Portfolio Optimisation with Sequential Monte Carlo

Yow Tzu Lim

Submitted in partial fulfilment of the requirements for the MSc in Statistics

Imperial College, London

Department of Mathematics

September 4, 2014

Declaration

The work contained in this thesis is my own work unless otherwise stated.

I am the primary author of all work reported in this thesis. Advice on specific aspects of the work was provided by my supervisor, Dr. Nikolas Kantas.

Abstract

Constructing an optimal portfolio is the ultimate goal of every investment manager; but the optimality criteria may be very different to each of them. For index tracker fund manager, the main objective of portfolio management is to track and replicate the exposure of a benchmark index.

In this study, we view this as a stochastic control problem. We adopt the Bayesian view and treat the investment decision controls as random variables. The objective is to search for the sequence of control parameters that results in a portfolio that tracks a benchmark index in an optimal way. We investigate here the potential of using Sequential Monte Carlo (SMC) as the means of determining such strategy. We examine the feasibility of this technique using two numerical examples. The first example is a toy example with the target reference signal set to be an oscillating wave. In the second example, a real world financial index — the German's DAX index level is used as the target reference signal. In both cases, the technique looks very promising. In the final part of the thesis, we demonstrate how this technique can be integrated into the Model Predictive Control (MPC) framework.

The thesis concludes with an evaluation on the work done, the extent to which the work justify the thesis hypothesis and some possible directions on how SMC can be applied to address a wider range of relevant problems on the domain of concern.

Contents

Acknowledgements

I would like to express my gratitude to all who have provided support in accomplishing this thesis. In particular, I would like to thank my supervisor, Dr. Nikolas Kantas, for the opportunity given to carry out this research, the guidance, expertise, encouragement and patience during the course of this research. I would also like to express my appreciation to my wife Ka Ki Lai for her endless support and care in my personal life during the course of this master degree study. Lastly, I would like to thank God Almighty, who has given me the strength and peace to work hard on this project.

Chapter 1

Introduction

Capital allocation is a common challenge for every investor. In the investment decision making process, investors decide how to allocate capital to different investable assets to form a portfolio that performs better than any other possible portfolios according to some criteria. These criteria can be very different among the investors. Some investors may also have additional considerations such as tax considerations and legal restrictions on investment assets or holding periods.

Two common objectives, which often contradicting, are the financial return and the investment risk. The Markowitz's modern portfolio theory [17] proposes a portfolio selection framework in which it is assumed that investors are interested in maximising their portfolio's return and minimising their portfolio's financial risk (measured by the variance of the portfolio's return). Based on these criteria, the set of non-dominated portfolios, commonly known as the *efficient portfolios*, of a given investment period can be found. However, using the variance of the portfolio's return as the risk measure has its limitation. Variance is a symmetric measure; an out-performing asset is deemed to as risky as an under-performing one. Many alternative risk measurements have been proposed, e.g. Sortino ratio, Conditional Value at Risk (CVaR), etc.; see [21] for details.

There are some investment fund managers who have no strong interest in maximising their portfolio's return. Instead, the main objective of portfolio management for such a fund is to simply track and replicate the exposure of a benchmark index as close as possible. These funds are attractive as they provide the investors the exposure to the market at low fees and taxes because only minimal active management is required. Passively tracking a benchmark index also makes the fund less vulnerable to the change of fund managers. The performance of these funds are often assessed in term of how well the fund tracks the benchmark index using some pre-defined metrics. These index tracking funds are the application of interest in this thesis.

1.1 Methodology

In the era before modern computing, portfolio optimisation have been explored in an analytical fashion, adopting necessary assumption as necessary. This seems rather restrictive; there are many instances where numerical methods have been used to derive an approximate or even more effective solution to the problem in question. For example, Monte Carlo methods are used to approximate integral, heuristic optimisation search techniques such as simulated annealing applied in engineering domains, etc.

In this thesis, we view a portfolio optimisation as a *stochastic* control problem. We adopt the Bayesian view and treat the controls as random variables. The objective is to find the sequence of control parameters that optimises the reward function defined in terms of the tracking error between a fund and its benchmark index. We investigate the potential of using Sequential Monte Carlo (SMC) as the means of determining the optimal strategies for this problem. The main reason of choosing SMC is its ability to carry out *sequential* update on the posterior distribution over time fit well with the problem in question and its success in its applications in many different domains. Of course, other heuristic search techniques are also potentially applicable.

To investigate this approach, we first used SMC to track the output of a simple deterministic reference model. This model is doubly useful. It demonstrates the concept nicely and serves as a basic model to allow us to gain further understanding on the tuneable parameters. We then considered the problem of tracking a real-world index with its constituent prices modelled as Arithmetic Brownian motion with drift. Using SMC, we search for the optimal strategy (the optimal values of the control parameters at each time point) with respect to the reward function. The reward function is defined in terms of the tracking error and the transaction costs involved in the changes of the portfolio's holding position. Lastly, we introduce the Model Predictive Control (MPC) framework and show how to integrate this technique into the MPC framework easily.

1.2 Contributions

In this thesis, it is found that SMC has the potential to be an effective means of searching the optimal strategy for index tracking funds. The following work has been carried out:

1. exploring the potential of SMC in searching the optimal strategy for minimising the tracking error and transaction costs of an index fund.

2. exploring the sensitivity of SMC in terms of the parameter settings, the trade-off between estimation accuracy and computational efforts numerically and providing suggestions for real-world problem.

3. introducing the Model Predictive Control (MPC) framework and demonstrate how to integrate SMC technique proposed into the MPC framework.

Given the time frame of the project, we fully understand it is impossible to evaluate our approach on full scale strategy. The aim is to establish the plausibility or, at the very least, a greater understanding of the strengths and weaknesses of the above approach.

1.3 Thesis organisation

The subsequent chapters of this thesis are organised as follows:

- Chapter 2 reviews some fundamental concepts in Monte Carlo methods that are related to this thesis. It begins with a brief introduction to basic sampling methods such as perfect Monte Carlo sampling, rejection sampling and importance sampling. It then presents Sequential Monte Carlo (SMC) methods include Sequential Importance Sampling (SIS), Sequential Importance Resampling (SIR) and various further enhancements proposed in the literature, e.g., Effective Sample Size (ESS), Resample-Move step, Rao-Blackwellisation, that are being used in this thesis.

- Chapter 3 briefly reviews the portfolio optimisation problem in practice. It then discusses how a portfolio problem can be transformed into a path-space parameter estimation problem. Lastly, it presents a simple experiment in which we attempt to use SMC to track a reference signal generated by a deterministic reference model.

- Chapter 4 details the application of the SMC technique to track the DAX index. It presents several experiments that have been carried out to demonstrate how the index can be tracked with full replication as well as partial replication of its constituents. Lastly, it introduces Model Predictive Control (MPC) framework and demonstrates how this technique can be integrated into the MPC framework easily.

- Chapter 5 concludes the thesis by evaluating the contributions of the thesis and discusses some potential work for the future.

Chapter 2

Monte Carlo methods

Monte Carlo Methods have achieved significant success in its application to various domains in the last few decades. This chapter reviews some fundamental concepts in Monte Carlo methods that are related to this thesis. It begins with a summary on the main concept of Bayesian inference. It then discusses some basic Monte Carlo methods such as perfect Monte Carlo sampling, rejection sampling and importance sampling. Lastly, it introduces Sequential Monte Carlo (SMC), along with various enhancement made to the technique, used in this thesis.

2.1 Bayesian inference

In Bayesian inference, each unknown parameter in the model is assumed to be random variable and is associated with a prior distribution that characterises the initial belief. The inference process is merely about updating the belief with new observable evidence in a systematic fashion using Bayes theorem.

Formally, let $\mathcal{M}$ be the model used, θ be the set of parameters of the model, $p(\theta \mid \mathcal{M})$ be the prior distribution (initial belief) and $p(x \mid \theta, \mathcal{M})$ be the likelihood (probability of observing an observation x given the model) then posterior distribution (updated belief) is given as follows:

$$\begin{aligned} p(\theta \mid x, \mathcal{M}) &= \frac{p(x \mid \theta, \mathcal{M})\ p(\theta \mid \mathcal{M})}{p(x \mid \mathcal{M})} \\ &\propto p(x \mid \theta, \mathcal{M})\ p(\theta \mid \mathcal{M}) \qquad (2.1) \\ \text{posterior} &\propto \text{likelihood} \times \text{prior} \qquad (2.2) \end{aligned}$$

This problem formulation is elegant, but there remains some subtle issues in practice. One particular issue is the calculation of the normalisation constant $p(x \mid \mathcal{M})$ in

(2.1), which demands us to be able to carry out the following integral analytically:

$$p(x \mid \mathcal{M}) = \int p(x \mid \theta, \mathcal{M}) p(\theta \mid \mathcal{M}) \, d\theta \tag{2.3}$$

This is often infeasible. A often way to circumvent this requirement is by making use of conjugate prior that yields posterior distributions from the same family in an analytical fashion. Moreover, the need of calculating integral that does not possess analytic solution also arises in the marginalisation process of nuisance parameters, calculating expectation of a function, etc.

2.2 Perfect Monte Carlo

Instead of a closed form solution, the Method Carlo methods offer a numerical solution in estimating the integral using simple sampling techniques. Consider the calculation the expectation of a function, I of the following form:

$$I = \mathbb{E}_p[f(x)] = \int f(x) p(x) \, dx \tag{2.4}$$

Assuming we are able to sample N independent and identically distributed (i.i.d.) samples of x from $p(\cdot)$, denote these as $\{x^{(i)}\}$ where $i \in \{1 \dots N\}$, a Monte Carlo estimate of I using the the point masses of the samples is:

$$\hat{I} = \frac{1}{N} \sum_{i=1}^{N} f(x^{(i)}) \tag{2.5}$$

This approximation can be viewed as discretisation of the continuous distribution with *random* grid points. This estimate is unbiased and converge almost surely to I as $N \to \infty$ by the Law of Large number [20].

Moreover, if the variance of $f(\cdot)$ is bounded ($\sigma_f^2 < \infty$), as $N \to \infty$, then the following central limit theorem holds:

$$\sqrt{N}(\hat{I} - I) \Longrightarrow N(0, \sigma_f^2) \tag{2.6}$$

where $\Longrightarrow$ denotes convergence in distribution [7]. The key point to note here is this convergence rate of $\frac{1}{\sqrt{N}}$ is independent of the dimensions of x. This is in contrast with any deterministic method that has a rate that decreases as the integral dimension increases [20]. This is the main advantage of Monte Carlo integration.

2.3 Rejection sampling

However, it is not always possible to sample directly from the distribution $p(\cdot)$. Suppose we can find an instrumental distribution (a.k.a. proposal distribution), $q(\cdot)$, that is easy to sample from and has the property such that $cq(x)$ dominates $p(x)$ for all x, i.e., $cq(x) \geq p(x) \geq 0$ for all x, then to get a random sample from $p(\cdot)$, we can first sample from $q(\cdot)$ instead and accept the sample with acceptance probability $\alpha(x) = \frac{p(x)}{cq(x)}$. If the sample is rejected, the process is repeated until success. This rejection algorithm is summarised in Algorithm 1[1].

