions of w, we need only consider positive roots of equation 6.1. For any positive roots to exist we require that

$$(\sigma z)^2 > 4\gamma(\mu - x) \tag{6.2}$$

and for an individual root to be positive we need (since $\gamma < 0$)

$$\pm\sqrt{(\sigma z)^2 - 4\gamma(\mu - x)} < \sigma z. \tag{6.3}$$

We first consider the situations under which two positive roots of equation 6.1 will exist. For condition 6.3 to be satisfied for both roots, we require that $z \geq 0$, as one root is positive. However for there to be two unique roots we require $z > 0$, as when $z = 0$ the solution will degenerate to one unique root. Furthermore, we require $4\gamma(\mu - x) > 0$, so that the positive root is less than σz (as $\sqrt{y^2 - \epsilon} < y \iff \epsilon > 0$, $\forall y \in \mathbb{R}$). Given that $\gamma < 0$, this simplifies to $-(\mu - x) > 0$, which rearranges to $x > \mu$. We also need to ensure condition 6.2 is true. Note that under these conditions $w^+ < w^-$.

If there are two roots, $\mu + \gamma w + \sigma\sqrt{w}z$ will be less than x when $w < w^+$ and when $w > w^-$. This statement follows from the asymptotic behaviour of X with respect to w. As $w \to \infty$, $\mu + \gamma w + \sigma\sqrt{w}z \to -\infty < x$ for any (fixed) choice of x or z (when $\gamma < 0$).

This implies $P(\mathbf{X} < x|z) = F_W(w^+) + 1 - F_W(w^-)$ when $\mu < x, (\sigma z)^2 > 4\gamma(\mu - x), z > 0$.

We now consider the conditions under which only one positive root of equation 6.1 will exist. If only one root exists it must be w^-. We can see this by considering the opposing case and finding a contradiction; if only the root of equation 6.1 linked to w^+ was positive we would have $+\sqrt{(\sigma z)^2 - 4\gamma(\mu - x)} \leq \sigma z < -\sqrt{(\sigma z)^2 - 4\gamma(\mu - x)}$, by condition 6.3, which is a contradiction for any choice of z.

For just w^- to be positive we require $4\gamma(\mu - x) \leq 0$ (from condition 6.3). If this condition is true then condition 6.2 must also be satisfied. Given $\gamma < 0$ this requires $\mu - x \geq 0$, or alternatively $\mu \geq x$.

By the same argument as for the two root case, $\mu + \gamma w + \sigma\sqrt{w}z$ will be less than x when $w > w^-$. This implies $P(\mathbf{X} < x|z) = 1 - F_W(w^-)$ when $\mu \geq x$.

Finally we consider the case when there are zero roots (which must be every case not covered above). This means that the outcome of condition ($\mu + \gamma w + \sigma\sqrt{w}z < x$) must be constant for all choices of w. When $w = 0$, the condition reduces to $\mu < x$, implying $P(\mathbf{X} < X|z) = \mathbf{1}_{\mu<x}$.

For the case when $\gamma > 0$ a symmetric argument applies (which we do not provide here).

When $\gamma = 0$, we have $x = \mu + \sigma\sqrt{w}z$, which rearranges to $\sqrt{w} = \frac{x-\mu}{\sigma z}$. Since we require $\sqrt{w} > 0$, we only have a solution when $\frac{x-\mu}{\sigma z} > 0$. Hence, the value of $P(\mathbf{X} < x|z)$ depends on the sign of z (by the same arguments as for $\gamma < 0$). We have

$$P(\mathbf{X} < x|z) = \begin{cases} F_W\left(\left(\frac{x-\mu}{\sigma z}\right)^2\right) & \text{if } \frac{x-\mu}{\sigma z} \geq 0, z > 0 \\ 1 - F_W\left(\left(\frac{x-\mu}{\sigma z}\right)^2\right) & \text{if } \frac{x-\mu}{\sigma z} \geq 0, z < 0 \\ \mathbf{1}_{\mu<x} & \text{otherwise.} \end{cases}$$

□

6.2.2 Estimating Extreme Risk in the NIG Distribution

Using the conditional probabilities defined above, we can construct a scheme for estimating $P(\mathbf{X} \leq x)$, for choices of x which give a very low probability (conditional on $\mathbf{Z}$). The scheme works as follows:

1. Generate N samples from $\mathbf{Z}$, denote them $\{z_i\}_{i=1}^N$.
2. Estimate $P(\mathbf{X} < x)$ using $\frac{1}{N}\Sigma_{i=1}^N P(\mathbf{X} \leq x|z_i)$.

It should be clear that this estimator would converge to the correct value, given sufficient N. However it is not an unbiased estimator as it will underweight the probabilities that either both $\mathbf{W}$ and $\mathbf{Z}$ are extreme, or that $\mathbf{Z} << 0$. However as discussed in the introduction, when $\gamma < 0$ we expect very little contribution from these cases. We will explore this hypothesis in the early results.

In the first example we also consider the opposite idea, where instead of using $P(\mathbf{X} \leq x|z_i)$ we use $P(\mathbf{X} \leq x|w_i)$ and random samples from $\mathbf{W}$ (conditional on $\mathbf{W}$). We examine this in order to confirm our original intuition.

6.2.3 Extending to d Dimensions

We can also apply the above method to a linear combination of the individual dimensions of a d dimensional NIG distribution. This would allow us to apply the method to a portfolio of stocks, rather than just a single stock. We can do this by applying proposition 2.3.12 to compress this d dimensional problem down to one dimension and hence apply the method above.

Mathematically, we consider the estimation of $P(\mathbf{Y} \leq y)$ where $\mathbf{Y} := \sum_{i=1}^{d} w_i \mathbf{X_i}$, $\mathbf{X} \sim GH_d(-\frac{1}{2}, \chi, \psi, \mu, \Sigma, \gamma)$ and $w_i \in \mathbb{R}_{>0}$ for $1 \leq i \leq d$. We note that by taking $B = \{w_1, \ldots, w_d\}$ we can apply proposition 2.3.12 to find that $\mathbf{Y} \sim GH_1(-\frac{1}{2}, \chi, \psi, B\mu, B\Sigma B', B\gamma)$. We can now apply the method in the previous section (6.2.2) to calculate the probabilities of a large loss.

This method is numerically tested in the third example by applying it to a simple basket of stocks.

In the following sections we apply this method (and hypothesis) to a simple test case and then to two financial examples, as outlined in the introduction (6.1).

6.3 Simple Test Case

In this section we apply the algorithm, as outlined above (6.2.2), to a simple one dimensional case. We choose a standard NIG distribution and vary the choice of γ so we can observe the performance of the algorithm in different contexts.

For this simple case we pick an arbitrary set of parameters which cover the different behaviours and probabilities of the NIG class of distributions. We choose $\chi = \frac{1}{2}, \psi = 2, \mu = 0, \sigma = 1$ and $\gamma \in \{-5, 0, 5\}$. The probability densities for the different γ are sketched in figure 6.1. It is possible to see the significant impact γ has on the skewness of the random variable $\mathbf{W}$. It is this skewness that we aim to exploit in the algorithm.

Given these example distributions, we calculate a range of different likelihoods for a set of given values. These values are (empirically) chosen to cover a wide range of probabilities.

Each of the three estimation techniques (standard Monte Carlo and the methods conditional on $\mathbf{Z}$ and $\mathbf{W}$ respectively) runs for $1,000$ samples and the results are then compared to the true likelihoods, as calculated using a significantly longer Monte Carlo simulation. The simulations were performed using MATLAB.

We present the results in table 6.1. We show both the true error and an estimation of the statistical error for each method. The statistical error was calculated

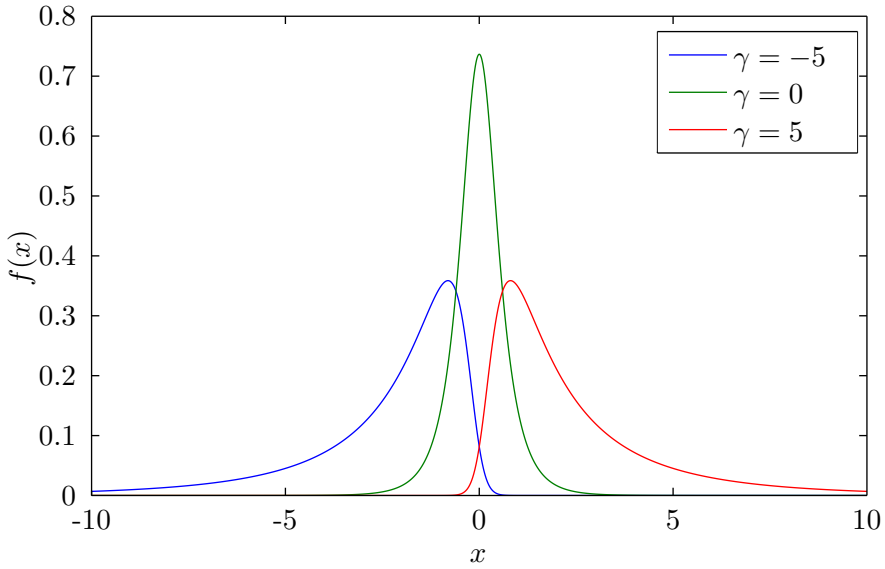


Figure 6.1: The Normal Inverse Gaussian probability density function, $f(x)$, for a range of γ. The remaining parameters are constant, with $\chi = \frac{1}{2}, \psi = 2, \mu = 0, \sigma = 1$. It is the heavy skewness, which γ induces, that we take advantage of in our algorithm.

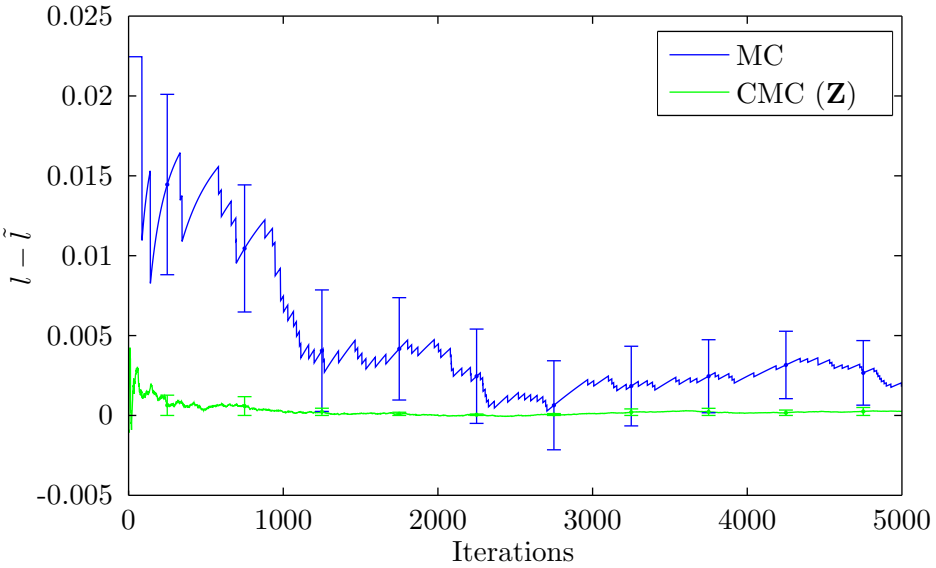


Figure 6.2: The new method demonstrates significantly faster convergence for $\gamma \leq 0$. The graph shows how the error in the likelihood estimate $(l - \tilde{l})$ changes for each method as the number of samples increases (for one run). The standard error on the estimator is also plotted. The NIG parameters are constant with $\chi = \frac{1}{2}, \psi = 2, \mu = 0, \sigma = 1$ and $\gamma = -5$.