Algorithm 1 Rejection Sampling

function RejectionSampling(N)
    $\mathcal{X} = \{\ \}$.
    repeat
        sample $x \sim q(\cdot)$.
        sample $u \sim \mathcal{U}(0, 1)$.
        if $u \leq \frac{p(x)}{cq(x)}$ **then**.
            $\mathcal{X} = \mathcal{X} \cup \{x\}$.
        end if
    until len($\mathcal{X}$) $\equiv$ N.
    return $\mathcal{X}$.
end function

Looking at the acceptance ratio formula, it is not difficult to see that the optimal instrumental distribution, q^*, is the one that minimises the space bounded by $cq(x)$ subject to the constraint that it still dominates the target density $p(x)$. As the dimension of x increases, this algorithm becomes very inefficient because the acceptance ratio which is essentially defined as the ratio of two spaces tends towards zero. Therefore, many generated examples would be rejected.

2.4 Importance sampling

Instead of making a binary accept-reject decision on each sample, the key concept in important sampling is assign weighting to each sample (obtained from the instrumental distribution, $q(\cdot)$) based on how well the sample resembles the target distribution, $p(\cdot)$. More formally, assume we have an instrumental distribution, $q(\cdot)$ that has support that

[1]The set notation is used here, despite it is actually a collection that allows duplicates. Therefore, the $\cup$ operation should be viewed as a simple join of two collections.

includes $p(\cdot)$, we can re-write (2.4) as:

$$\begin{aligned} I &= \int f(x)\frac{p(x)}{q(x)}q(x)\ dx \\ &= \int f(x)w(x)q(x)\ dx \\ &= \mathbb{E}_q[f(x)w(x)] \end{aligned} \tag{2.7}$$

where $w(x)$ is commonly referred as the importance weight. This reformulation leads to the following Monte Carlo estimate of I:

$$\begin{aligned} \hat{I} &= \frac{\frac{1}{N}\sum_{i=1}^{N}\tilde{w}(x^{(i)})f(x^{(i)})}{\frac{1}{N}\sum_{j=1}^{N}\tilde{w}(x^{(j)})} \\ &= \sum_{i=1}^{N}\frac{\tilde{w}(x^{(i)})}{\sum_{j=1}^{N}\tilde{w}(x^{(j)})}f(x^{(i)}) \\ &= \sum_{i=1}^{N}\hat{w}(x^{(i)})f(x^{(i)}) \end{aligned} \tag{2.8}$$

where $\tilde{w}(x^{(i)}) = \dfrac{p(x^{(i)})}{q(x^{(i)})}$ and $\hat{w}(x^{(i)}) = \dfrac{\tilde{w}(x^{(i)})}{\sum_{j=1}^{N}\tilde{w}(x^{(j)})}$ are referred to as unnormalised and normalised importance weight respectively [4]. This estimate is biased as it consists of the ratio of two estimates, yet it is still asymptotically consistent.

To obtain samples from the target distribution, $p(\cdot)$, an additional resampling step can be introduced. In the first step, we draw a set of samples $\{\tilde{x}^{(i)}\}$ from the instrumental distribution and compute their associated normalised importance weights, $\hat{w}(\tilde{x}^{(i)})$. In the resampling step, we draw the final sample set, $\{x^{(i)}\}$ from this intermediate set of samples by taking into account the importance weights. This importance sampling algorithm is summarised in Algorithm 2

2.5 Sequential Monte Carlo

To motivate why Sequential Monte Carlo is useful, consider a target distribution of interest $p(x_{0:n})$, and for simplicity, assuming we can sample directly from the distribution $p(x_{0:n})$, the minimal computational complexity of the sampling scheme would be at least linear in n. Sequential Monte Carlo (SMC) provides a way to obtain samples of x for each sequential time step in a *fixed* amount of computational time in Hidden Markov Models (HMMs). We shall begin with a brief introduction on HMMs that is crucial to understand SMC in the next section. Refer [4] for further details of inference techniques for HMMs.

Algorithm 2 Importance Sampling

function ImportanceSampling(N)
  $\tilde{\mathcal{X}} = \{\ \}$
  repeat
    sample $\tilde{x} \sim q(\cdot)$.
    $\tilde{\mathcal{X}} = \tilde{X} \cup \{\tilde{x}\}$.
  until len($\tilde{X}$)=N.
  calculate importance weights, $\hat{w}(\tilde{x}^{(i)}) = \dfrac{\tilde{w}(\tilde{x}^{(i)})}{\sum_{j=1}^{N} \tilde{w}(\tilde{x}^{(j)})}$.
  $\mathcal{X} = \{\ \}$.
  repeat
    sample x from $\tilde{\mathcal{X}}$ according to the importance weights, $\hat{w}$.
    $\mathcal{X} = \mathcal{X} \cup \{x\}$.
  until len($\mathcal{X}$) $\equiv$ N.
  return $\mathcal{X}$.
end function

2.5.1 Hidden Markov Models

HMMs can be seen as a class of models that consist of two related processes: an underlying Markov process, X_t, which is the target process of interest, and a observable process, Y_t, which its state can be measured and therefore provides some information about X_t. Moreover, it is assumed that these two processes have conditional independence properties as shown using the graphical model representation in Figure 2.1. These properties can be summarised as follows:

$$\begin{aligned} p(x_t \mid x_{0:t-1}) &= f(x_t \mid x_{t-1}) \\ p(y_t \mid x_{0:t}, y_{0:t-1}) &= g(y_t \mid x_t) \end{aligned} \tag{2.9}$$

where $f(x_t \mid x_{t-1})$ is the Markov transition density and $g(y_t \mid x_t)$ is the conditionally independent likelihood.

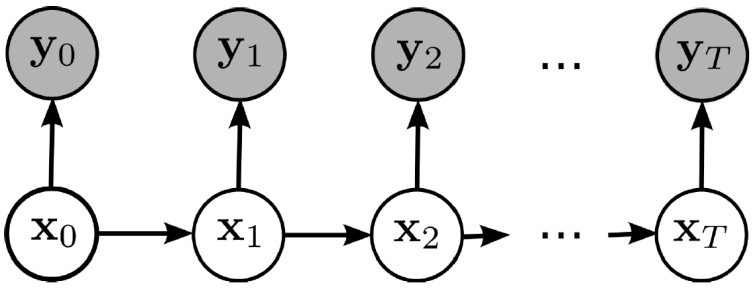

Figure 2.1: Hidden Markov Models

This class of models are designed to model evolving systems that output some observable events over time, which depends on the internal state of the system in a on-line manner. This imposes an implicit requirement on computation perspective: the estimate calculation cost should remain constant over time, i.e., the calculation cost does not increase with the increasing number of states.

Arguably, the most common inference problem in HMMs is the smoothing distribution, $p(x_{0:t} \mid y_{0:t})$, that is estimating the states $x_{0:t}$ based on the sequence of observations up to time t, $y_{0:t}$. Using Bayes rules, we can write the density of the distribution of interest in the recursion form as follows:

$$\begin{aligned} p(x_{0:t} \mid y_{0:t}) &= \frac{p(x_{0:t}, y_{0:t})}{p(y_{0:t})} \\ &= \frac{p(x_{0:t-1}, y_{0:t-1}) f(x_t \mid x_{t-1}) g(y_t \mid x_t)}{p(y_{0:t})} \\ &= p(x_{0:t-1} \mid y_{0:t-1}) \frac{f(x_t \mid x_{t-1}) g(y_t \mid x_t)}{p(y_t \mid y_{0:t-1})} \end{aligned} \tag{2.10}$$

This recursion is often re-written into two separate steps: the prediction step (the estimation of distribution of all states up to time t given only states up to time $t-1$) and the update step (the correction of the predicted distribution taking into account the new observation) as follows:

$$\begin{aligned} p(x_{0:t} \mid y_{0:t-1}) &= p(x_{0:t-1} \mid y_{0:t-1}) f(x_t \mid x_{t-1}) \\ p(x_{0:t} \mid y_{0:t}) &= \frac{p(x_{0:t} \mid y_{0:t-1}) g(y_t \mid x_t)}{p(y_t \mid y_{0:t-1})} \end{aligned} \tag{2.11}$$

Moreover, any smoothing distribution $p(x_{j:k} \mid y_{1:t})$ where $(0 \leq j \leq k \leq t)$ can be obtained by integrating out x_i that are not interested in as follows:

$$p(x_{j:k} \mid y_{0:t}) = \int p(x_{0:t} \mid y_{0:t}) dx_{0:j-1,k+1:t} \tag{2.12}$$

A common smoothing distribution of interest is the marginal distribution at time t, $p(x_t \mid y_{0:t})$, which is also referred to as the filtering distribution.

Another distribution of interest is the prediction distribution, that is the estimation of the distribution of any unseen *future* states using only all observations up to current time. If we let $j = 0$ and $k \geq t$ in (2.12), we obtain the following equation:

$$p(x_{0:k} \mid y_{0:t}) = p(x_{0:t} \mid y_{0:t}) \prod_{i=t+1}^{k} f(x_i \mid x_{i-1}) \tag{2.13}$$

Therefore, any prediction density can be obtained by simply integrating out the variables of not interest from the above equation.

While these expressions *appear* to be very simple, the distribution estimation problem is in fact far from being resolved in practice. The integrals involved in the above equations are often intractable because they require the evaluation of complex high-dimensional integrals[1].

2.5.2 Sequential Important Sampling (SIS)

Sequential Monte Carlo provides a systematic way to approximate the solution to the distribution estimation problem. Let assume that it is possible to decompose the selected proposal distribution into recursion form as follows:

$$\begin{aligned} q_{0:t}(x_{0:t} \mid y_{0:t}) &= q_{0:t-1}(x_{0:t-1} \mid y_{0:t-1}) q_t(x_t \mid x_{0:t-1}, y_{0:t}) \\ &= q_0(x_0 \mid y_0) \prod_{i=1}^{t} q_i(x_i \mid x_{0:i-1}, y_{0:i}) \end{aligned} \tag{2.14}$$

then it is possible to obtain sample $x_{0:t}$ by first sampling $x_0 \sim q_0(\cdot)$ at time 0 and then $x_i \sim q_i(x_i \mid x_{0:i-1}, y_{0:i})$ for all time i from 1 to t. The corresponding weight associated to each sample $x_{0:t}^{(i)}$ can also be decomposed into recursion form as follows:

$$\begin{aligned} \tilde{w_t^{(i)}} &= \frac{p_{0:t}(x_{0:t}^{(i)} \mid y_{0:t})}{q_{0:t}(x_{0:t}^{(i)} \mid y_{0:t})} \\ &\propto \tilde{w}_{t-1}^{(i)} \frac{f_t(x_t^{(i)} \mid x_{t-1}^{(i)}) g_t(y_y \mid x_t)}{q_t(x_t^{(i)} \mid x_{0:t-1}^{(i)}, y_{0:t})} \qquad (2.15) \\ &\propto \tilde{w}_0^{(i)} \prod_{i=1}^{t} \alpha_i(x_i^{(i)}) \qquad (2.16) \end{aligned}$$

where $\alpha_t(x_t)$ is often referred to as incremental importance weight function. This is the key concept in SIS.