v	l	Monte Carlo		Conditional Monte Carlo (on $\mathbf{Z}$)		Conditional Monte Carlo (on $\mathbf{W}$)	
		$l - \tilde{l}$	SE	$l - \tilde{l}$	SE	$l - \tilde{l}$	SE
-1.0	7.10×10^{-1}	7.64×10^{-3}	5.10×10^{-3}	-3.42×10^{-4}	1.52×10^{-3}	-6.02×10^{-3}	8.19×10^{-3}
-10.0	2.25×10^{-2}	1.15×10^{-3}	1.56×10^{-3}	-5.06×10^{-5}	8.14×10^{-5}	-5.97×10^{-3}	4.62×10^{-3}
-20.0	1.41×10^{-3}	1.09×10^{-4}	3.99×10^{-4}	-8.27×10^{-7}	6.60×10^{-6}	-9.79×10^{-4}	1.32×10^{-3}
-30.0	1.19×10^{-4}	-1.81×10^{-4}	1.70×10^{-4}	-1.12×10^{-6}	6.50×10^{-7}	1.19×10^{-4}	1.07×10^{-13}
-40.0	1.14×10^{-5}	?	?	2.86×10^{-8}	6.66×10^{-8}	1.14×10^{-5}	3.54×10^{-12}
-50.0	1.19×10^{-6}	?	?	7.75×10^{-9}	7.63×10^{-9}	1.19×10^{-6}	8.36×10^{-27}

(a) $\gamma = -5$

v	l	Monte Carlo		Conditional Monte Carlo (on $\mathbf{Z}$)		Conditional Monte Carlo (on $\mathbf{W}$)	
		$l - \tilde{l}$	SE	$l - \tilde{l}$	SE	$l - \tilde{l}$	SE
0.0	5.00×10^{-1}	9.70×10^{-3}	4.30×10^{-3}	8.90×10^{-3}	4.89×10^{-3}	8.92×10^{-7}	0
-1.0	6.55×10^{-2}	2.25×10^{-3}	2.53×10^{-3}	-9.63×10^{-4}	1.40×10^{-3}	1.19×10^{-3}	2.04×10^{-3}
-2.0	8.39×10^{-3}	-1.91×10^{-3}	1.08×10^{-3}	1.58×10^{-4}	3.68×10^{-4}	7.61×10^{-4}	6.31×10^{-4}
-4.0	2.19×10^{-4}	-1.81×10^{-4}	2.41×10^{-4}	-2.11×10^{-5}	2.50×10^{-5}	-7.57×10^{-5}	8.09×10^{-5}
-6.0	7.61×10^{-6}	?	?	1.11×10^{-6}	1.38×10^{-6}	-1.10×10^{-5}	1.55×10^{-5}
-8.0	3.02×10^{-7}	?	?	8.60×10^{-8}	7.98×10^{-8}	-7.05×10^{-7}	7.41×10^{-7}

(b) $\gamma = 0$

[Table Continued Overleaf]

v	l	Monte Carlo		Conditional Monte Carlo (on $\mathbf{Z}$)		Conditional Monte Carlo (on $\mathbf{W}$)	
		$l - \tilde{l}$	SE	$l - \tilde{l}$	SE	$l - \tilde{l}$	SE
0.5	1.15×10^{-1}	-5.79×10^{-3}	2.84×10^{-3}	-8.04×10^{-4}	1.27×10^{-3}	-2.14×10^{-3}	5.53×10^{-3}
0.2	3.87×10^{-2}	1.51×10^{-3}	1.70×10^{-3}	2.19×10^{-3}	7.53×10^{-4}	-6.45×10^{-4}	2.23×10^{-3}
-0.1	6.59×10^{-3}	5.88×10^{-4}	7.14×10^{-4}	2.08×10^{-4}	4.06×10^{-4}	-5.80×10^{-5}	3.65×10^{-4}
-0.4	5.70×10^{-4}	-2.30×10^{-4}	2.98×10^{-4}	6.44×10^{-5}	1.33×10^{-4}	3.36×10^{-6}	2.18×10^{-5}
-0.7	3.24×10^{-5}	?	?	?	?	5.53×10^{-7}	9.86×10^{-7}
-1.0	1.52×10^{-6}	?	?	?	?	-1.12×10^{-7}	4.57×10^{-8}

(c) $\gamma = 5$

Table 6.1: Performance of the proposed algorithm (Conditional Monte Carlo (on $\mathbf{W}$)) versus regular Monte Carlo sampling. The algorithm performs very strongly for all likelihoods when $\gamma = -5$ and smaller likelihoods when $\gamma \geq 0$. Results are also provided for Conditional Monte Carlo (on $\mathbf{Z}$) which performs very poorly (except when $\gamma > 0$). These results are as predicted in section 6.2.2. Results marked with a ? indicate the algorithm failed to calculate a prediction. l is the true likelihood and $\hat{l}$ is the likelihood estimate found using each method. SE is the standard error as predicted by the method.

using the standard error ($\frac{\sigma}{\sqrt{50}}$), where the variance (σ^2) was calculated by bucketing the individual estimates into fifty buckets.

Table 6.1a ($\gamma = -5$) shows that the new method (Conditional Monte Carlo (on $\mathbf{Z}$)) significantly outperforms regular Monte Carlo for all likelihoods considered (71% to 10^{-6}%). The method generally reaches a result that is up to a thousand times more accurate. (Note that for likelihoods smaller than 10^{-4} regular Monte Carlo fails to get a reasonable estimate.) The improved accuracy implies a much faster convergence. Figure 6.2 supports this significantly faster convergence (when $\gamma = -5$). In this example Conditional Monte Carlo (on $\mathbf{Z}$) has converged within 500 samples, whereas regular Monte Carlo is still converging after 5000 samples.

Table 6.1b ($\gamma = 0$) shows similar results to the previous case when $v \leq -1$ (or, alternatively, when the probability is less then 6.5%). This suggests that the method reduces in effectiveness as γ increases but increases in effectiveness as the probability reduces. It is surprising that the method performs so well in this situation. At $\gamma = 0$ the asymptotic behaviour of the distribution is different compared to when $\gamma < 0$ and so our original intuition into the problem does not hold.

Table 6.1c ($\gamma = 5$) shows that the new method has a similar performance to regular Monte Carlo for all likelihoods (both methods fail to produce any reasonable results for $v < -0.1$). This is as expected as in this case the lower risk tail should be dominated by the normal distribution.

In contrast, the 'mirrored' method (Conditional Monte Carlo (on $\mathbf{W}$)) performs poorly for $\gamma \leq 0$, as expected from the intuition provided in the previous section (6.2). Its strong performance when $\gamma = 5$ is more surprising. We do not further consider CMC on $\mathbf{W}$ in later sections as it becomes incalculable in higher dimensional problems.

Finally, we note that for all choices of γ the true error is consistently bounded by the statistical errors for the Conditional Monte Carlo (on $\mathbf{Z}$) method. This implies that there is very low bias in the estimator for any choice of γ. The same cannot always be said for the Conditional Monte Carlo (on $\mathbf{W}$) method. In this case there is clearly a systematic error at work, making it very hard to predict the true error when this method is used.

These empirical results largely support the intuition provided in section 6.2. They show that the method is particularly effective for $\gamma \leq 0$ but less effective for positive γ. Additionally, the results have shown that the quality of the estimation at worst matches that of regular Monte Carlo and that the standard error is a good estimator for the true error.

In this section we have shown that the method is effective in a simple example.

In later sections we will apply this method to two financial examples.

6.4 Large Single Stock Loss

As highlighted in the introduction, we now apply the Conditional Monte Carlo method to a selection of different stocks.

Stock returns can be calculated both logarithmically (X_t) and by a simple ratio (Y_t). For a stock with historical prices $S_t, t \geq 0$, we can consider either

$$X_t = \log S_t - \log S_{t-1} \text{ or}$$
$$Y_t = \frac{S_t - S_{t-1}}{S_{t-1}}.$$

A significant number of authors prefer X_t as it allows for continuous compounding (i.e. for a period of n then the sum is simply $X_t = \log S_{t+n-1} - \log S_{t-1}$). We also note that both methods of calculating the rate of return are actually very similar in value ($Y_t - X_t = \frac{1}{2}X_t^2 + \frac{1}{6}X_t^3 + \dots$ where X_t is typically of order 10^{-2} orders) [Enerlein and Keller, 1995].

We use Y_t in the example below. Although it would be possible to use either return series in the single stock case, it is much simpler to apply Y_t in the portfolio example in the next section.

For data fitting we use the Multi Cycle, Expectation, Conditional Expectation (MCECM) algorithm as implemented in the R package ghyp [Breymann and Luthi, 2013]. More information on this algorithm can be found in [McNiel et al., 2005].

We consider the stocks of three major global banks; Goldman Sachs (GS), Morgan Stanley (MS) and Credit Suisse (CS). We use stock return data from 2010 to fit our NIG distribution and then run the method to calculate the various risks. We make reference to the Monte Carlo case in our comparisons and leave questions of the validity of the NIG distribution to stocks returns at different probability levels to other authors, for example [Bingham et al., 2001].

The parameters resulting from the data fitting to the NIG distribution can be found in table 6.2. We then calculate the probabilities of various returns on these distributions in tables 6.3, 6.4 and 6.5. As in the previous example, we run each method for $1,000$ samples and the results are compared to the true likelihoods, as calculated using a significantly longer Monte Carlo simulation. The simulations were performed using MATLAB. The results are displayed in the same form as previously.

In the results v represents a percentage return. For example, the chance of

Stock	χ	ψ	μ	σ	γ
Goldman Sachs (GS)	1.16	1.16	1.79×10^{-3}	1.85×10^{-2}	-1.61×10^{-3}
Morgan Stanley (MS)	8.71	8.71	-2.69×10^{-3}	2.10×10^{-2}	2.58×10^{-3}
Credit Suisse (CS)	1.15	1.15	2.36×10^{-2}	2.36×10^{-2}	-3.25×10^{-3}

Table 6.2: The parameters (of the NIG distribution) resulting from the data fitting to the various stocks using the MCECM algorithm [McNiel et al., 2005] and 2010 return data. We note we have a variety of γ values.

v	l	Monte Carlo		CMC (on $\mathbf{Z}$)	
		$l - \tilde{l}$	SE	$l - \tilde{l}$	SE
0.01	7.36×10^{-1}	3.70×10^{-3}	1.52×10^{-2}	1.88×10^{-2}	1.49×10^{-2}
-0.02	1.13×10^{-1}	-6.12×10^{-3}	1.09×10^{-2}	2.71×10^{-3}	6.95×10^{-3}
-0.05	1.12×10^{-2}	2.22×10^{-3}	2.74×10^{-3}	1.70×10^{-3}	1.17×10^{-3}
-0.08	1.35×10^{-3}	-6.46×10^{-4}	1.40×10^{-3}	-1.90×10^{-4}	2.90×10^{-4}
-0.11	1.85×10^{-4}	?	?	3.03×10^{-5}	3.80×10^{-5}
-0.14	2.72×10^{-5}	?	?	3.72×10^{-6}	1.18×10^{-5}

Table 6.3: The results of the likelihood estimation for Goldman Sachs. Conditional Monte Carlo (CMC) shows an order of magnitude improvement verses Monte Carlo for probabilities less than 1%. Results marked with a ? indicate the algorithm failed to calculate a prediction.

the GS stock losing at least 2% of its value is approximately 11%.

The results for the financial stocks echo the results in the simple example (6.3). For GS and CS (where $\gamma \leq 0$) the Conditional Monte Carlo provides improved results over regular Monte Carlo, generally with an order of magnitude improvement in both the estimation and standard error for likelihoods (l) less than 1%. For greater cumulative probabilities Conditional Monte Carlo matches the performance of traditional Monte Carlo.

For MS (where $\gamma > 0$) we see less encouraging results. The method has similar performance to regular Monte Carlo, however for more extreme events the standard error reports a much higher accuracy than has actually been obtained (although this is also true of regular Monte Carlo). This implies that there could be a stronger systematic basis at work for this case. Despite this, the method still shows there is at least some probability of the event occurring whereas regular Monte Carlo appears to show with absolute certainty the event would never occur (and has a standard error of 0).