Using these weighted samples $\left\{(x_{0:t}^{(i)}, \tilde{w}_t^{(i)})\right\}_{1 \le i \le N}$, it is possible to estimate any function f defined on the space using the self normalised importance sampling estimator in the same way as (2.8) as follows:

$$\begin{aligned} \hat{f}(x_{0:t}) &= \sum_{i=1}^{N} \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^{N} \tilde{w}_t^{(j)}} f(x_{0:t}^{(i)}) \\ &= \sum_{i=1}^{N} \hat{w}^{(i)} f(x_{0:t}^{(i)}) \end{aligned} \tag{2.17}$$

[1] Except under very special setting, in which the marginal filtering density $p(x_t \mid y_{0:t})$, the prediction density $p(x_t \mid y_{0:t-1})$ and the recursive likelihood $P(y_t \mid y_{0:t-1})$ are all Gaussian densities, then their means and variances can be computed analytically using Kalman Filter as shown in Appendix A.

A particular important instance of SMC is obtained when the prior distribution is used as the importance distribution:

$$\begin{aligned} q_{0:t}(x_{0:t} \mid y_{0:t}) &= p(x_{0:t}) \\ &= p_0(x_0) \prod_{i=1}^{t} p_i(x_i \mid x_{0:i-1}) \end{aligned} \tag{2.18}$$

This greatly simplifies importance weight to follows:

$$\tilde{w}_t^{(i)} \propto \tilde{w}_{t-1}^{(i)} g(y_t \mid x_t^{(i)}) \tag{2.19}$$

The SIS algorithm is summarised in Algorithm 3.

Algorithm 3 Sequential Importance Sampling

function SEQUENTIALIMPORTANCESAMPLING(N, T)
  Set $t = 0$.
  For $i \in 1, \ldots, N$, sample $x_0^{(i)} \sim q(x_0^{(i)} \mid y_0^{(i)})$.
  For $i \in 1, \ldots, N$, calculate the unnormalised importance weights:

$$\tilde{w}_0^{(i)} = \frac{f(x_0^{(i)}) g_0(y_0 \mid x_0^{(i)})}{q_0(x_0^{(i)} \mid y_0)}$$

  For $i \in 1, \ldots, N$, normalize the importance weights:

$$\hat{w}_0^{(i)} = \frac{\tilde{w}_0^{(i)}}{\sum_{j=1}^{N} \tilde{w}_0^{(j)}}$$

  Set $t = t + 1$.
  while $t \leq T$ **do**
    For $i \in 1, \ldots, N$, sample $x_t^{(i)} \sim q(x_t^{(i)} \mid y_{t-1}, x_{t-1}^{(i)})$.
    For $i \in 1, \ldots, N$, set $x_{0:t}^{(i)} = (x_{0:t-1}^{(i)}, x_t^{(i)})$.
    For $i \in 1, \ldots, N$, calculate the unnormalised importance weights:

$$\tilde{w}_t^{(i)} = w_{t-1}^{(i)} \frac{f_t(x_t^{(i)} \mid x_{t-1}^{(i)}) g_t(y_t \mid x_t^{(i)})}{q_t(x_t^{(i)} \mid x_{0:t-1}^{(i)}, y_{0:t})}$$

    For $i \in 1, \ldots, N$, normalize the importance weights:

$$\hat{w}_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^{N} \tilde{w}_t^{(j)}}$$

  end while
end function

2.5.3 Optimal proposal distribution

While SIS is attractive, it is nothing but a specialised version of importance sampling introduced earlier in Section 2.4. As the state space increases with the number of time step t, direct importance sampling on a state space that is increasing in size is not very efficient. The weights of the samples will degenerate over time, in the sense that the weights start to concentrate only on a small number of samples. Consequently, many samples will have negligible weights and do not contribute much in the estimating the expectation. See [10] for a step-by-step illustration. The weight degeneracy issue cause the quality of the estimate degrades over time.

To alleviate this issue, looking at (2.15), it is obvious that the variance of importance weights can be minimised by using a proposal distribution of the following form:

$$p_t(x_t \mid x_{t-1}, y_t) \propto f_t(x_t \mid x_{t-1}) g_t(y_t \mid x_t) \tag{2.20}$$

This is often referred to as the optimal proposal distribution.

In general, it is not always possible to sample from this optimal proposal distribution. Yet, the knowledge of its form can be helpful in designing a reasonable good proposal distribution, which one can sample from. Better proposal *reduces* the amount of variance introduced, but it *does not eliminate* the weight degeneracy problem.

2.5.4 Sequential Importance Resampling (SIR)

The variance in importance weights accumulates over iterations. This suggests a possible solution is to "reset" the weights associated to the samples somehow during the iterations [10]. Sequential Importance Resampling (SIR) introduces an additional resampling step to SIS step in a similar fashion as discussed in Section 2.4. After resampling, the weight of each sample is reset to be equal, i.e., $\frac{1}{N}$. This resampling step however introduces extra variance to the estimators.

A simple direct implementation of resampling step is to select the sample from the intermediate stage according to a Multinomial distribution with the success probability parameter set to the vector of normalised weights, $\hat{w}(x^{(i)})$, i.e., the chance of a sample point being replicated is proportional to its weight.

There are many other resampling schemes have been proposed in the literature. For example, stratified resampling [14] as the name suggested splitting the samples into strata to ensure the good coverage on the resulting sample set, residual resampling [15] which is able to decrease the variance of the weights due to resampling, etc. See [6] for further details on the comparison of these sampling schemes. The SIR algorithm is now summarised in Algorithm 4.

2.5.5 Effective Sample Size (ESS)

Resampling step induces additional Monte Carlo variance to the weights. Yet, this step is necessary to avoid accumulation of estimation variance onto the weights over time and therefore results in a more accurate estimate.

To trade-off these two competing requirements, one possible way is to monitor the Effective Sample Size (ESS) of the sample set which provides a measure on the quality of the weighted samples. The ESS value can be estimated as follows:

$$\text{ESS} \approx \frac{1}{E[w^2]} \approx \frac{\left(\sum_{i=0}^{N} w_i\right)^2}{\sum_{i=0}^{N} w_i^2} \tag{2.21}$$

A common way to integrate this idea is to trigger the resampling step only if the ESS value at time step t fall below certain threshold at time t, say $\frac{N}{2}$. See [10] for further details on ESS.

Algorithm 4 Sequential Importance Resampling

1: **function** SequentialImportanceResampling(N, T)
2: Use Steps 2-11 of Algorithm 3
3: **Resample:** For $i \in 1, \ldots, N$, resample $x_{0:t}^{(i)} \sim \frac{\sum_{i=1}^{N} \hat{w}_t^{(i)} \delta_{x_{0:t}^{(i)}}}{\sum_{j=1}^{N} \hat{w}_t^{(j)}}$
4: **end function**

2.5.6 Resample-Move Algorithm

However, resampling is not a silver bullet for sampling impoverishment. Essentially, resampling provides a mechanism to eliminate low weight samples to give way to replicate *copies* of high weight samples. This allows all samples to participate and contribute to the distribution estimation. This is obvious beneficial for the case of estimating filtering distribution and predictive distribution. Over time, this replication results in decrease in the number of distinct samples for previous time steps. Eventually, many samples will have the share the same sample trajectory. This is a fundamental weakness of SMC, in which the sample path history is never re-written. This lose of diversity in the sample set will have a negative impact when it comes to any smoothing distribution estimation.

To counteract this sample impoverishment, Resample-Move Algorithm [3, 8] is proposed to introduce some perturbation to the samples (so to diversify them) without changing the distribution they represent. In the original paper, this is accomplished by adding a MCMC "move" step to each sample after the resampling step with a Markov

Kernel, K that is invariant to the target distribution, $p(\cdot)$. This Resample-Move algorithm is summarised in Algorithm 5.

This does not entirely solve the smoothing distribution estimation problem. To apply Markov Kernels with invariant distribution corresponding to the smoothing distribution, the space that Markov kernel is defined has to increase at each iteration. This implies the computation time increases linearly with time. Moreover, fast mixing high dimension Markov kernel in itself is not easy to design [10].

To trade-off between the two requirements, one could adopt a sliding windows approach, in which the MCMC Kernel is used to diversify only the samples of the previous n time step at each iteration. Adding this sliding window approach into the standard SIR algorithm will make each iteration has just an additional *fixed* computational cost.

Algorithm 5 Resample-Move Algorithm

1: **function** ResampleMoveAlgorithm(N, T)
2: Use Steps 2-11 of Algorithm 3
3: **Resample:** For $i \in 1, \ldots, N$, resample $x_{0:t}^{(i)} \sim \frac{\sum_{i=1}^{N} \hat{w}_t^{(i)} \delta_{x_{0:t}^{(i)}}}{\sum_{j=1}^{N} \hat{w}_t^{(j)}}$
4: **Move:** For $i \in 1, \ldots, N$, sample $x_{0:t}^{(i)} \sim K_t(\cdot)$, where K_t is p_t-invariant.
5: **end function**

2.5.7 Rao-Blackwellised SMC

In practice, many models may not be entirely non-linear or non-Gaussian. Some states may be linear and Gaussian, conditional upon other states. Instead of naively modelling all state spaces using SMC, a better approach would be carrying out analytical computation as much as possible on the linear Gaussian states and use SMC to model only the non-linear states. This marginalisation setting will yield estimate that has smaller variance.

To illustrate this, let consider the a simple conditional linear Gaussian Model as follows:

$$X_t^L = A_t(X_t^N)X_{t-1}^L + B_t(X_t^N)W_t + F_t(X_t^N) \tag{2.22}$$

$$Y_t = C_t(X_t^N)X_t^L + D_t(X_t^N)V_t + G_t(X_t^N) \tag{2.23}$$

where $\{X_t^N\}_{t\geq 0}$ is a non-linear Markov process, A_t, B_t, C_t, D_t, F_t, G_t are appropriate matrix/vector of X_t^N and $\{W_t\}_{t\geq 0}$ and $\{V_t\}_{t\geq 0}$ are independent sequences of standard Gaussian random variables, i.e., $W_t, V_t \sim \mathcal{N}(0, I)$. In such a case, the transition density and likelihood of this model are Gaussian distributions with centre lied at a point of a

linear combination of the known x_t^N that can be obtained by using the classical Kalman Filtering recursions as shown in Appendix A.

The discussion here has been focused on the conditional linear Gaussian model. The discrete state space HMMs is another important class of HMMs in which HMM forward algorithm [18] can be used to marginalise the internal states. See [4] for further details.

2.6 Conclusion

This chapter presents a review of Monte Carlo methods, with a particular focus on Sequential Monte Carlo (SMC), along with various extensions proposed in the literature, used in this thesis for portfolio optimisation.

It is worth to have a remark here that the SMC is not only restricted to solving sequential filtering/smoothing problem. For example, it has been established that it is possible to use SMC within MCMC framework (pMCMC, where p stands for particle) [2] to build high dimensional proposal distribution to improve standard MCMC. In the next chapter, we will show how SMC can be used as a maximiser to search for a optimal strategy for the optimal portfolio, given a multiplicative reward function.

Chapter 3

Portfolio optimisation

Optimal portfolio is the ultimate goal of every investment manager; but the optimality criteria can be very different to each of them. In the infamous Markowitz's modern portfolio theory [17], it is assumed that investors are always interested in maximising the portfolio's return and minimising the portfolio's financial risk. For index tracker fund manager, the main objective of portfolio management is to track and replicate the exposure of a benchmark index. The lack of active management generally makes the fund less vulnerable to change of management and has the advantages of lower fees and taxes. It is the latter that the focus of the thesis lies.