In this example we have shown that the method is effective in a real situation,

v	l	Monte Carlo		Conditional Monte Carlo (on $\mathbf{Z}$)	
		$l - \tilde{l}$	SE	$l - \tilde{l}$	SE
0.01	6.93×10^{-1}	-1.18×10^{-2}	1.66×10^{-2}	-2.23×10^{-2}	1.60×10^{-2}
-0.02	1.65×10^{-1}	-1.09×10^{-2}	1.32×10^{-2}	-7.95×10^{-3}	1.29×10^{-2}
-0.05	9.85×10^{-3}	8.47×10^{-4}	3.41×10^{-3}	3.35×10^{-3}	1.71×10^{-3}
-0.08	2.49×10^{-4}	?	?	2.00×10^{-4}	2.78×10^{-5}
-0.11	4.21×10^{-6}	?	?	4.14×10^{-6}	3.94×10^{-8}
-0.14	5.94×10^{-8}	?	?	5.94×10^{-8}	2.34×10^{-17}

Table 6.4: The results of the likelihood estimation for Morgan Stanley. Conditional Monte Carlo (CMC) has similar performance to that of Monte Carlo. However it does indicate there is some probability weight for lower probability events, where Monte Carlo shows no weight. Results marked with a ? indicate the algorithm failed to calculate a prediction.

v	l	Monte Carlo		Conditional Monte Carlo (on $\mathbf{Z}$)	
		$l - \tilde{l}$	SE	$l - \tilde{l}$	SE
-0.02	1.66×10^{-1}	1.79×10^{-2}	9.58×10^{-3}	1.01×10^{-2}	8.72×10^{-3}
-0.05	2.79×10^{-2}	8.89×10^{-4}	4.78×10^{-3}	-1.17×10^{-3}	3.05×10^{-3}
-0.08	5.32×10^{-3}	-6.79×10^{-4}	2.32×10^{-3}	-6.14×10^{-5}	7.97×10^{-4}
-0.11	1.12×10^{-3}	-1.88×10^{-3}	1.70×10^{-3}	-4.14×10^{-5}	2.10×10^{-4}
-0.14	2.54×10^{-4}	-7.46×10^{-4}	1.00×10^{-3}	-3.23×10^{-5}	8.80×10^{-5}
-0.17	6.01×10^{-5}	?	?	-4.28×10^{-6}	2.02×10^{-5}

Table 6.5: The results of the likelihood estimation for Credit Suisse. Conditional Monte Carlo (CMC) shows an order of magnitude improvement verses Monte Carlo for probabilities less than 1%. Results marked with a ? indicate the algorithm failed to calculate a prediction.

when $\gamma \leq 0$. We have, however, found that there may be some concerns for the method when $\gamma > 0$. This is to be expected given the original intuition that underlies the method. In the final example we consider a portfolio that contains all three stocks.

6.5 Large Portfolio Loss

In this final example we show the method can be applied to a portfolio (or basket) of stocks. We use the same data as in the previous example (6.4). We fit the returns of the three stocks to a multivariate NIG distribution using the MCECM algorithm and then apply the ideas from section 6.2.3.

v	l	Monte Carlo		CMC (on $\mathbf{Z}$)	
		$l - \tilde{l}$	SE	$l - \tilde{l}$	SE
0.00	4.28×10^{-1}	1.89×10^{-2}	1.45×10^{-2}	-6.18×10^{-3}	4.38×10^{-3}
-0.01	9.38×10^{-4}	-1.06×10^{-3}	1.40×10^{-3}	1.53×10^{-5}	1.41×10^{-5}
-0.02	7.48×10^{-6}	?	?	-2.29×10^{-7}	2.26×10^{-7}
-0.03	8.05×10^{-8}	?	?	1.60×10^{-9}	2.29×10^{-9}
-0.04	1.00×10^{-9}	?	?	-2.52×10^{-11}	3.35×10^{-11}

Table 6.6: The results for a portfolio consisting of 25% of GS stock, 25% of MS Stock and 50% of CS Stock. CMC outperforms MC by one or two orders of magnitude for all the probabilities tested. Results marked with a ? indicate the algorithm failed to calculate a prediction.

We consider a portfolio which contains 25% of GS stock, 25% of MS Stock and 50% of CS Stock (so that there is an even stock weighting between global regions). By fitting the data to a 3-dimensional NIG distribution and applying the relevant transformations from proposition 2.3.12 we get a one dimensional NIG distribution for the whole portfolio, with parameters $\chi = 1.81$, $\psi = 1.81$, $\mu = 1.21 \times 10^{-3}$, $\sigma = 3.92 \times 10^{-4}$ and $\gamma = -1.44 \times 10^{-3}$.

Running both Conditional Monte Carlo (CMC) and Monte Carlo (MC) for $1,000$ iterations gives the results shown in table 6.6. We see very impressive results for all choices of v. CMC outperforms MC by one or two orders of magnitude for all the probabilities tested. We note that since the portfolio overall had negative gamma we would expect strong results.

In this (final) example we have further demonstrated both the effectiveness of the method and shown its possible application to calculating portfolio returns.

6.6 Discussion

In this chapter we have demonstrated the effectiveness of our new method (under certain conditions) for calculating the likelihoods of rare events modelled using a Normal Inverse Gaussian (NIG) distribution. We have further applied this method to two separate financial situations. We have consistently shown that when $\gamma \leq 0$ an adapted method from that of [Chan and Kroese, 2011] shows, at worst, an order of magnitude of improvement over regular Monte Carlo for rare events.

We believe the extension of this method to the NIG distribution and the application to a financial portfolio to both be new and novel results.

It would be an interesting extension to consider logarithmic stock returns.

This is relatively simple for a single stock, but would be more challenging for the portfolio case as it would require using a numerical solver. This is because we would not be able to use proposition 2.3.12 and would need to solve an expression of the form $\sum_{i=1}^{n} e^{\mu_i+\gamma_i \mathbf{W}+z_i}$ for $\mathbf{W}$ in order to calculate the conditional probabilities.

In later chapters we will begin to challenge the assumptions made in this chapter and consider alternative ideas for calculating the likelihood of extreme events. In this chapter we relied heavily on both the availability of the cumulative distribution function for the Inverse Gaussian distribution ($\mathbf{W}$) and on our intuition informing us of the dominant variable driving the system. These assumptions may not always be valid and hence we dispose of the first assumption in the following chapter (7) and both assumptions when we discuss the ideas further in chapter 8.

Chapter 7

Estimating Extreme Risk: Thermal Integration

7.1 Introduction

In this set of chapters we are considering the estimation and understanding of extreme risk.

In the previous chapter (6) we developed an effective method for calculating the likelihood of rare events in Normal Inverse Gaussian distributed random variables. This method required knowledge of the cumulative distribution function and an intuition into the driving variables of the distribution.

In this chapter we consider the challenges involved when we do not have a relevant cumulative distribution function available. We take the ideas used in thermal integration (in physics) and adapt them to the problem at hand. With the developed tools we are able to explore a wider set of financial problems than in the previous chapter.

As discussed in section 2.5.5, we do not consider extreme value theory to be applicable to the style of problem we wish to solve in this chapter and hence do not discuss it further.

In the conclusion (chapter 8), we will consider a change of focus. We will consider taking a more holistic approach in an effort to understand the causes and drivers of the risk in a given problem. This should allow us to drop our second key assumption; we would no longer need to have an understanding of the variables driving a given problem.

In this chapter we estimate the probability of certain extreme events by building up a sequence of conditional probability estimates. These estimates are

then combined to give an overall estimate of the probability. This, at least in theory, should provide an improved estimate compared to a single, large scale, simulation.

The fundamental idea is that if we can efficiently estimate these conditional probability estimates we should be able to design an algorithm which requires $O(p^{-1})$ samples instead of $O(p^{-2})$ samples that a regular Monte Carlo chain would require to estimate the likelihood of an event with probability p of occurring.

However, as with most heuristic methods, there is a multitude of different options to consider. For simplicity, we construct and consequently evaluate just two such methods.

We will show that both of these methods are more effective than regular Monte Carlo, both in a simple test problem and in several real world financial problems.

7.2 Methodology

In this section we extend the ideas of thermal integration (see 2.5.4) to a method of estimating the likelihood of rare events of Generalised Hyperbolic distributed random variables.

In this chapter we have assumed we no longer have an analytic inverse cumulative distribution function for the mean term of the normal mean-variance mixture distribution ($\mathbf{W}$) and must therefore take a different approach to the previous chapter. In the last chapter we used the c.d.f. to design a scheme by which we did not need to randomly sample from the 'mean term' ($\mathbf{W}$) but could simply estimate the likelihoods using normal samples ($\mathbf{Z}$). By contrast, in this chapter we consider the use of importance sampling to drive our sampler deep into the risk tail and use the resulting samples to estimate the probability of rare events efficiently.

Consider a random variable, $\mathbf{Y}$, with state space Ω_Y and an associated value function which maps an element $y \in \Omega_Y$ to the real line, $V(y) : \Omega \to \mathbb{R}$. We wish to calculate the probability of a relatively rare value of $V(\mathbf{Y})$ occurring, $P(V(\mathbf{Y}) < v)$. Note that in this case Ω_Y may be uncountable. This probability can be trivially expanded to

$$P(V < v) = P(V < v | V < v_{n-1}) \left(\prod_{i=1}^{n-1} P(V < v_i | V < v_{i-1}) \right) P(V < v_0)$$

where $V = V(\mathbf{Y})$ and $\{v_i\}_{i=0}^{n-1} \in \mathbb{R}$ is a decreasing sequence, with $v_{n-1} > v$. We can consider $P(V < v | V < v_{n-1})$ to be the estimation term and
$\left(\prod_{i=1}^{n-1} P(V < v_i | V < v_{i-1}) \right) P(V < v_0)$ to be the renormalizing term to the true

probability distribution, as given by $\mathbf{Y}$. We hypothesise that this separation of the problem should allow for a more effective estimation of the likelihood of rare events.

Assuming we can estimate the conditional probabilities, as above, in a more effective way than just taking samples from the whole distribution, there should be a significant performance increase in likelihood estimation for extreme events using this method. We consider two such methods for the Generalised Hyperbolic class of distributions.

Our aim is to construct importance samplers which can estimate the conditional probabilities outlined above, e.g. $P(V < v_2|V < v_1)$, with $O(100)$ samples as opposed to the $O(P(V < v_2)^{-2})$ usually required by regular Monte Carlo approaches. For sufficiently rare events, this should lead to a very significant performance increase. These performance estimates are detailed in section 7.2.2.

Both our estimation methods rely on a similar intuition to the previous chapter for Normal Inverse Gaussian distributions. By considering the functional form of the Generalised Hyperbolic distributions (2.3.6), we note that when $\gamma < 0$ (sufficiently) large negative values must be driven by extreme draws of $\mathbf{W}$, rather than $\mathbf{Z}$. Further discussion of this intuition can be found in section 6.2.

7.2.1 The Algorithm

We use the same algorithm for both methods. For a target GH distributed random variable, $\mathbf{X}$, we estimate $P(V < v)$ as below:

1. Generate N samples from $\mathbf{X}$. Estimate the q_0 quantile of $V(\mathbf{X})$ from the samples and set v_0 equal to it (i.e. $P(V < v_0) = q_0$).

2. Set $i = 1$.

3. If $i > i_{max}$ go to step 8 or if $v_i < v$ go to step 7.

4. Draw N samples from an appropriate importance sampler, $\mathbf{X_i}$ (two samplers are given below). Calculate the q_i quantile of $V(\mathbf{X})$ from the samples and set v_i equal to it (i.e. $P_{\mathbf{X_i}}(V < v_i) = q_i$).