In this chapter, we introduce the index tracking fund and summarise some causing factors of the tracking error between a fund and its benchmark index. We then present how the portfolio optimisation problem in terms of minimising the tracking error can be formulated as a stochastic control problem. We show this stochastic control problem can be turned into a path-space parameter estimation problem, in which Rao-Blackwellised SMC can be used to estimate these parameters efficiently. Lastly, we present a simple numerical example to demonstrate how to apply this technique to track a deterministic reference signal.

3.1 Index tracking fund

An financial index can be thought of a summary statistics of a market, typically formed as weighted average of the prices of some financial instruments. For example, FTSE 100 is an index that attempts to represent the performance of the 100 biggest companies in UK. This index is however not an investable financial product one can invest directly.

To allows investors to take exposure to the markets (as represented by the indices), index tracker funds are introduced. These funds generally follow very strict and transparent rules with the main objective to track their benchmark indices as close

as possible. Investors of these funds therefore are only exposed to the market risks of the chosen indices, with minimal exposure to the risk associated with the investment process of the traditional active fund management.

3.1.1 Index replication

To track an index, the simplest way is to replicate the index by investing in all the components of the benchmark index. However, this can be costly and difficult to manage, considering some of the indices may have huge number of components, e.g., MSCI World Index that consists of ≈ 1600 components from many different countries. Instead, one could partial replicate the index by sampling some of the components that are most representative. This can mean those components with larger weights, more volatiles or less correlated in returns (assuming these returns average out the gains/losses among them). This partial replication could help in reducing the transaction costs but introduce additional tracking errors to the fund.

Alternatively, the portfolio managers also have the option to enter a swap agreement with a counter-party (typically an investment bank) that provides the exact return of index in exchange for a fixed cost. Essentially, this transfers the market risk to the counter-party, at the same time, introduces the counter-party default risk. This replication technique is known synthetic replication.

3.1.2 Causing factors of tracking error

To evaluate the performance of an index tracking fund, different metrics have been introduced to quantify the mismatch between the performance of the fund and its benchmark index. For example, the tracking difference is the sum of absolute difference in returns between of a fund and its benchmark index. Here, we adopt the tracking error as our metric, which is defined to be the standard deviation of the absolute difference of the returns of a fund and its benchmark index defined in [11] as follows:

$$\epsilon = \sqrt{\mathrm{Var}[r_p - r_b]} \tag{3.1}$$

where r_p is the return of the fund and r_b is the return of the benchmark index.

There are many factors that can cause the tracking errors, some of which are summarised as follows:

1. Index rebalancing — the benchmark index is re-weighting its constituents periodically to reflex the changes on the market based on its methodology. To track the index, the fund has to adjust its portfolio accordingly. This will incur some

transaction costs. During the index rebalance period, cash drag may also happen between the liquidation of the constituents that have weights reduced/dropped and the addition of the constituents that have weights increased/added. This cash is essentially not participating in the market and therefore it does not reflex the changes on the benchmark index.

2. Replication and sampling techniques — funds may choose to replicate the benchmark index by selecting a subset of the constituents (often the ones with larger weights and more liquid) in an effort to minimise the transaction costs. This exclusion of the smaller, less liquid constituents may introduce another source of tracking error, especially under stressed market condition.

3. Different assumption on dividend reinvestment and taxation — the benchmark index calculation often assumes immediate reinvestment of dividends on ex-dividend dates but the fund may only able to reinvest the dividend after receiving it. The tax treatment that is applied on the dividends may also be different.

4. Total Expense Ratio (TER) — there is an additional expense charged to the fund on daily basis to cover the management cost.

This list is by no mean exclusive. See [11] for further details.

3.2 Methodology

Instead of using a traditional deterministic state space models, we choose to use a *stochastic* modelling approach to carry out the portfolio optimisation. In particular, we focus here the the problem in minimising the tracking error between a portfolio and its benchmark index. Our aim is to determine what investment actions (buy or sell) a portfolio manager needs to do for each index component on a daily basis across the investment horizon to track the benchmark index well.

We adopt here a conditional Gaussian linear state space model as follows:

$$\begin{aligned} X_t &= A_t(U_t)X_{t-1} + B_t(U_t)W_t + F_t(U_t) \\ Y_t &= C_t(U_t)X_t + D_t(U_t)V_t + G_t(U_t) \end{aligned} \tag{3.2}$$

where $\{U_t\}_{t\geq 0}$ is a deterministic control input sequence that is used regulate the hidden states, A_t, B_t, C_t, D_t, F_t, G_t are appropriate matrix/vector functions of U_n and $\{W_t\}_{t\geq 0}$ and $\{V_t\}_{t\geq 0}$ are independent sequences of standard Gaussian random variables, i.e., $W_t, V_t \sim \mathcal{N}(0, I)$. The transition density and likelihood of this model are Gaussian

distributions with centre lied at a point of a linear combination of the known conditional control parameters, u_t of the following form:

$$\begin{aligned} p_t(u_t \mid u_{t-1}) &= (\text{any given form}) \\ f_t(x_t \mid x_{t-1}, u_t) &= \mathcal{N}(A_t(u_t)x_{t-1} + F_t(u_t), B_t(u_t)B_t(u_t)^T) \\ g_t(y_t \mid x_t, u_t) &= \mathcal{N}(C_t(u_t)x_t + G_t(u_t), D_t(u_t)D_t(u_t)^T) \end{aligned} \tag{3.3}$$

Based on this model, the optimisation objective is to search for a sequence of controls $u_{1:t}$ that would result in a sequence of observations $y_{1:t}$ that tracks the reference signal $y_{1:T}^{ref}$ as close as possible. This problem is often known as stochastic regulation problem. We adopt the finite horizon multiplicative reward function proposed in [13] here:

$$J(u_{1:T}, y_{1:T}^{ref}, x_0) = \mathbb{E}_{x_0}\left[\exp\left(-\frac{1}{2}\sum_{t=1}^{T}\left(||y_t^{ref} - C_t(u_t)x_t - G_t(u_t)||^2_{Q_t(u_t)^{-1}} + ||u_t - u_{t-1}||^2_{L_t}\right)\right)\right] \tag{3.4}$$

where $Q(u_t) = D_t(u_t)D_t(u_t)^T$ and L_t are assumed to known. The expectation here is taken with respect to the whole path of the Markov Chain $\{X_t\}_{t\geq 0}$ with the starting point $X_0 = x_0$, i.e., $E_{x_0}[\phi(X_{1:t})] = \int \phi(X_{1:T}) \prod f_t(x_t \mid x_{t-1})\, dx_{1:t}$.

The corresponding optimal open loop policy is:

$$u_{1:T}^* = \arg\max_{u_{1:T}} J(u_{1:T}; y_{1:T}^{ref}; x_0) \tag{3.5}$$

with the assumption that the maximum is attainable. This reward function is closely related to the risk sensitive control discussed in [24], with a slight difference. This reward function used here is absence of an explicitly risk sensitive constant and it also assumes the presence of $y_{1:T}^{ref}$ as the reference target signal instead of the observable states. Nevertheless, a risk sensitivity constant could still be introduced through D_n and L_n if necessary.

3.2.1 Problem formulation

We assume here that the sequence of controls $\{U_t\}_{t\geq 0}$ is a Markov process with transition distribution $p(u_t \mid u_{t-1})$. The objective is to compute the marginal posterior distribution density:

$$\pi_t(u_{1:t}) = p(u_{1:t} \mid y_{1:t}^{ref}) \tag{3.6}$$

by standard marginalisation the distribution $p(x_{1:t}, u_{1:t} \mid y_{1:t})$ which in itself satisfies the following recursive equation:

$$p(x_{1:t}, u_{1:t} \mid y_{1:t}) = p(x_{1:t-1}, u_{1:t-1} \mid y_{1:t-1}) \frac{p(y_t \mid x_t, u_t)p(x_t, u_t \mid x_{t-1}, u_{t-1})}{p(y_t \mid y_{1:t-1})} \tag{3.7}$$

One straight-forward way to approximate the recursion is directly apply SMC by consider the hidden states is a sequence of paired states $\{(U_t, X_t)\}_{t\geq 0}$. A better approach would be using the Rao-Blackwellised SMC mentioned in Section 2.5.7, attempts to do as much computation analytically as possible, by splitting the states into the linear Gaussian states and the non-linear states. Then, the Kalman Filter can be used to model and track the linear Gaussian states and the SMC is only used to model and track the non-linear states. This marginalisation setting often yields estimates with smaller variance.

To use Rao-Blackwellised SMC on this model, we consider the following factorisation on the full posterior distribution as follows:

$$p(u_{1:t}, x_{1:t} \mid y_{1:t}) = p(x_{1:t} \mid u_{1:t}, y_{1:t}) p(u_{1:t} \mid y_{1:t}) \tag{3.8}$$

Looking at the right hand side of the equation, the first term is obviously Gaussian and can therefore can be estimated optimally using Kalman Filter recursions described in Appendix A. For the second term, we can rewrite it into the following recursion form:

$$\begin{aligned} p(u_{1:t} \mid y_{1:t}) &\propto p(y_t \mid u_{1:t}, y_{1:t-1}) p(u_{1:t} \mid y_{1:t-1}) \\ &= p(y_t \mid u_{1:t}, y_{1:t-1}) p(u_t \mid u_{1:t-1}, y_{1:t-1}) p(u_{1:t-1} \mid y_{1:t-1}) \\ &= p(u_{1:t-1} \mid y_{1:t-1}) p(y_t \mid u_{1:t}, y_{1:t-1}) p(u_t \mid u_{1:t-1}, y_{1:t-1}) \end{aligned} \tag{3.9}$$

Assuming that it is also possible to decompose the selected importance density into recursion form as follows:

$$q_{1:t}(u_{1:t} \mid y_{1:t}) = q_{1:t-1}(u_{1:t-1} \mid y_{1:t-1}) q_t(u_t \mid u_{1:t-1}, y_{1:t}) \tag{3.10}$$

then the associated unnormalised weight of each sample can be derived in a similar fashion as given in Section 2.5.2 into the recursion form as follows:

$$\tilde{w}_t = \tilde{w}_{t-1} \frac{p_t(u_t \mid u_{t-1}) p_t(y_t \mid u_{1:t}, y_{1:t-1})}{q_t(u_t \mid u_{1:t-1}, y_{1:t})} \tag{3.11}$$

Compare to (2.15), the main difference here is that the weights now depends on the whole path space from time step 1 to t.

3.2.2 Extensions

As discussed in Section 2.5.5, the resampling step introduces variance to the the weights, but it is necessary to avoid accumulation of estimation variance onto the weights over time. Effective sample size (ESS) can be used to measure the quality of the weight and

trigger the resampling step only if necessary, e.g., when the ESS value at time step t fall below certain threshold at time t, say $\frac{N}{2}$.

To add diversity in the population of particles, the Resample-Move step introduced in Section 2.5.6 can also be included. This step perturbs the samples yet leave the distribution the samples represent unchanged using a MCMC kernel that is invariant in density. With this optional step in place, the whole algorithm is summarised in Algorithm 6.