5. Estimate $P(V < v_i|V < v_{i-1})$ by reweighing the samples (as in regular importance sampling).

6. Set $i = i + 1$ and go to step 3.

7. If $i < i_{max}$ take an additional $N(i_{max} - i)/i$ samples for each X_i and update the conditional probability estimates.

8. Estimate $P(V < v)$ using

$$P(V < v|V < v_{i-1}) \left(\prod_{j=1}^{i-1} P(V < v_j|V < v_{j-1}) \right) P(V < v_0).$$

$N, i_{max} \in \mathbb{N}$ and $q_i \ \forall 0 \leq i \leq i_{max}$ are parameters of the method.

As previously discussed, assuming sufficiently good importance samplers we should significantly outperform regular Monte Carlo when estimating extreme risks, particularly when $\gamma > 0$.

We now consider the potential performance increase from such a method, before considering two different possible importance samples for the generalised hyperbolic distribution.

7.2.2 Expected Performance

We can predict the expected performance increase by estimating the number of samples, N, required by both regular Monte Carlo and the proposed method to estimate a given probability, p, with a fixed level of accuracy, a.

For regular Monte Carlo we can assume that the samples follow a binomial distribution with probability p. This should lead to a probability estimate of $\hat{p} = \frac{\hat{\mu}}{N} \pm \frac{\hat{\sigma}}{N}$. Substituting for $\hat{\mu}$ (estimated mean) and $\hat{\sigma}$ (estimated standard deviation) using standard binomial statistics and rearranging gives $\hat{p} = p(1 \pm \frac{1}{\sqrt{Np}})$. Thus, to achieve an accuracy of $a\%$ we require $N = O\left(\frac{1}{a^2 p}\right)$. We further note that in practice $N = O\left(\frac{1}{p^2}\right)$ is often used.

In contrast, for the proposed method we assume that each iteration need only consider probabilities of 10%. This means we require $-\log_{10}(p)$ iterations. Furthermore, any errors in the estimation will compound from the individual iterations. Consequently if we require an overall accuracy of $a\%$, each individual iteration requires an accuracy of $\sqrt[-\log_{10}(p)]{(a+1)} - 1$. Assuming each iteration uses the same method of calculating the number of samples required as regular Monte Carlo, then the method requires $N = O\left(-\log_{10}(p) \frac{10}{\left(\sqrt[-\log_{10}(p)]{(a+1)} - 1\right)^2} \right)$ samples, or $N = O\left(-\log_{10}(p) 100\right)$ if we use the common convention for regular Monte Carlo.

To put this in perspective, we should only require $O(10^2)$ to $O(10^4)$ samples to estimate a probability of 0.1%, within a 10% accuracy, whereas a standard Monte Carlo approach would require between $O(10^5)$ to $O(10^6)$ samples. Figure 7.1 shows the number of samples required for both methods for a variety of probabilities.

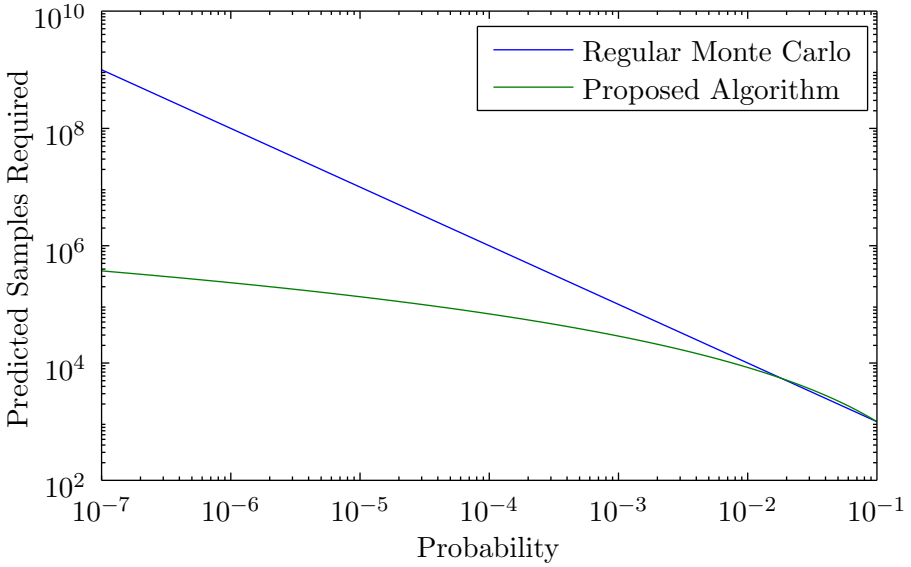


Figure 7.1: An estimate of the number of samples required for regular Monte Carlo and the newly proposed algorithm for a variety of different probabilities. Significant improvement is seen for $p \leq 10^{-3}$.

We now consider two different importance sampling techniques. We consider replacing the Generalised Inverse Gaussian (GIG) distribution with both a log normal distribution and an alternative GIG distribution.

7.2.3 Sampler: Log Normal Distribution

We define X_i to be a Mean-Variance model where $\mathbf{W}_i$ has a log normal distribution (instead of the GIG distribution in the GH case). The log normal distribution was chosen as it matches the requirements of $\mathbf{W}$ in the definition (positive and scalar valued) and is heavy tailed so it should exhibit similar behaviour.

It is worth noting that it is also much cheaper (computationally) to produce log normal samples as compared to generating GIG samples. (A log normal sampler requires approximately half the amount of uniform random inputs compared to a GIG sampler.)

The new importance sampler was designed to sample a range of values that would cover the previous cut off (v_{i-1}) in one tail and significantly rarer outcomes in the opposite tail.

The parameters for the log normal distribution used for sampling were calculated using the mean and variance of $\mathbf{W}$ in the tail of the previous sample, with respect to the original distribution. Mathematically, consider $p_0(w)$ to be the prob-

ability density function (p.d.f.) of $\mathbf{W}$ of the original distribution, X_0, and $p_i(w)$ to be the p.d.f.s of $\mathbf{W}$ in the various importance samplers and associated distributions, X_i, used by the algorithm. We then have that

$$m_i' = \left\langle w_{i-1}\mathbf{1}_{V(x_{i-1})<v_{i-1}} \right\rangle_{X_0} \text{ and}$$
$$v_i^2 = Var_{X_0}\left(w_{i-1}\mathbf{1}_{V(x_{i-1})<v_{i-1}}\right)$$

where $\langle - \rangle_{X_0}$ and $Var_{X_0}(-)$ are calculated with respect to X_0 . This is estimated by appropriately reweighting the samples drawn from the importance sampler, using the relation $\langle w \rangle_{X_0} = \left\langle \frac{p_0(w)}{p_i(w)} \right\rangle_{X_i}$.

The mean is then further shifted into the tail using the variance ($m_i = m_i' + 2v_i^2$). Both m and v_i can then be converted into the log normal parameters

$$\mu_i = \log\left(\frac{m_i^2}{\sqrt{v_i + m_i^2}}\right) \text{ and} \tag{7.1}$$

$$\sigma_i = \sqrt{\log\left(\frac{v_i}{m_i^2 + 1}\right)}. \tag{7.2}$$

The resulting Mean-Variance distribution ($\mathbf{X_i}$), which is based on $\mathbf{W}_i$, should cover a wide range of values. Our intuition that large $\mathbf{W}$ leads to rare events suggests that one tail will contain values around v_{i-1} and the other tail significantly rarer events (with a coverage of approximately 4σ).

This sampler was used, in combination with the algorithm above (7.2.1), in a variety of situations, including some real world examples. The results section of this chapter contains further information.

7.2.4 Sampler: Chi Adjustment

This sampler used a Generalised Hyperbolic distribution as our importance sampler with an adjusted choice of χ to shift the mean. All other parameters remained the same.

For each iteration we estimated the expected value of $\mathbf{W}$ which occurred in the tail of the previous sampler (when $V < v_{i-1}$). This is m_i' in the description above (equation 7.1). We then numerically solved for χ_i such that $m_i' = \langle \mathbf{W}_i(\lambda, \chi_i, \psi) \rangle$, using the expression in proposition 2.3.13. Given m_i' is always greater than zero we can be confident such a χ_i exists.

This sampler was tested against a range of scenarios when used with the algorithm outlined above. These examples can be seen in the results section.

7.2.5 Algorithm Parameters

In order to compare the methods across a range of situations, one set of parameters (N, i_{max} and q_i for $1 \leq i \leq i_{max}$) were chosen for all the different test cases. This also allowed us to ensure that the methods were not optimised for each specific test case.

We gave each sampling method a fixed budget of $1,250$ samples. They could use up to 5 iterations ($i_{max} = 4, N = 250$). q_i was set to 0.1 for all i. The methods were compared to a Monte Carlo estimate that also used $1,250$ samples.

The results from 50 runs were then averaged and the standard error calculated. We would expect regular Monte Carlo to perform well for probabilities as small as 10^{-3} but begin to suffer after this. As outlined above our two sampling methods should be able to estimate significantly smaller probabilities.

7.3 Example 1: 1D NIG Distribution

In this section we apply the algorithm, with the two samplers, to a simple one dimensional test problem. We choose the same standard NIG distribution as used in the previous chapter (6.3) so that the results are comparable. The probability densities for each of the cases considered can be seen in figure 6.1.

As in previous numerical examples, the values considered were empirically selected to cover a wide range of probabilities. All the simulations were performed in MATLAB.

The results are presented in table 7.1. The algorithm parameters used are as described in section 7.2.5.

Table 7.1a ($\gamma = -5$) shows a solid performance for both methods. The Chi adjustment method however performs best. It shows an order of magnitude improvement for probabilities smaller than 10^{-3} compared to regular Monte Carlo. In contrast, the log-normal sampler struggles to outperform regular Monte Carlo for low probabilities. For probabilities smaller than 10^{-5} regular Monte Carlo fails to get any estimate of the risk. This gives the opportunity for both samplers to come into their own. They achieve reasonable estimates for probabilities as small as 10^{-9}.

Table 7.1b ($\gamma = 0$) shows a reasonable performance for both methods. The Chi adjustment method however maintains its increased performance. The performance of both methods is not quite as good as regular Monte Carlo for probabilities greater than 10^{-5}, although the Chi adjustment method mostly manages to match Monte Carlo. However, as before, for probabilities smaller than 10^{-5} regular Monte Carlo fails to get any estimate of the risk whereas both samplers at least obtain an

v	l	Monte Carlo		Thermal Integration: Log N		Thermal Integration: Chi	
		$l - \tilde{l}$	SE	$l - \tilde{l}$	SE	$l - \tilde{l}$	SE
-10.00	1.16×10^{-1}	1.09×10^{-3}	1.15×10^{-3}	-2.98×10^{-4}	1.27×10^{-3}	2.23×10^{-3}	1.12×10^{-3}
-20.00	1.30×10^{-2}	-2.73×10^{-4}	4.32×10^{-4}	-8.34×10^{-3}	1.81×10^{-3}	2.99×10^{-4}	3.36×10^{-4}
-30.00	1.56×10^{-3}	7.80×10^{-6}	1.57×10^{-4}	-1.09×10^{-3}	2.65×10^{-4}	1.17×10^{-5}	7.19×10^{-5}
-40.00	1.95×10^{-4}	3.48×10^{-5}	4.57×10^{-5}	-2.01×10^{-4}	1.52×10^{-4}	6.93×10^{-6}	8.64×10^{-6}
-50.00	2.49×10^{-5}	8.89×10^{-6}	1.60×10^{-5}	-6.86×10^{-5}	3.26×10^{-5}	1.09×10^{-7}	2.02×10^{-6}
-60.00	3.23×10^{-6}	?	?	-1.06×10^{-6}	8.80×10^{-7}	1.05×10^{-7}	2.03×10^{-7}
-70.00	4.23×10^{-7}	?	?	-1.67×10^{-6}	1.26×10^{-6}	-1.89×10^{-8}	3.22×10^{-8}
-80.00	5.66×10^{-8}	?	?	-5.82×10^{-8}	5.35×10^{-8}	-5.44×10^{-9}	5.66×10^{-9}
-90.00	7.59×10^{-9}	?	?	-1.45×10^{-8}	8.82×10^{-9}	-1.21×10^{-9}	7.11×10^{-10}

(a) $\gamma = -5$

v	l	Monte Carlo		Thermal Integration: Log N		Thermal Integration: Chi	
		$l - \tilde{l}$	SE	$l - \tilde{l}$	SE	$l - \tilde{l}$	SE
-1.00	1.28×10^{-1}	1.17×10^{-3}	1.24×10^{-3}	3.06×10^{-3}	1.37×10^{-3}	-4.10×10^{-5}	1.43×10^{-3}
-2.00	2.77×10^{-2}	3.00×10^{-4}	6.75×10^{-4}	-1.68×10^{-2}	2.83×10^{-3}	-3.17×10^{-3}	1.60×10^{-3}
-3.00	6.04×10^{-3}	-6.31×10^{-4}	3.30×10^{-4}	-1.04×10^{-2}	1.57×10^{-3}	-8.04×10^{-4}	7.09×10^{-4}
-5.00	3.00×10^{-4}	-1.97×10^{-5}	6.86×10^{-5}	-3.98×10^{-3}	6.97×10^{-4}	-3.24×10^{-4}	1.17×10^{-4}
-6.00	6.81×10^{-5}	4.12×10^{-6}	3.10×10^{-5}	-1.41×10^{-3}	2.70×10^{-4}	-1.03×10^{-4}	7.61×10^{-5}
-8.00	3.58×10^{-6}	?	?	-2.87×10^{-4}	9.24×10^{-5}	-2.50×10^{-6}	3.58×10^{-6}
-9.00	8.31×10^{-7}	?	?	-1.03×10^{-4}	4.49×10^{-5}	-8.64×10^{-7}	1.19×10^{-6}

(b) $\gamma = 0$

[Table Continued Overleaf]

v	l	Monte Carlo		Thermal Integration: Log N		Thermal Integration: Chi	
		$l - \tilde{l}$	SE	$l - \tilde{l}$	SE	$l - \tilde{l}$	SE
0.50	3.07×10^{-2}	-3.66×10^{-4}	6.15×10^{-4}	6.05×10^{-4}	8.04×10^{-4}	-2.48×10^{-4}	9.52×10^{-4}
0.20	9.55×10^{-3}	1.30×10^{-4}	3.35×10^{-4}	1.25×10^{-3}	3.85×10^{-4}	-4.26×10^{-4}	4.08×10^{-4}
-0.20	8.37×10^{-4}	5.27×10^{-5}	1.24×10^{-4}	1.50×10^{-4}	9.97×10^{-5}	1.42×10^{-4}	1.28×10^{-4}
-0.50	7.39×10^{-5}	5.79×10^{-5}	1.60×10^{-5}	6.73×10^{-5}	4.63×10^{-6}	1.75×10^{-5}	3.99×10^{-5}
-0.80	4.83×10^{-6}	?	?	?	?	?	?
-1.00	7.16×10^{-7}	?	?	?	?	?	?

(c) $\gamma = 5$

Table 7.1: Performance of the adapted thermal integration algorithm for the two samplers. Results marked with a ? indicate the algorithm failed to calculate a prediction. Both algorithms performed very well compared to regular Monte Carlo for $\gamma \leq 0$. For $\gamma = 5$ the new methods estimate probabilities which are up to 10^{-4} times smaller in the cases where regular Monte Carlo has failed to generate any estimate. The closer fit of the Adjusted Chi sampler has led it to outperform the log normal sample by one or two orders of magnitude in most cases. l is the true likelihood and $\hat{l}$ is the likelihood estimate found using each method. SE is the standard error as predicted by the method.

estimate. The log-normal method has significant relative errors compared to the true value.

Table 7.1c ($\gamma = 5$) show similar performance to Monte Carlo for all likelihoods. This supports the hypothesis that our method is most effective when $\gamma \leq 0$.

For all choices of γ the standard error provides a reasonable estimate of the actual error. This is encouraging as it suggests there is no (major) systematic error in the methodology.

Overall, these empirical results are very encouraging. The areas where we would expect strong performance have shown a very clear performance jump over regular Monte Carlo. We still struggle to beat Monte Carlo when $\gamma > 0$, as predicted. When $\gamma > 0$ both distributions have a significant impact on rare events and so we can no longer disregard the normal distribution.

The Chi adjustment method has generally matched or exceeded regular Monte Carlo. This means it could safely be used anywhere we use Monte Carlo without performance degradation concerns.

The log-normal sampler doesn't perform as well and in some cases is worse than Monte Carlo. It is worth reiterating that the log normal sampler uses significantly less compute time to generate samples. However the overall impact of this may be minimal if the majority of the simulation time is spent evaluating V.

In later sections we will apply these methods to real world financial situations.

7.4 Example 2: Financial Stocks (one dimension)

In this section we consider three individual stocks. In contrast to the last chapter we consider logarithmic returns for our stocks. This is because we no longer require $V(x)$ to be a linear combination of the dimensions of the sample, in fact $V(x)$ can now take any functional form.

At this stage we also expand our horizon beyond NIG distributions. We fit the same three stocks as before to the Generalised Hyperbolic distribution, which is a significantly larger class of distributions and allows for an additional degree of freedom. The stocks are fitted using the same algorithm as before (MCECM). More information can be found in section 6.4. As established in the previous chapter, this is a well known and utilised fitting method for this type of distribution and has been used extensively in the finance literature. See [Breymann and Luthi, 2013] and [McNiel et al., 2005] for further information.

Stock data from 2010 is used to fit and estimate risk on three banks: Goldman Sachs (GS), Morgan Stanley (MS) and Credit Suisse (CS). As before, our comparison

Stock	χ	ψ	μ	σ	γ	λ
GS	3.1	3.2×10^{-4}	3.2×10^{-3}	1.8×10^{-2}	-3.1×10^{-3}	-2.6
MS	7.1	9.9	1.2×10^{-3}	2.1×10^{-2}	-1.5×10^{-3}	1.0
CS	4.0×10^{-6}	2.7	3.3×10^{-3}	2.4×10^{-2}	-4.1×10^{-3}	1.4

Table 7.2: The parameters (of the GH distribution) resulting from the data fitting to the various log stock returns using the MCECM algorithm [McNiel et al., 2005] and 2010 return data. The Morgan Stanley (MS) returns were fitted to a hyperbolic distribution as a valid GH fit could not be found.

is to regular Monte Carlo. The validity of these models at different risk levels is beyond the scope of this work.

Table 7.2 shows the resulting parameters from the fitting. Unfortunately we could not fit the MS returns to a Generalised Hyperbolic distribution and so they were instead fitted to a Hyperbolic distribution ($\lambda = 1$). We used the same settings as previously outlined in the method (7.2.5) and preceding example (6.3).

The results for the financial stocks are displayed in table 7.3. They generally mirror those of the original test case. The Chi adjustment method always matches the performance of Monte Carlo for larger probabilities, but also shows good results for probabilities one to three magnitudes smaller than those which Monte Carlo can reach.

The Chi adjustment method was particularly effective for the CS stock. This could be due to the higher λ in this case, leading to heavier tails and therefore exacerbating the effects we are exploiting.

The results for the extreme probabilities do have a high relative error (up to 80%) however this is still a significant improvement over Monte Carlo.

The log normal sampler is indicative that areas of risk exist, where Monte Carlo shows zero risk, however it has huge errors at certain points. These errors are likely in part due to numerical errors from the calculation of the various weighted expectations. This suggests that the log normal and Generalised Inverse Gaussian may not have as significant an overlap as hoped once we move into the risk tail.

7.5 Example 3: Financial Portfolio (n dimensions)

In this final results section we apply the method to a portfolio of stocks. We consider a portfolio of three stocks whose logarithmic returns are modelled using a three dimensional generalised hyperbolic distribution.

The portfolio contained 25% of both GS and MS stocks and 50% CS stock.

v	l	Monte Carlo		Thermal Integration: Log N		Thermal Integration: Chi	
		$l - \tilde{l}$	SE	$l - \tilde{l}$	SE	$l - \tilde{l}$	SE
0.05	9.94×10^{-1}	2.06×10^{-4}	3.38×10^{-4}	-3.06×10^{-4}	3.41×10^{-4}	1.90×10^{-4}	3.26×10^{-4}
-0.02	1.10×10^{-1}	-1.46×10^{-3}	9.47×10^{-4}	-1.71×10^{-3}	1.33×10^{-3}	7.86×10^{-4}	1.36×10^{-3}
-0.06	5.83×10^{-3}	-7.32×10^{-4}	3.44×10^{-4}	-3.22×10^{-2}	4.02×10^{-3}	-1.14×10^{-3}	6.55×10^{-4}
-0.10	8.45×10^{-4}	-3.09×10^{-6}	1.08×10^{-4}	-1.78×10^{-2}	3.30×10^{-3}	-4.43×10^{-4}	1.22×10^{-4}
-0.18	8.19×10^{-5}	1.79×10^{-5}	3.85×10^{-5}	-5.79×10^{-3}	1.92×10^{-3}	-6.59×10^{-5}	1.76×10^{-5}
-0.34	6.55×10^{-6}	-9.45×10^{-6}	1.60×10^{-5}	-2.44×10^{-3}	1.61×10^{-3}	-8.37×10^{-6}	2.71×10^{-6}
-0.54	9.32×10^{-7}	?	?	-1.02×10^{-3}	4.84×10^{-4}	-2.43×10^{-7}	4.26×10^{-7}

(a) Goldman Sachs

v	l	Monte Carlo		Thermal Integration: Log N		Thermal Integration: Chi	
		$l - \tilde{l}$	SE	$l - \tilde{l}$	SE	$l - \tilde{l}$	SE
0.05	9.89×10^{-1}	3.72×10^{-4}	5.09×10^{-4}	2.60×10^{-4}	4.25×10^{-4}	-3.32×10^{-4}	4.61×10^{-4}
-0.01	3.14×10^{-1}	9.43×10^{-4}	1.77×10^{-3}	-1.81×10^{-3}	2.00×10^{-3}	2.09×10^{-3}	2.06×10^{-3}
-0.03	7.39×10^{-2}	-5.18×10^{-4}	9.66×10^{-4}	-1.24×10^{-4}	2.30×10^{-3}	7.98×10^{-4}	2.27×10^{-3}
-0.06	3.04×10^{-3}	2.23×10^{-4}	2.21×10^{-4}	-4.70×10^{-3}	1.57×10^{-3}	-3.20×10^{-4}	4.67×10^{-4}
-0.08	2.30×10^{-4}	7.02×10^{-5}	4.57×10^{-5}	-6.18×10^{-4}	2.87×10^{-4}	-2.23×10^{-5}	1.11×10^{-4}
-0.10	1.38×10^{-5}	-2.25×10^{-6}	1.60×10^{-5}	-1.03×10^{-4}	6.80×10^{-5}	-3.27×10^{-7}	9.73×10^{-6}
-0.11	3.13×10^{-6}	?	?	-3.06×10^{-7}	3.44×10^{-6}	2.52×10^{-6}	6.14×10^{-7}

(b) Morgan Stanley

[Table Continued Overleaf]

v	l	Monte Carlo		Thermal Integration: Log N		Thermal Integration: Chi	
		$l - \tilde{l}$	SE	$l - \tilde{l}$	SE	$l - \tilde{l}$	SE
0.05	9.80×10^{-1}	-1.89×10^{-4}	5.63×10^{-4}	-5.25×10^{-4}	5.45×10^{-4}	1.11×10^{-3}	5.45×10^{-4}
0.01	7.06×10^{-1}	-4.99×10^{-4}	1.81×10^{-3}	1.32×10^{-3}	2.01×10^{-3}	-1.57×10^{-3}	1.68×10^{-3}
-0.03	9.53×10^{-2}	-2.54×10^{-4}	1.14×10^{-3}	-3.62×10^{-4}	1.53×10^{-3}	3.95×10^{-3}	1.44×10^{-3}
-0.07	8.24×10^{-3}	3.34×10^{-5}	3.03×10^{-4}	-1.00×10^{-2}	1.29×10^{-3}	-8.03×10^{-4}	7.03×10^{-4}
-0.11	5.93×10^{-4}	2.57×10^{-4}	7.26×10^{-5}	-3.98×10^{-3}	5.75×10^{-4}	1.27×10^{-5}	7.95×10^{-5}
-0.15	3.64×10^{-5}	2.04×10^{-5}	1.60×10^{-5}	-7.63×10^{-4}	1.38×10^{-4}	-1.99×10^{-5}	1.72×10^{-5}
-0.19	1.91×10^{-6}	?	?	-9.32×10^{-5}	2.50×10^{-5}	-3.38×10^{-6}	4.60×10^{-6}
-0.22	1.88×10^{-7}	?	?	-2.36×10^{-5}	1.04×10^{-5}	-4.40×10^{-7}	6.28×10^{-7}

(c) Credit Suisse

Table 7.3: Performance of the adapted thermal integration algorithm for the two samplers on financial test cases. Results marked with a ? indicate the algorithm failed to calculate a prediction. Both algorithms performed well. They were both able to estimate significantly smaller likelihoods than standard Monte Carlo. The Chi adjusted sampler however significantly outperformed the log normal sampler in terms of accuracy.