3.2.3 MAP estimation for the control sequence

Based on the model defined in (3.2), the conditional likelihood density of the model is as follows:

$$g_t(y_t^{ref} \mid x_t, u_t) \propto \exp\left(-\frac{1}{2}||y_t^{ref} - C_t(u_t)x_t - G_t(u_t)||^2_{Q_t(u_t)^{-1}}\right) \tag{3.13}$$

and the posterior distribution $p(y_t^{ref} \mid y_{t-1}^{ref}, u_{1:t})$ is as follows:

$$p(y_t^{ref} \mid y_{t-1}^{ref}, u_{1:t}) \propto \exp\left(-\frac{1}{2}||y_t^{ref} - m_{t|t-1}(u_{1:t})||^2_{S_t(u_{1:t})^{-1}}\right) \tag{3.14}$$

where $m_{t|t-1}$ and S_t are the mean and covariance matrix of the measurement at time t obtained in the Kalman Filter recursions discussed in Appendix A.

By assuming the Markov transition density for u_t to be:

$$p(u_t \mid u_{t-1}) \propto \exp\left(-\frac{1}{2}||u_t - u_{t-1}||^2_{L_t}\right) \tag{3.15}$$

the Maximum a Posteriori (MAP) estimate of $U_{1:T}$ is given to be:

$$\tilde{u}^*_{1:t} = \arg\max_{u_{1:t}} \pi_t(u_{1:t}) \tag{3.16}$$

which essentially specifies the model that best explains the reference $y_{1:t}^{ref}$ as the output sequence from a general set of models defined in (3.2).

In [13], it had been shown that assuming a Markov transition density for U_t in (3.15), the optimal control defined in (3.5) and the MAP estimate defined in (3.16) are the same. Moreover, following the monotonicity of the transformation $p(\cdot)^\gamma$, the mode of $p(\cdot)^\gamma$ is the same for all $\gamma > 0$. The main difference is that when it comes to sampling, it is easier to sample closer to the mode of the distribution when $\gamma > 1$ because the density has sharper peak and vice versa.

Putting all these together, we have the following:

$$u^*_{1:t} = \tilde{u}^*_{1:t} = \arg\max_{u_{1:t}} \pi_t(u_{1:t})^\gamma, \ \gamma > 0 \tag{3.17}$$

Algorithm 6 Rao-Blackwellised SMC to search for the optimal control parameters

function OPTIMALPARAMETERS(N, T)

Set $t = 1$.

For $i \in 1, \ldots, N$, sample $u_1^{(i)} \sim q(u_1^{(i)} \mid y_1^{(i)})$.

For $i \in 1, \ldots, N$, calculate the unnormalised importance weight:

$$\tilde{w}_1^{(i)} = \frac{p(u_1^{(i)}) g_1(y_1 \mid u_1^{(i)})}{q_1(u_1^{(i)}, y_1)}$$

For $i \in 1, \ldots, N$, normalize the importance weight:

$$\hat{w}_1^{(i)} = \frac{\tilde{w}_1^{(i)}}{\sum_{j=1}^{N} \tilde{w}_1^{(j)}}$$

Set $t = t + 1$.

while $t \leq T$ **do**

For $i \in 1, \ldots, N$, sample $u_t^{(i)} \sim q(u_t^{(i)} \mid y_{t-1}^{(i)}, u_{t-1}^{(i)})$.

For $i \in 1, \ldots, N$, calculate the unnormalised importance weight:

$$\tilde{w}_t^{(i)} = w_{t-1}^{(i)} \frac{p_t(u_t^{(i)} \mid u_{t-1}^{(i)}) g_t(y_t \mid u_{1:t}^{(i)}, y_{1:t-1})}{q_t(u_t^{(i)} \mid u_{1:t-1}^{(i)}, y_{1:t})}$$

where

$$p(y_t \mid u_{1:t}^{(i)}, y_{1:t-1}) \sim N(m_{t|t-1}, S_t) \tag{3.12}$$

with $m_{t|t-1}$ and S_t are updated with Kalman Filter recursions:

$$\begin{aligned}
\mu_{t|t-1} &= A_t(u_t)(\mu_{t-1|t-1}) X_{t-1} + F_t(u_t) \\
\Sigma_{t|t-1} &= A_t(u_t) \Sigma_{t-1|t-1} A_t(u_t)^T + B_t(u_t) B_t(u_t)^T \\
S_t &= C_t(u_t) \Sigma_{t|t-1} C_t(u_t)^T + D_t(u_t) D_t(u_t)^T \\
m_{t|t-1} &= C_t(u_t) \mu_{t|t-1} + G_t(u_t) \\
\mu_{t|t} &= \mu_{t|t-1} + \Sigma_{t|t-1} C_t(u_t) S_t^{-1} (y_t - m_{t|t-1}) \\
\Sigma_{t|t} &= \Sigma_{t|t-1} - \Sigma_{t|t-1} C_t(u_t) S_t^{-1} C_t(u_t) \Sigma_{t|t-1}
\end{aligned}$$

For $i \in 1, \ldots, N$, normalize the importance weight:

$$\hat{w}_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^{N} \tilde{w}_t^{(j)}}$$

Resample: For $i \in 1, \ldots, N$, resample $u_{1:t}^{(i)} \sim \dfrac{\sum_{i=1}^{N} \hat{w}_t^{(i)} \delta_{u_{1:t}^{(i)}}}{\sum_{j=1}^{N} \hat{w}_t^{(j)}}$.

Move: For $i \in 1, \ldots, N$, sample $u_{1:t}^{(i)} \sim K_t(\cdot)$, where K_t is p_t-invariant.

end while

Compute the MAP estimate: $\hat{u}_{1:t}^{*} = \arg\max_{u_{1:t}} p(u_{1:t}^{(i)} \mid y_{1:t}^{ref})^{\gamma}$.

end function

In SMC algorithm, this can be easily estimated as follows:

$$\hat{u}^*_{1:t} = arg\max_{u_{1:t}} \pi_t(u^{(i)}_{1:t})^\gamma = \arg\max_{u_{1:t}} p(u^{(i)}_{1:t} \mid y^{ref}_{1:t})^\gamma \tag{3.18}$$

As long as the support of the proposal density include the support of $(u^{(i)}_{1:t} \mid y^{ref}_{1:t})$, this estimate converge asymptotically to the $\tilde{u}^*_{1:t}$ as $N \to \infty$. A better estimate can be obtained by embedding the Viterbi strategy as discussed in [9], but this comes with a computational cost that is quadratic in terms of the number of samples.

3.3 Numerical example: Tracking an oscillating wave

We first consider here a simple linear Gaussian state space model as follows:

$$\begin{aligned} X_t &= X_{t-1} + W_t + U_t \\ Y_t &= X_t + V_t \end{aligned} \tag{3.19}$$

with $W_t, V_t \sim \mathcal{N}(0, I)$. This model is essential an instance of the class of model presented earlier in (3.2), with $A_t = B_t = C_t = D_t = I$, $F_t(U_t) = U_t$ and $G_t(U_t) = 0$. The target reference is set to be an oscillating wave: $y^{ref}_t = \cos(0.2\pi t + 0.3)$ and L_1 is set to 0.1 similar to the setting used in [13].

This model serves two purposes here. Firstly, it is sufficient simple to verify the implementation[1]. Secondly, it serves as good benchmark problem in which we can examine the sensitivity of the algorithm to different parameter settings.

We proceed by examining the proposed SMC technique with proposal density $q_t(\cdot)$ set to be the same as the Markov transition density and all the possible combination of settings as follows:

1. Various time period length settings, T: 5, 10 and 20.
2. Various sample size settings, N: 100, 500, 1000, 5000 and 10000.
3. Resampling step: Always vs. Selectively (trigger condition set to be ESS $< \frac{N}{2}$).
4. Resample-Move step with MCMC (random walk proposal): Always vs. disabled.
5. Various γ settings: Constant function of 1, 50, 100, 1000 and increasing function of time t, $10t$, $50t$ and $100t$.

For each of the setting, the experiment is repeated for 30 times. Therefore, there are in total $3 \times 5 \times 2 \times 2 \times 8 \times 30 = 14400$ runs.

[1] Strictly speaking, testing only increases confidence of the implementation correctness but does not prove no bug.

3.3.1 Results and discussion

This sections presents a summary of the experimental results from all the runs. Instead of presenting the outputs produced by all the runs, only a subset of outputs that demonstrates our findings is presented here. The complete set of implementation code and output produced can be obtained from the repository of this project at GitHub.

3.3.1.1 Selectively resampling step with ESS

We first evaluate the effect of selectively enabling resample step with ESS mechanism. In *all* the settings, it is found that the simple runs with resample step enabled at each iteration, i.e., with ESS disabled, always have equivalent or relative better performance in comparison to selectively resampling with ESS. Figure 3.1 shows the result the performance obtained in terms of $\log \pi_T(\hat{u}^*_{1:t})$ of 30 independent runs of the algorithms with $T = 20$ and $N = 10000$ for various γ settings. All other runs with different settings of T and N concur with this finding. From here onwards, we assumes ESS mechanism is disabled unless stated otherwise.

To explain this counter-intuitive result, recall that ESS is essentially a proxy to measure the quality of the weighted samples in representing the target distribution. A sample set that is well spread over the target distribution would have high ESS value and vice versa. However, the application of SMC as a maximiser here is *only* concerned with the mode of the distribution. At each time step, the process of resampling the samples proportional to the weights essentially pushes the sample set towards the mode of the distribution. Selectively resampling using ESS essentially reduces the strength of this push and therefore leads to sub-optimal solution.

3.3.1.2 Resample-Move step with MCMC

In Figure 3.1, we can also observe that the Resample-Move step does improve performance, albeit the improvement is marginal in most of the cases. It is also found that the improvement is more significant when γ is set to be high. Figure 3.2, Figure 3.3 and Figure3.4 show the box plots of $\log \pi_T(\hat{u}^*_{1:t})$ of 30 independent runs for time period T of 5, 10, 20 respectively. Each sub-figure shows the runs for a selected γ setting with the box-plots grouped by whether the Resample-Move step is used (the leftmost 5 box-plots) or not (the rightmost 5 box-plots), sorted in terms of number of samples used N.