This creates a portfolio which is balanced across two continents and so should be somewhat hedged. In this case the stocks were fitted to a multivariate Generalised Hyperbolic distribution. The MCECM algorithm was used to fit the returns to the distribution. The resulting parameters where $\chi = 2.5, \lambda = -2.1, \psi = 0.08$,

$$\mu = \begin{pmatrix} 1.4 \\ -1.3 \\ 3.2 \end{pmatrix} 10^{-3}, \gamma = \begin{pmatrix} -1.2 \\ 1.2 \\ -3.7 \end{pmatrix} 10^{-3} \text{ and } \Sigma = \begin{pmatrix} 3.5 & 3.1 & 2.6 \\ 3.1 & 5.0 & 3.8 \\ 2.6 & 3.8 & 6.1 \end{pmatrix} 10^{-4}.$$

The results for the portfolio of stocks are displayed in table 7.4. The results yet again show a strong performance by the new algorithm and generally reflect those of the individual stocks.

Both samplers explore events significantly further into the tail than regular Monte Carlo. The Chi adjustment method however has significantly smaller errors. We again note the high absolute value of λ in this case, which leads to heavier tails and therefore exacerbates the effects we are exploiting.

7.6 Discussion

In this chapter we have developed and explored two new methods for estimating the likelihood of rare events. We have applied these methods to a simple test case and two differing financial examples.

The Chi adjustment method has proved consistently effective. It has always matched performance of Monte Carlo (MC) sampling and has often significantly surpassed the performance of MC methods. This is particularly true for low probabilities where regular MC has failed to get any estimate for the probability.

The log-normal method has not shown as good a performance in terms of predictive power, but has been able to detect regions where some probability mass exists. In these regions MC methods showed there was no probability weight.

The challenge with this class of methods is in finding an effective way of generating sequential importance samplers. If the samplers are not a good fit to the distribution being searched (in this case GH/GIG distributions) it will struggle to estimate the conditional probabilities. Furthermore, if two (or more) of the sequential samplers do not overlap we will get significant errors in our estimation process as the conditional probability estimate will not be accurate.

An interesting result has been the ability of both methods to indicate which areas of the state space do contain weight. This information alone could potentially help guide a risk analyst when considering a given problem.

v	l	Monte Carlo		Thermal Integration: Log N		Thermal Integration: Chi	
		$l - \tilde{l}$	SE	$l - \tilde{l}$	SE	$l - \tilde{l}$	SE
0.05	9.89×10^{-1}	2.49×10^{-4}	4.67×10^{-4}	4.13×10^{-5}	4.21×10^{-4}	3.45×10^{-4}	4.36×10^{-4}
0.01	7.24×10^{-1}	3.10×10^{-6}	2.03×10^{-3}	1.01×10^{-3}	1.89×10^{-3}	-3.49×10^{-4}	2.02×10^{-3}
-0.03	5.76×10^{-2}	-9.02×10^{-4}	1.02×10^{-3}	-2.35×10^{-2}	2.33×10^{-3}	7.19×10^{-4}	1.50×10^{-3}
-0.06	7.30×10^{-3}	-1.71×10^{-4}	3.25×10^{-4}	-2.96×10^{-2}	3.49×10^{-3}	-2.09×10^{-3}	6.52×10^{-4}
-0.10	9.33×10^{-4}	8.51×10^{-5}	1.01×10^{-4}	-1.73×10^{-2}	2.53×10^{-3}	-6.18×10^{-4}	1.18×10^{-4}
-0.16	9.25×10^{-5}	-1.95×10^{-5}	3.97×10^{-5}	-4.38×10^{-3}	1.10×10^{-3}	-7.37×10^{-5}	2.55×10^{-5}
-0.24	7.63×10^{-6}	?	?	-2.20×10^{-3}	8.20×10^{-4}	-6.87×10^{-6}	5.44×10^{-6}
-0.34	4.53×10^{-7}	?	?	-8.78×10^{-4}	2.21×10^{-4}	8.73×10^{-8}	3.66×10^{-7}
-0.44	2.64×10^{-8}	?	?	-3.54×10^{-4}	1.04×10^{-4}	-4.23×10^{-8}	6.87×10^{-8}

Table 7.4: Performance of the adapted thermal integration algorithm for the two samplers on a portfolio test case. Results marked with a ? indicate the algorithm failed to calculate a prediction. Both algorithms performed well. They were both able to estimate significantly smaller likelihoods than standard Monte Carlo. The chi adjusted sampler however significantly outperformed the log normal sampler in terms of accuracy.

We believe this extension of thermal integration, alongside the associated importance samplers, to be a new and novel method for estimating extreme risk in the Generalised Hyperbolic class of distributions.

In this chapter we have relied on an underlying understanding of how the process worked. However, a sufficiently convoluted choice of value function could break this intuition and, as we have seen, the intuition does not hold for all parameter choices.

An interesting extension of this method would be to consider using different q_i for each step. It is plausible that the early iterations could reliably use a smaller quantile (q_i), increasing the algorithm's 'reach'. Coupled with this idea, we could use a different number of samples for each step, possibly driven by current error estimates. This would allow us to target extra samples to the sampler that has the largest contribution to the overall (estimated) error.

In the conclusion (chapter 8) we will consider some further ideas on how to estimate the likelihood of extreme events when there is no underlying intuition to guide us.

Chapter 8

Conclusion

As promised at the start of this thesis, we have discussed problems relating to both quantile optimisation and reverse stress testing. In this chapter we will not only review the work completed thus far, but also look ahead to questions as yet unanswered and consider the potential for future work.

In chapters 3, 4 and 5 we constructed a variety of new algorithms for solving several different styles of quantile optimisation problem. These enabled the optimisation of a system at either a single quantile (as in chapter 3), at the worst case scenario (chapter 4) or at multiple quantiles (chapter 5). We underpinned the design of each algorithm with an analytical foundation. The algorithms were then successfully tested on real world financial problems. Chapters 3 and 5 focused on the development of the proposed optimisation process. In contrast, chapter 4 focused on how the optimisation process would actually behave under different conditions. One of the more surprising results from these chapters was the bad performance of BF400 in chapters 4 and 5. Our naive (but generally accepted) assumption was that 400 samples should be sufficient to reliably optimise a system at 95% risk, however this was (numerically) proved to not be the case. This failure of BF400 highlights the importance of this work.

Whilst good algorithms were constructed in each of these chapters, little discussion was given to how these algorithms could be developed and further improved. The most obvious adaptation would be to consider adding a form of either simulated tempering or parallel tempering to the algorithm. Both methods are more likely to find a global maximum than regular simulated annealing. Simulated tempering uses a single chain that varies temperature (β^{-1}) as it progresses (according to some preset probability distribution). Parallel tempering uses multiple chains that are run at different temperatures. These chains then have the opportunity to swap designs

between themselves. The advantage of parallel tempering is that it is easier to calculate the correct swap probabilities (between chains) and hence preserve detailed balance without having to use an arbitrary distribution for temperature. Both of these methods could be applied either directly or in variants where the level of risk or the number of samples is varied. Each of these adaptations should increase mobility of the optimisation method and help ensure the best global solution is found. However, for these methods to work we would have to consider how to reduce the number of samples (n) used to estimate $V_D(R_T)$. Currently we only change n by increasing it if $\langle R_k \rangle$ becomes too large.

An additional extension would be to consider an adaptation of the ideas of [Calafiore and Campi, 2005] to this problem. [Calafiore and Campi, 2005] suggest using a fixed set of samples throughout the simulation (in the context of convex optimisation). This reduces the noisy optimisation problem to a deterministic one. [Calafiore and Campi, 2005] provides results to calculate the precise number of samples needed in the convex case but since we do not operate on that domain careful thought would be required as to how best select the number of samples required. It would be further interesting if it was still possible to increase the samples required as the simulation progressed so as to reduce the computational burden. Such a method would also reduce the biasing effect on $\mathbf{X}$ of the simulation.

One of the ongoing questions we considered was how to minimise the number of samples required by the optimisation methods. This was so that raw historical data could be used directly without the added complication of fitting a model. Unfortunately, almost all the methods required a significant number of samples to operate. The one exception was the example used in section 4.6 where we notionally optimised at 90% risk, but in fact optimised at 95% risk. In this example around 600 samples were needed, which in financial problems is approximately 2.5 years of historical data. It would be nice to reduce this further but it is not clear, at least yet, how this could be done. That said, we could reduce the number of samples required in chapter 5 by applying some more of the ideas from chapter 4 to the algorithms (for example, by fixing k or j equal to 1).

We also note that the estimators for $\langle R_k \rangle$ developed in chapter 4 could be further utilised. Using the method described in that chapter we actually find a description for the entire risk curve. This could be utilised in various ways. For example, it could be used in the context of chapter 5 to better control how we change k when we are optimising two quantiles. This is because it should allow for a reasonably quick estimate of $\frac{d\langle R_k \rangle}{dk}$. Linked to this, it would be interesting if this estimate of the whole risk curve could be used to find a way of optimising a

combination of μ_D and $V_D(R_T)$, for some choice of R_T, which counteracted any bias induced by the $e^{\beta(\mu_D+V_D(R_k))}$ term. This could be trivially done if F_D was known to be symmetric.

We successfully applied the described algorithms to a variety of financial problems. That said, it would be interesting to consider the application of the algorithms to other problems. One particular example is that of optimising how to provide for a service level agreement. This was originally discussed in section 3.8 as being infeasible, however by using the tools developed in chapter 5 it should now be possible to find a suitable methodology for solving this class of problem. This would allow the work to be applied to a far greater selection of situations.

Finally, we note that a significant amount of this work has been based on fitting complex systems to various models and then using these models to make observations or decisions about the system. Discussion on the validity of the models has generally been considered beyond the scope of this work; however, it is worth briefly mentioning concerns around overfitting, specifically in terms of the various covariance matrices. This problem has been highlighted in biology where a small amount of gene sequencing data was often used to fit a huge covariance matrix [Felsenstein, 1988]. Any results determined from such a matrix are highly likely to be biased by the samples used to fit the model. There is an open question if we too have pushed the models further than they are capable of going during the course of this work. In the optimisation of quantile risk work this question sits around the variance. Have we pushed so far into the tail that the variance of the system is strongly biased by the samples used to fit the original data model? The second, more interesting, question is can we stop, or even harness, this bias to better find a solution? It may be possible to incorporate an additional layer of error awareness into the optimisation methods we have developed, in order to increase the weight in areas of the model that have little data.

In chapters 6 and 7 we focused on the estimation of the likelihood of extreme events. We applied the resulting algorithms to reverse stress testing problems. This was done using intuition into the statistical models used. The methods proved to be very successful under the correct conditions when compared to Monte Carlo methods.

In both chapters it would have been nice to get a firmer analytic handle on when the methods would work best and the likely errors they would each induce. Unfortunately, the complexities of doing this proved to be too great and so the results were largely numerically driven. It would be nice to add an improved analytic framework to this style of problem, but it is hard to see how to do this with a

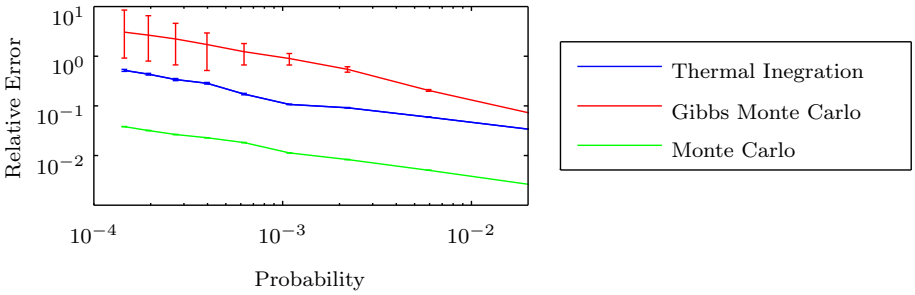


Figure 8.1: The above figure shows the impact of using an algorithm similar to that of chapters 6 and 7 which biases the distribution at each iteration using a thermal weighting. The test problem is the sum of two identical Pareto distributions. The chart shows the relative error in estimating the likelihood of this sum exceeding a certain value for each method. The errors are plotted against the true likelihood. Each method used the same number of samples. The results show that the adapted method described above performs well versus a Monte Carlo method already reliant on a Gibbs sampler (Gibbs Monte Carlo), but not versus a Monte Carlo method which has access to a more efficient (distribution specific) sampler.

significant number of simplifying assumptions.

We also note that in both methods we could have been smarter in how we used the budgeted number of samples. It would have been relatively trivial to run an initial likelihood estimate to better tune the different distributions used in the final estimation. Alternatively, an initial sample could be found and then extra samples targeted at the distribution which most contributed to the statistical error of the overall likelihood estimate. Coupled with this, once all the samples have been taken it should be possible to optimise the placement of the transition points (q_i) to minimise the statistical error.

Both of these methods also relied on an intuitive understanding of the system. It is possible to forgo this assumption by using thermal distributions. If one wished to find the likelihood of an extreme outcome of a financial portfolio then one could bias the sequential estimators using the value of the portfolio, e.g. one could construct a sequence of distributions biased by $e^{\beta_1 V(x)}, \ldots, e^{\beta_n V(x)}$. This would mean we would no longer require intuition into how the underlying system works. That said, such a method might struggle if the likelihood of exceeding a certain value is actually split across multiple regions of the search space. To combat this, parallel tempering (or similar) could be used to ensure good likelihood estimates at each temperature level. As before, each choice of β could be calculated by the method, making it entirely self tuning. The second problem with such a method is that it

forces the use of a Gibbs sampler (or similar) to generate random samples. This is in contrast to chapters 6 and 7, where we were able to use distributions for our biased likelihood estimators that we could easily draw independent samples from. Unfortunately, a Gibbs sampler is a relatively inefficient method of drawing samples when compared to a distribution specific sampler. Figure 8.1 shows the resulting tension for a very simple problem. We can see that while this adapted method, which uses this sequence of thermal biases, comfortably beats a Monte Carlo estimator which already has to draw samples using a Gibbs sampler, it is beaten by a Monte Carlo sampler that has an alternative (easier) way of drawing samples. It would be a nice extension to see if either; given a sufficient number of dimensions this variant of thermal integration eventually beats regular Monte Carlo, or if such a method could be adapted to simply warn of dangerous regions more efficiently than regular Monte Carlo.

Appendix A | Annealing Algorithms Used

Algorithm	Estimators Used				n			k
	$\hat{R}_k$	$\hat{R}_k^2$	$\hat{R'}_k$	$\hat{R'}_k^2$	Initial Value	When Incremented	Increment Amount	
SA1	-	-	-	-	Varies	After fixed number of steps	Fixed per problem	$k = \max(\lfloor nR_T \rfloor, 1)$
SA2	✓	-	-	-	Varies	When $\hat{R}_k > R_T$	Fixed per problem	$k = \max(\lfloor nR_T \rfloor, 1)$
SA3	✓	✓	-	-	Varies	When $\hat{R}_k > R_T$	Dependant on $\frac{d\langle \hat{R}_k \rangle}{dn}$	$k = \max(\lfloor nR_T \rfloor, 1)$
WSA1	-	-	-	-	Varies	After fixed number of steps	Fixed per problem	$k = 1$
WSA2	✓	-	-	-	Varies	When $\hat{R}_k > R_T$	Fixed per problem	$k = 1$
WSA3	✓	✓	-	-	Varies	When $\hat{R}_k > R_T$	Dependant on $\frac{d\langle \hat{R}_k \rangle}{dn}$	$k = 1$
qSA2	-	-	✓	-	Varies	When $\hat{R'}_k > R_T$	Fixed per problem	$k = \max(\lfloor nR_T \rfloor, 1)$
qSA3	-	-	✓	✓	Varies	When $\hat{R'}_k > R_T$	Dependant on $\frac{d\langle \hat{R'}_k \rangle}{dn}$	$k = \max(\lfloor nR_T \rfloor, 1)$
qWSA2	-	-	✓	-	Varies	When $\hat{R'}_k > R_T$	Fixed per problem	$k = 1$
qWSA3	-	-	✓	✓	Varies	When $\hat{R'}_k > R_T$	Dependant on $\frac{d\langle \hat{R'}_k \rangle}{dn}$	$k = 1$
BF	-	-	-	-	$\lceil R_T^{-2} \rceil$	Never	n/a	$k = \max(\lfloor nR_T \rfloor, 1)$
BF10000	-	-	-	-	10000	Never	n/a	$k = \max(\lfloor nR_T \rfloor, 1)$

Table A.1: The above table shows the different algorithms used during chapters 3 and 4. R_T is the target value of risk which is being optimised. n is the number of samples used to estimate $V_D(R)$. $\hat{R}_k$, $\hat{R}_k^2$ and $\frac{d\langle \hat{R}_k \rangle}{dn}$ are defined in chapter 3. $\hat{R'}_k$, $\hat{R'}_k^2$ and $\frac{d\langle \hat{R'}_k \rangle}{dn}$ are defined in chapter 4.

Bibliography

E. Anderson and M. Ferris. A direct search algorithm for optimization with noisy function evaluations. *SIAM J. Optim.*, 11 (3):837–857, 2001.

Soren Asmussen and Peter W. Glynn. *Stochastic Simulation: Algorithms and Analysis*. Springer, 2007.

Adriano Azevedo-Filho and Ross D. Shachter. Laplaces method approximations for probabilistic inference in belief networks with continuous variables. *Uncertainty in Artificial Intelligence*, pages 28–36, 1994.

Robin C. Ball, Thomas M. A. Fink, and Neill E. Bowler. Stochastic annealing. *Phys. Rev. Lett.*, 91:030201, Jul 2003. doi: 10.1103/PhysRevLett.91.030201. URL `http://link.aps.org/doi/10.1103/PhysRevLett.91.030201`.

Basel III: A global regulatory framework for more resilient banks and banking systems, Bank for International Settlements Communications CH-4002 Basel, Switzerland, 2011. Bank for International Settlements.

O. Barndorff-Nielsen. Exponentially decreasing distributions for the logarithm of particle size. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical*, 353:401–419, 1977.

O.E. Barndorff-Nielsen and D.R. Cox. *Asymptotic Techniques for Use in Statistics*. Chapman & Hall, 1989.

O.E. Barndorff-Nielsen, J. Kent, and M. Srensen. Normal variance-mean mixtures and z-distributions. *International Statistical Review*, 50:145159, 1982.

Dimitris Bertsimas and John Tsitsiklis. Simulated annealing. *Statistical Science*, 8 (1):10–15, 02 1993. doi: 10.1214/ss/1177011077. URL `http://dx.doi.org/10.1214/ss/1177011077`.

Macro Better, Fred Glover, Gary Kochenberger, and Haibo Wang. Simulation optimization: Applications in risk management. *International Journal of Information Technology and Decision Making*, 7:571–587, 2008.

Hans-Georg Beyer and Bernhard Sendhoff. Robust optimization a comprehensive survey. *Computer Methods in Applied Mechanics and Engineering*, 196 (3334):3190 – 3218, 2007. ISSN 0045-7825. doi: http://dx.doi.org/10.1016/j.cma.2007.03.003. URL `http://www.sciencedirect.com/science/article/pii/S0045782507001259`.

N. H. Bingham, S. Bond, A. Fritsche, S. Hwang, G. Jiang amd R. Kiesel, J. Knight, C. Lizieri, C. Rodgers, S. Satchell, G. Wang, C. Ward, Y. Yoon, and J. Yu. *Return Distributions in Finance*. Butterworth-Heinemann Finance, 2001.

Thomas Breuer, Martin Jandacka, Klaus Rheinburger, and Martin Summer. Does adding up of economic capital for market and credit risk amount to conservative risk assement? *Journal of Banking and Finance*, 34:703–712, 2010.

Wolfgang Breymann and David Luthi. ghyp: A package on generalized hyperbolic distributions. Technical report, Institute of Data Analysis and Process Design, 2013.

Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Gener. Comput. Syst.*, 25(6): 599–616, June 2009. ISSN 0167-739X. doi: 10.1016/j.future.2008.12.001. URL `http://dx.doi.org/10.1016/j.future.2008.12.001`.

Giuseppe Calafiore and M.C. Campi. Uncertain convex programs: randomized solutions and confidence levels. *Math. Program., Ser. A*, 102:25–46, 2005.

V. Cerny. Thermodynamical approach to the travelling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45(1):41–51, 1985. ISSN 0022-3239. doi: 10.1007/BF00940812. URL `http://dx.doi.org/10.1007/BF00940812`.

Joshua C.C. Chan and Dirk P. Kroese. Rare-event probability estimation with conditional monte carlo. *Annals of Operations Research*, 189:43–61, 2011. ISSN 0254-5330. doi: 10.1007/s10479-009-0539-y. URL `http://dx.doi.org/10.1007/s10479-009-0539-y`.

Ding-Jun Chen, Chung-Yeol Lee, Cheol-Hoon Park, and Pedro Mendes. Parallelizing simulated annealing algorithms based on high-performance computer. *J Glob Optim*, 39:261–289, 2007.

Domenico Cuocoa and Hong Liu. An analysis of var-based capital requirements. *Journal of Financial Intermediation*, 15:362–394, 2006.

W. Edwards Deming. *Sample Design in Business Research*. John Wiley & Sons, 1990.

Luc Devroye. *Non-Uniform Random Variate Generation*. Springer-Verlag., 1986.

Ernst Eberlein and Karsten Prause. The generalized hyperbolic model: Financial derivatives and risk measures. In *MATHEMATICAL FINANCE BACHELIER CONGRESS 2000, GEMAN*, pages 245–267, 1998.

A.E. Eiben and J.E. Smith. *Introduction to Evolutionary Computing*. Springer, Natural Computing Series, 2 edition, 2007.

Mohammed El-Beltagy, Prasanth B. Nair, and Andy J. Keane. Metamodeling techniques for evolutionary optimization of computationally expensive problems: Promises and limitations. In *Conference: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 1999), 13-17 July 1999, Orlando, Florida, USA*, 1999.