One possible explanation to this result is that high γ settings sharpen the distribution shape. Such a setting encourages optimisation by pushing the sample towards

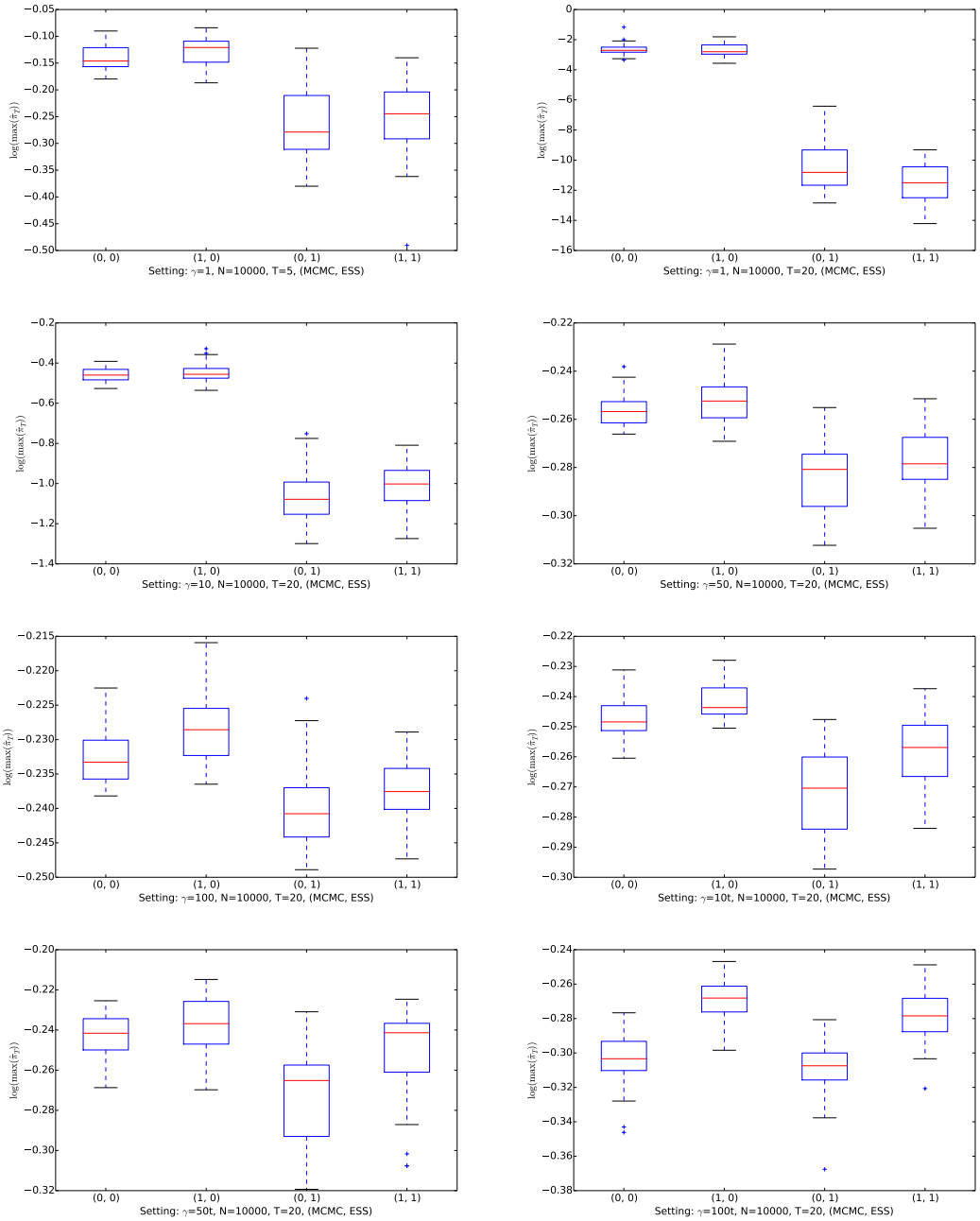

Figure 3.1: The performance in terms of $\log \pi_T(\hat{u}^*_{1:t})$ of 30 independent runs of the algorithms with $T = 20$ and $N = 10000$ for various γ settings. The four box-plots in each sub-figure represent the results of runs with the following settings (from left to right): a) MCMC disabled, ESS disabled, b) MCMC enabled, ESS disabled, c) MCMC disabled, ESS enabled, and d) MCMC enabled, ESS enabled.

high density area but at the same time reduces the state space exploration. With excessively high γ settings, the sample set may be led to a local optimal with relatively high density area, but may not be the mode of the distribution. The perturbation that Resample-Move step introduced at each iteration may help particles to escape from local optimal.

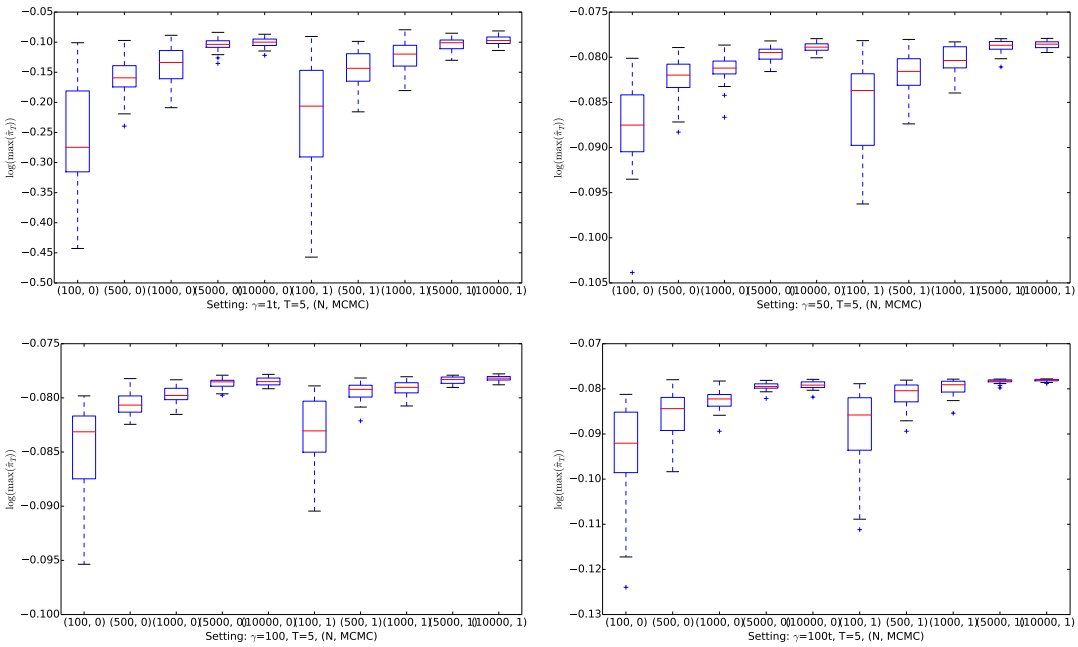

Figure 3.2: The $\log \pi_T(\hat{u}^*_{1:t})$ box-plots of 30 independent runs with various settings for $T = 5$. Each sub-figure shows the runs for a selected γ setting with the box-plots grouped by whether the Resample-Move step is used (the leftmost 5 box-plots) or not (the rightmost 5 box-plots), sorted in terms of number of samples used N.

3.3.1.3 Various sample size settings

In terms of the sample size setting used, the performance increased monotonically with the number of samples used for all time periods. This can seen clearly from the box-plots shown in Figure 3.2, Figure 3.3 and Figure 3.4. This is in-line with the theory that more samples is always better. For the shorter time-period considered, says $T = 5$, the performance improvement sometimes is "saturated" even with lower number of samples, says $T = 1000$ or $T = 5000$. This suggests the optimal control parameters may have been attained.

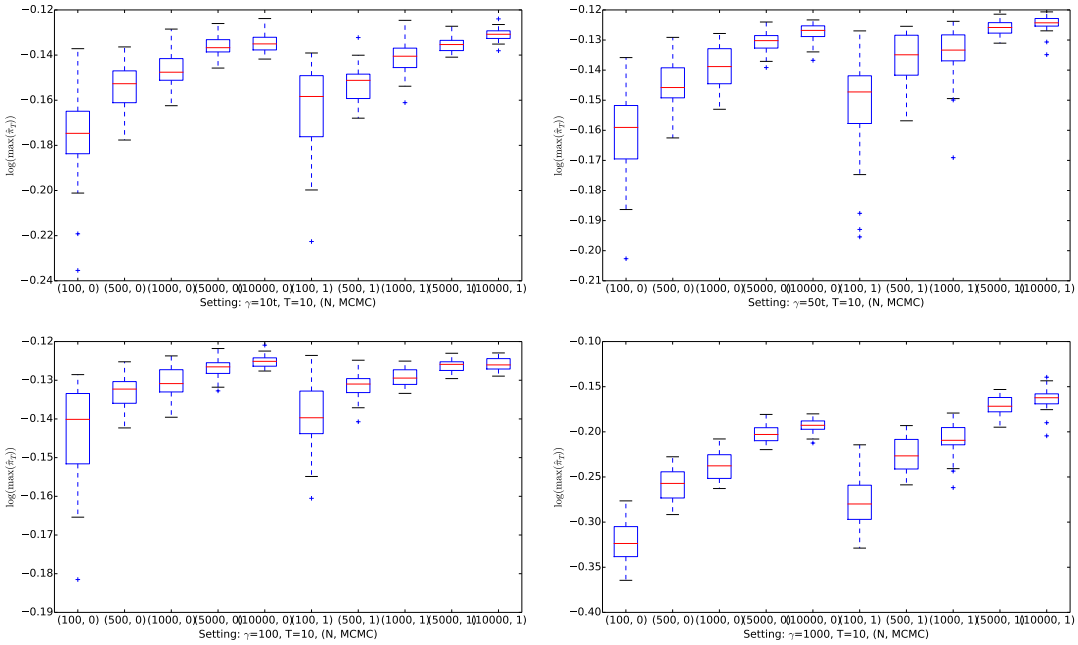

Figure 3.3: The $\log \pi_T(\hat{u}^*_{1:t})$ box-plots of 30 independent runs with various settings for $T = 10$. Each sub-figure shows the runs for a selected γ setting with the box-plots grouped by whether the Resample-Move step is used (the leftmost 5 box-plots) or not (the rightmost 5 box-plots), sorted in terms of number of samples used N.

3.3.1.4 Various γ settings

A more interesting result is to compare the performance with different γ settings. Let first consider the fixed γ settings. Excessive high γ settings may lead to the issue discussed earlier and stuck at local optimal. On the other hand, low γ settings also does not perform too well either (consider the process of searching for the mode by randomly sampling a very flat distribution). The optimal γ setting seems to lie sometimes in between. In this case, the $\gamma = 100$ setting results in the best performance.

In [13], it was suggested that using an increasing γ setting may be a good pragmatic compromise between accuracy and good mixing. We investigate further on this by setting γ to be increasing function of the form αt, where α is constant and t is the time step. The results show that the performance seems to be reasonably good when the α value is set low, but worse when the α value is set too high. This suggests a capping on the maximum value on α or a more moderate increase in t may be more suitable. We therefore carry out two extra γ settings: $50 \log(t + 1)$ and $100 \log(t + 1)$ to investigate this. The results of these runs are shown in Figure 3.5. The results from these runs look good but they are not statistically different from a simple fixed $\gamma = 100$ setting.