Ernst Enerlein and Ulrich Keller. Hyperbolic distributions in finance. *Bernoulli*, 3: 281–299, 1995.

William Feller. *An Introduction to Probability Theory and Its Applications*. Wiley, 1968.

Joseph Felsenstein. Phylogenies from molecular sequences: Inference and reliability. *Annual Review of Genetics*, 22:251–265, 1988.

K V Fernando. Practical portfolio optimization. Technical report, NAG Ltd, 2000.

P. D. Finch. A limit theorem for markov chains with continuous state space. *Journal of the Australian Mathematical Society*, 3:351–358, 8 1963. ISSN 1446-8107. doi: 10.1017/S1446788700028378. URL `http://journals.cambridge.org/article_S1446788700028378`.

Daniel Frenkel and Berend Smit. *Understanding Molecular Simulation: From Algorithms to Applications*. Academic Press, 2002.

F.S.A. Policy statement 09/20: Stress and scenario testing, 2009. URL http://www.fsa.gov.uk/pubs/policy/ps09_20.pdf.

Jitesh S.B. Gajjar. Laplace's method, 2010. URL http://www.maths.manchester.ac.uk/~gajjar/MATH44011/notes/44011_note3.pdf.

Guillermo Gallego and Garrett Van Ryzin. Optimal dynamic pricing of inventories with stochastic demand over finite horizons. *Management Science*, 40(8):pp. 999–1020, 1994. ISSN 00251909. URL http://www.jstor.org/stable/2633090.

Dani Gamerman and Hedibert F. Lopes. *Markov Chain Monte Carlo*. Chapman & Hall, 2006.

ALAN E. Gelfand and ADRIAN F. M. Smith. Gibbs sampling for marginal posterior expectations. Technical report, THE OFFICE OF NAVAL RESEARCH, November 1991. Report Number 448.

S. B. Gelfand and S. K. Mitter. Simulated annealing with noisy or imprecise energy measurements. *Journal Of Optimisation Theory and Applications*, 62, 1989.

W.R. Gilks, S. Richardson, and D.J. Spiegelhalter, editors. *Markov Chain Monte Carlo in Practice*. Chapman & Hall, 1996.

Manfred Gilli and Evis Kellezi. An applicaton of extreme value theory for measuring financial risk. *Computational Economics*, 27:1–23, 2006.

P. Glasserman. *Monte Carlo Methods in Financial Engineering*. Springer, 2004.

P. Glasserman, C. Kang, and W. Kang. Stress scenarion selection by empirical likelihood. *SSRN*, June 2012. URL http://papers.ssrn.com/sol3/papers.cfm?abstract_id=2101465.

Peter Grundke. Reverse stress tests with bottom-up approaches. *The Journal of Risk Model Validation*, 5(1):71–90, Spring 2011.

Peter Grundke. Further recipes for quantitative reverse stress testing. *The Journal of Risk Model Validation*, 6(2):81–102, Summer 2012.

J.Q. Guoa and L. Zhengb. A modified simulated annealing algorithm for estimating solute transport parameters in streams from tracer experiment data. *Environmental Modelling & Software*, 20:811–815, 2005.

WALTER J. Gutjahr and GEORG CH. Pflug. Simulated annealing for noisy cost functions. *Journal of Global Optimization*, 8:1–13, 1996.

Darrall Henderson, SheldonH. Jacobson, and AlanW. Johnson. The theory and practice of simulated annealing. In Fred Glover and GaryA. Kochenberger, editors, *Handbook of Metaheuristics*, volume 57 of *International Series in Operations Research & Management Science*, pages 287–319. Springer US, 2003. ISBN 978-1-4020-7263-5. doi: 10.1007/0-306-48056-5_10. URL http://dx.doi.org/10.1007/0-306-48056-5_10.

Wenbo Hu and Alec Kercheval. Risk management with generalized hyperbolic distributions. In *Proceedings of the Fourth IASTED International Conference on Financial Engineering and Applications*, FEA '07, pages 19–24, Anaheim, CA, USA, 2007. ACTA Press. ISBN 978-0-88986-681-2. URL http://dl.acm.org/citation.cfm?id=1650320.1650324.

J. L. W. V. Jensen. Sur les fonctions convexes et les ingalits entre les valeurs moyennes. *Acta Mathematica*, 30:175–193, 1906.

John Kelly. A new interpretation of information rate. *Bell System Technical Journal*, 35:917–926, 1956.

S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983. doi: 10.1126/science.220.4598.671. URL http://www.sciencemag.org/content/220/4598/671.abstract.

Petter N. Kolm, Reha Ttnc, and Frank J. Fabozzi. 60 years of portfolio optimization: Practical challenges and current trends. *European Journal of Operational Research*, 234(2):356 – 371, 2014. ISSN 0377-2217. doi: http://dx.doi.org/10.1016/j.ejor.2013.10.060. URL http://www.sciencedirect.com/science/article/pii/S0377221713008898. 60 years following Harry Markowitzs contribution to portfolio theory and operations research.

M. Kolonko. Some new results on simulated annealing applied to the job shop scheduling problem. *European Journal of Operational Research*, 113(1):123 – 136, 1999. ISSN 0377-2217. doi: http://dx.doi.org/10.1016/S0377-2217(97)00420-7. URL http://www.sciencedirect.com/science/article/pii/S0377221797004207.

Faming Liang, Chuanhai Liu, and Raymond J. Carroll. *Advanced Markov Chain Monte Carlo Methods*. Wiley, 2010.

Elisa Luciano and Patrizia Semeraro. A generalized normal mean variance mixture for return processes in finance. Carlo Alberto Notebooks 97, Collegio Carlo Alberto, 2008. URL http://ideas.repec.org/p/cca/wpaper/97.html.

Harry Markowitz. Portfolio selection. *Journal of Finance*, 7:77–91, 1952.

Mario Mazzocchi. *Statistics for Marketing and Consumer Research*. SAGE, 2008.

Alexander J. McNiel, Rudiger Frey, and Paul Embrechts. *Quantative Risk Management*. Princeton University Press, 2005.

Robert Minlos. *Introduction to Mathematical Statistical Physics*, volume 19 of *University Lecture Series*. American Mathematical Society, 1999.

Claudia Neves and M. Isabel Fraga Alves. Testing extreme value conditions - an overview and recent approaches. *REVSTAT*, 6:83–100, 2008.

Yaghout Nourani and Bjarne Andresen. A comparison of simulated annealing cooling strategies. *Journal of Physics A: Mathematical and General*, 31(41):8373, 1998. URL `http://stacks.iop.org/0305-4470/31/i=41/a=011`.

W. Olbricht. On mergers of distributions and distributions with exponential tails. *Computational Statistics & Data Analysis*, 12:315–326, 1991.

Li Lian Ong, Rodolfo Maino, and Nombulelo Duma. Into the great unknown: Stress testing with weak data. Technical report, IMF Working Paper, 2010.

Mohnish Pabrai. *The Dhandho Investor: The Low-Risk Value Method to High Returns*. Wiley, 2007.

M. Papadrakakis, N. Lagaros, and V. Plevris. Design optimization of steel structures considering uncertainties. *Engrg. Struct.*, 27 (9):1408–1418, 2005.

Rgis Pouillot, Nicolas Miconnet, Anne-Laure Afchain, Marie Laure Delignette-Muller, Annie Beaufort, Laurent Rosso, Jean-Baptiste Denis, and Marie Cornu. Quantitative risk assessment of listeria monocytogenes in french cold-smoked salmon: I. quantitative exposure assessment. *Risk Analysis*, 27(3):683–700, 2007. ISSN 1539-6924. doi: 10.1111/j.1539-6924.2007.00921.x. URL `http://dx.doi.org/10.1111/j.1539-6924.2007.00921.x`.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical recipes : the art of scientific computing*. Cambridge University Press, 2007.

Riccardo Rebonato. *Coherent Stress Testing*. Wiley, first edition, 2010.

H. Robbins and S. Monro. A stochastic approximation method. *Ann. Math. Statist.*, 29:400–407, 1951.

Gareth O. Roberts and Jeffrey S. Rosenthal. General state space markov chains and mcmc algorithms. *Probability Surveys*, 1:20–71, 2004. doi: 10.1214/154957804100000024. URL http://dx.doi.org/10.1214/154957804100000024.

Reuven Y. Rubinstein and Peter W. Glynn. How to deal with the curse of dimensionality of likelihood ratios in monte carlo simulation. *Stochastic Models*, 25: 547–568, 2009.

Reuven Y. Rubinstein and Dirk P. Kroese. *Simulation and the Monte Carlo Method*. Wiley, second edition, 2008.

S. Samaranayake, S. Blandin, and A. Bayen. A tractable class of algorithms for reliable routing in stochastic networks. *Transportation Research Part C: Emerging Technologies*, 20(1):199 – 217, 2012. ISSN 0968-090X. doi: 10.1016/j.trc.2011.05.009. URL http://www.sciencedirect.com/science/article/pii/S0968090X11000799.

Rafael Schmidt, Tomas Hrycejc, and Eric Sttzlec. Multivariate distribution models with generalized hyperbolic margins. *Computational Statistics & Data Analysis*, 50:2065–2096, 2006.

Til Schuermann. Stress testing banks. Technical report, Wharton Financial Institutions Center, 2012.

Jimmy Skoglund and Wei Chen. Risk contributions, information and reverse stress testing. *The Journal of Risk Model Validation*, 3(2):61–77, Summer 2009.

PhilipN. Strenski and Scott Kirkpatrick. Analysis of finite length annealing schedules. *Algorithmica*, 6(1-6):346–366, 1991. ISSN 0178-4617. doi: 10.1007/BF01759050. URL http://dx.doi.org/10.1007/BF01759050.

Nassim Nicholas Taleb. *The Black Swan: The Impact of the Highly Improbable*. Penguin, 2007.

Jonathan M. Thompson and Kathryn A. Dowsland. A robust simulated annealing based examination timetabling system. *Computers & Operations Research*, 25(78):637 – 648, 1998. ISSN 0305-0548. doi: http://dx.doi.org/10.1016/S0305-0548(97)00101-9. URL http://www.sciencedirect.com/science/article/pii/S0305054897001019.

Johannes H. Venter and Pieter J. de Jongh. Risk estimation using the normal inverse gaussian distribution. *Journal of Risk*, 2002.

Long Wen, Liang Gao, Xinyu Li, and Liping Zhang. Free pattern search for global optimization. *Applied Soft Computing*, 13 (9):3853–3863, 2013.

Ronald P. White and Hagai Meirovitch. A simulation method for calculating the absolute entropy and free energy of fluids: Application to liquid argon and water. *Proceedings of the National Academy of Sciences of the United States of America*, 101(25):9235–9240, 2004. doi: 10.1073/pnas.0308197101. URL `http://www.pnas.org/content/101/25/9235.abstract`.

S. S. Wilks. Determination of sample sizes for setting tolerance limits. *The Annals of Mathematical Statistics*, 12(1):91–96, 03 1941. doi: 10.1214/aoms/1177731788. URL `http://dx.doi.org/10.1214/aoms/1177731788`.

R Wong. *Asymptotic Approximations of Integrals*, chapter 2, pages 55–146. SIAM, 2001. doi: 10.1137/1.9780898719260.ch2. URL `http://epubs.siam.org/doi/abs/10.1137/1.9780898719260.ch2`.

Tony Siu Tung Wong and Wai Keung Li. A threshold approach for peaks-over-threshold modeling using maximum product of spacings. *Statistica Sinica*, 20: 1257–1272, 2010.

Delisle Worrell. Stressing to breaking point: Interpreting stress test results. Technical report, Caribbean Centre for Money & Finance, 2010.

M. Wright. *Direct search methods: once scorned, new respectable*, pages 191–208. Addison Wesley Longman Limited, Redwood, 1995.