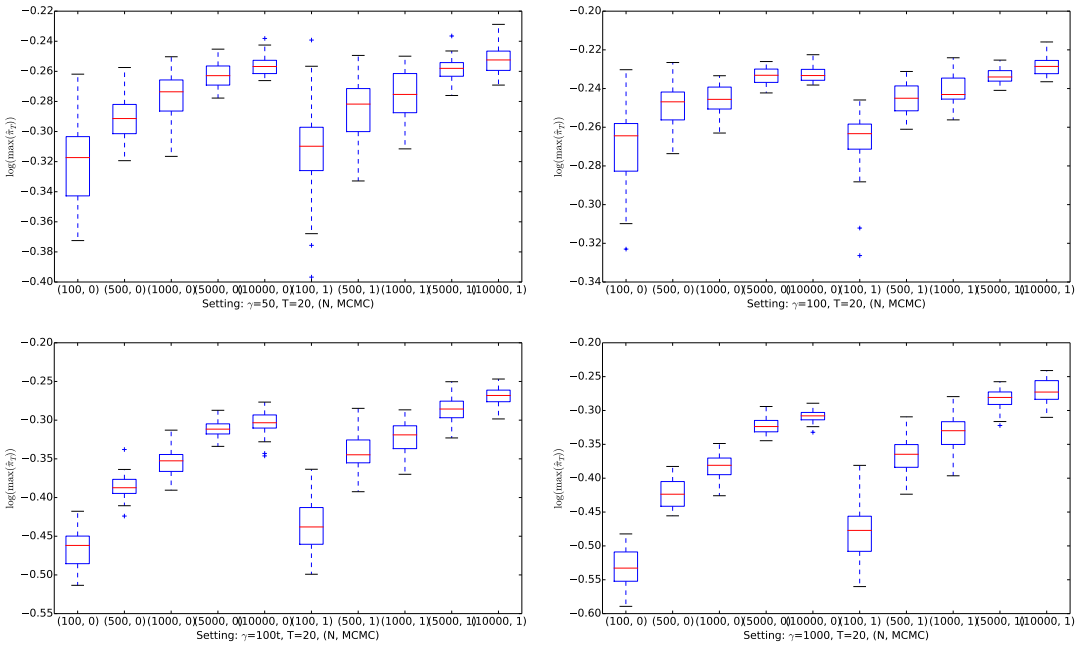

Figure 3.4: The $\log \pi_T(\hat{u}^*_{1:t})$ box-plots of 30 independent runs with various settings for $T = 20$. Each sub-figure shows the runs for a selected γ setting with the box-plots grouped by whether the Resample-Move step is used (the leftmost 5 box-plots) or not (the rightmost 5 box-plots), sorted in terms of number of samples used N.

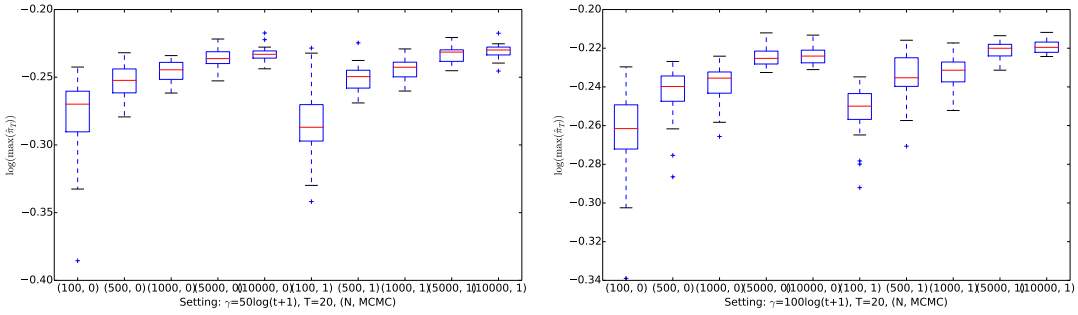

Figure 3.5: The $\log \pi_T(\hat{u}^*_{1:t})$ box-plots of 30 independent runs with γ settings: $50\log(t+1)$ and $100\log(t+1)$ and $T = 20$. Each sub-figure shows the runs for a selected γ setting with the box-plots grouped by whether the Resample-Move step is used (the leftmost 5 box-plots) or not (the rightmost 5 box-plots), sorted in terms of number of samples used N.

Lastly, we conclude this section by showing the mean of the recursive likelihood $m_{t|t-1}(u^*_{1:T})$ obtained in various runs in Figure 3.6 and their corresponding tracking errors in Figure 3.7. The results looks very promising. Except in the runs with the $\gamma = 1$ setting in which there is a lot of noise in tracking, all other runs are able to closely track the reference signal.

3.4 Conclusions

In this chapter, we introduce index tracking funds and summarise some causing factors of the tracking error between a fund and its benchmark index. We then present how the portfolio optimisation problem in terms of minimising the tracking error can be formulated as a stochastic control problem. This stochastic control problem can be turned into a path-space parameter estimation problem, in which Rao-Blackwellised SMC can be used to estimate these parameters efficiently. Lastly, we presents a proof-of-concept experiment with a oscillating wave as target reference signal. The results show that the SMC technique proposed is able to track the reference signal well.

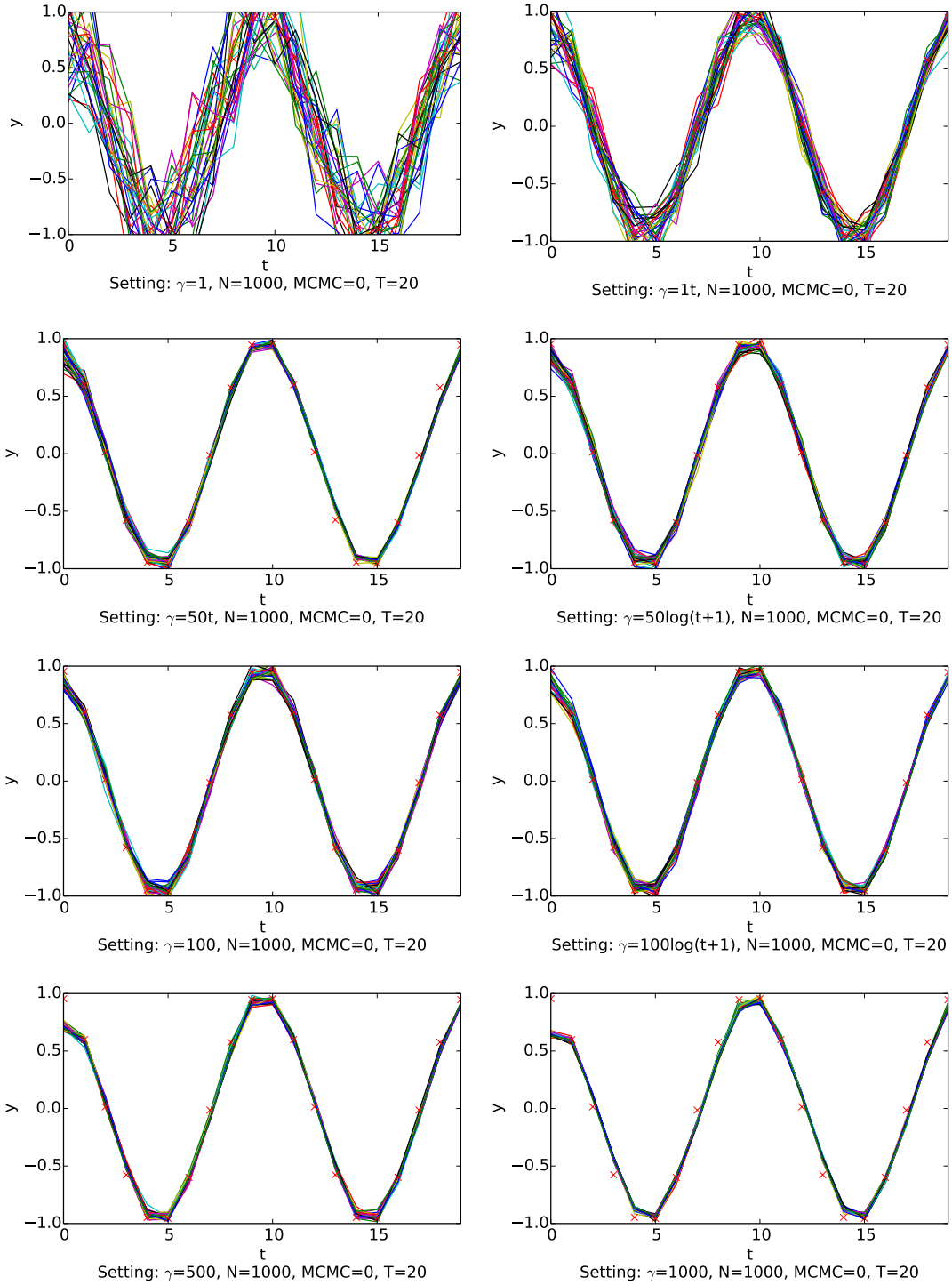

Figure 3.6: The means of the recursive likelihood $m_{t|t-1}(u^*_{1:T})$ obtained in 30 independent runs with $T = 20$, $N = 1000$, various γ settings and Resample-Move step disabled are plotted in lines with different colours. The target reference signal y^{ref} is also plotted with a 'x' marker.

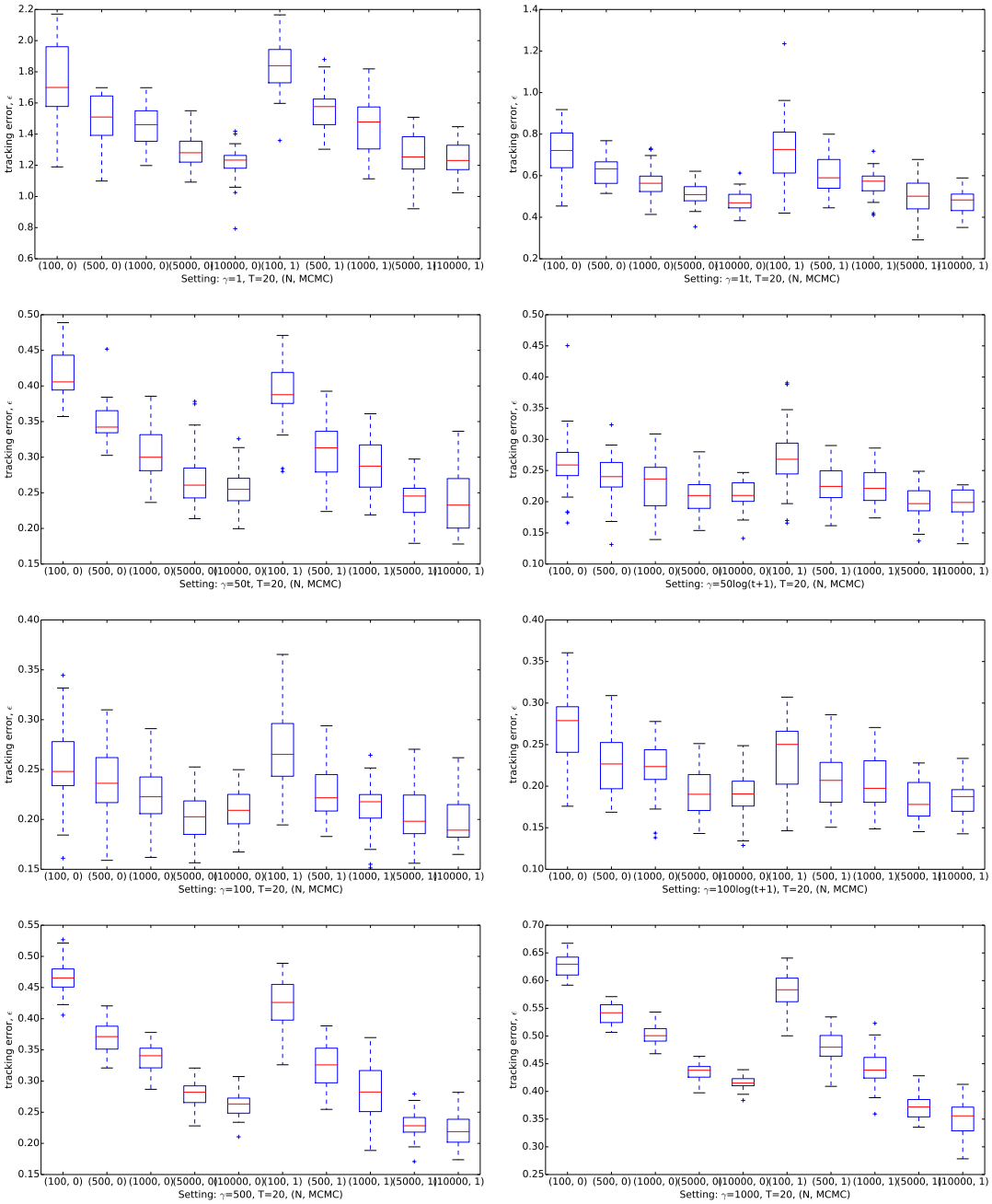

Figure 3.7: The box-plots of tracking errors of the model using $u^*_{1:T}$ obtained in 30 independent runs with $T = 20$, $N = 1000$, various γ settings and Resample-Move step disabled.

Chapter 4

Tracking the DAX index

In this chapter, we attempt to use the same SMC technique proposed to a track a real-world financial index. We look at the German's DAX (Deutscher Aktienindex) Index, which consists of 30 German's blue chip stocks listed on Frankfurt Stock Exchange as the constituents from 1st January 2014 up to 30th June 2014. It represents 80% of the aggregated prime standard's market capitalisation. The DAX Index is chosen for various pragmatic reasons. It is one of the major world indices that is tracked by various index funds, it has small number of components and the data is freely accessible[1]. For further details on the DAX Index methodology, refer [1].

4.1 Tracking the DAX4 sub-index

In the first experiment, we define a simple hypothetical sub-index, DAX4 consists of four stocks from the DAX Index that have the highest weights as of 2nd January 2014, namely Bayer (BAYN), Siemens (SIE), BASF (BAS) and Daimler (DAI). Then, the DAX4 index level is calculated as the simple weighted average of the adjusted close prices obtained from Yahoo Finance as follows:

$$y^{DAX4} = \sum_{s \in \mathcal{S}} w_s x_s \tag{4.1}$$

where y^{DAX4} is the index level, $\mathcal{S}$ consists of the four stocks and w_s and x_s are the weight and price of stock s respectively. Here the weights of these four stocks are assigned to be 0.4, 0.3, 0.2 and 0.1 respectively. The adjusted close price of each stock along with the calculated index level are shown in Figure 4.1.

[1] Actually, we first looked at Dow Jones Industrial Average (DJIA) Index obtained. It was later found some of the data is not publicly available. Having said that, the preliminary findings concur with the findings we have with the DAX Index reported here.

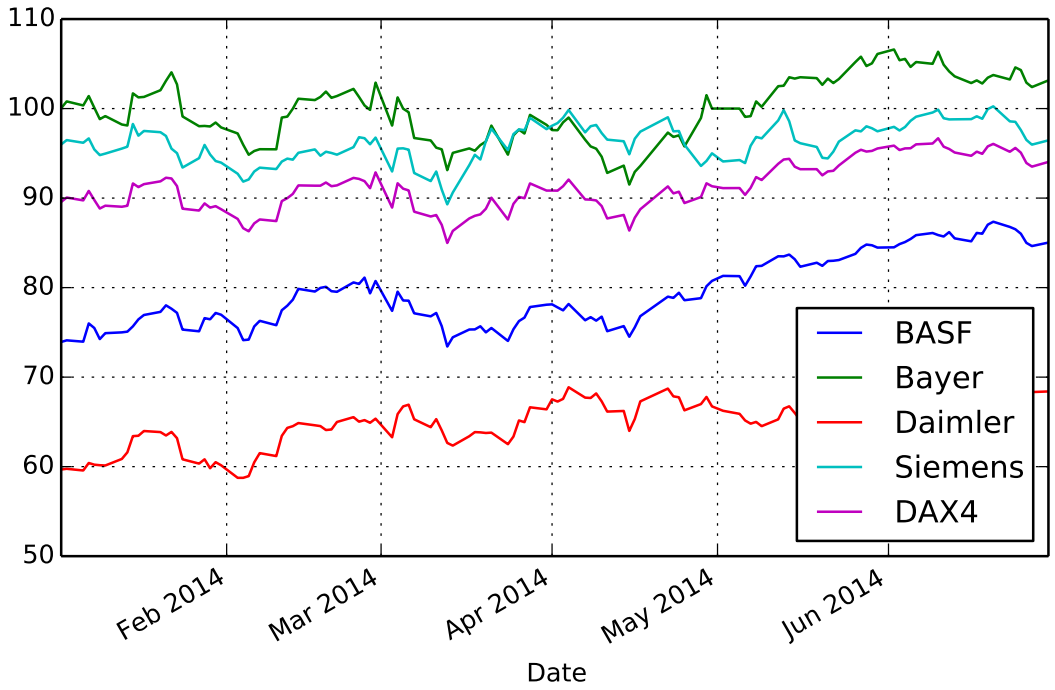

Figure 4.1: The adjusted close price of the 4 stocks and the calculated DAX4 index level, y^{DAX4}.

The portfolio optimisation problem is formulated as such y^{DAX4} is viewed as the target reference level that a portfolio manager attempt to replicate as close as possible by changing the holding position on each component stocks in $\mathcal{S}$, at the same time, minimise the transaction cost incurred in position changes.

To solve this problem using the SMC, the following state space modelling is used:

$$X_t = X_{t-1} + F_t(U_t) + W_t \tag{4.2}$$

$$Y_t = U_t^T X_t + 0.01 V_t \tag{4.3}$$

where $W_t \sim \mathcal{N}(\mu_{t_0}, \Sigma_{t_0})$, $V_t \sim \mathcal{N}(0, I)$, $\{X_t\}_{t\geq 0}$ is a vector of stock price processes modelled as Arithmetic Brownian Motion with drift, $\{U_t\}_{t\geq 0}$ is a vector of control input processes, each component represents the position we have for each stock, $F_t(U_t)$ can be viewed as the market impact on price due to position changes and is set to be $0.0001U_t$ here, μ_{t_0} and Σ_{t_0} are vector of the estimated mean of price changes and the estimated covariance matrix of the price changes and $\{Y_t\}_{t\geq 0}$ is the process represents the index level.

The values of μ_t and Σ_t here are estimated using Exponential Weighted Moving Average (EWMA) approach with the decay rate, λ, set to 0.94. For example, μ_t can

be easily calculated in a recursion form as follows:

$$\mu_t = \lambda\mu_{t-1} + (1-\lambda)r_t \tag{4.4}$$

where r_t is the price change at time step t. It is obvious from these equations that the EWMA estimate at time t depends on all the preceding estimates at time s, where $s < t$. To ensure the quality of EWMA estimate, a 6 months warm up period is used, i.e., the EWMA estimate is calculated starting from 1st July 2013 onwards. The estimates of μ_{t_0} and Σ_{t_0} obtained are as follows:

$$\mu_{t_0} = \begin{pmatrix} 0.03125069 \\ 0.20020445 \\ 0.08702132 \\ 0.18319923 \end{pmatrix} \Sigma_{t_0} = \begin{array}{c} \begin{matrix} BAYN & SIE & BAS & DAI \end{matrix} \\ \begin{pmatrix} 0.60011676 & 0.69688309 & 0.44198994 & 0.66709016 \\ 0.69688309 & 1.34168488 & 0.58394515 & 0.76533660 \\ 0.44198994 & 0.58394515 & 0.48855826 & 0.56500417 \\ 0.66709016 & 0.76533660 & 0.56500417 & 1.13794411 \end{pmatrix} \end{array} \begin{matrix} BAYN \\ SIE \\ BAS \\ DAI \end{matrix} \tag{4.5}$$

It is worth here to re-iterate here that this model is just a means to an end to demonstrate the idea. Other sophisticated models, e.g., Geometric Brownian Motion with drift, Jump Diffusion model, etc., are possible, perhaps even better.

Using this model, we can write the reward function as follows:

$$J(u_{1:T}, y_{1:T}^{ref}, x_0) = \mathbb{E}_{x_0}\left[\exp\left(-\frac{1}{2}\sum_{t=1}^{T}\left(||y_t^{ref} - u_t^T x_t||^2_{Q_t(u_t)^{-1}} + ||u_t - u_{t-1}||^2_{L_t}\right)\right)\right] \tag{4.6}$$

Here, we define $L = 0.1\,\mathrm{diag}(4,3,2,1)$ and also impose additional constraints on the holding position of each individual component of U_t such that the range is within $[0, 0.5]$. The idea behind these choices is to construct a scenario in which the transaction cost involved in changing the holding position on each individual component is different and impose a constraint on the maximum amount on the holding position of each individual component one portfolio can have.

For the sake of simplicity, we will use the SMC algorithm to target π^γ where $\gamma = 100$ and important sampling step, $U_{1,n}$ is sampled uniformly from the range bounded by the constraints mentioned above. The experiment is carried out for 30 independent runs, each with sample size is fixed to 10000 and no Resample-Move step is used here.

4.1.1 Results and discussion

As in previous experiment, the estimated optimal control $u^*_{1:T}$ and the mean of the recursive likelihood $m_{t|t-1}(u^*_{1:T})$ obtained in one of the independent runs are shown in Figure 4.4. The output of the model produce using the estimated $\hat{u}^*_{1:T}$ is closely

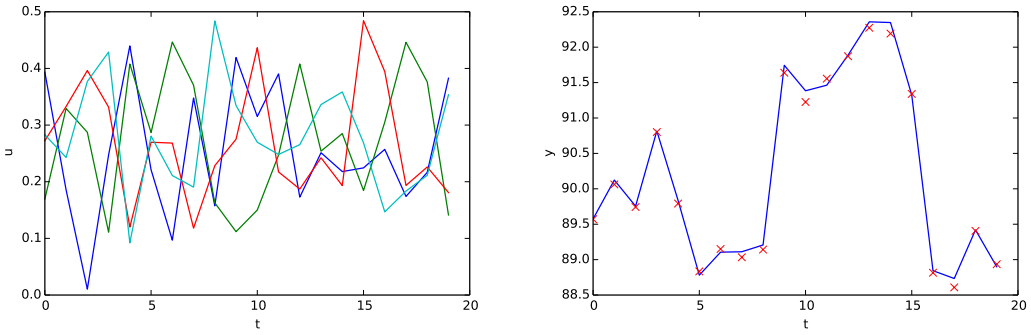

Figure 4.2: The estimated optimal control $u^*_{1:T}$ and the mean of the recursive likelihood $m_{t|t-1}(u^*_{1:T})$ obtained in one of the independent runs with the y^{DAX4} set to be the target reference using $L = 0.1\,\mathrm{diag}(4, 3, 2, 1)$.

tracking the reference index level, y^{DAX4}. The average annualised[1] tracking error of 0.023288 over the 30 runs. Having said that, the estimated set of control parameter $\hat{u}^*_{1:T}$ which represents the holding position of each stock does not seem very stable.

To reduce the number of changes in holding positions, the value of L matrix can be scaled up. To demonstrate this, the same experiment is re-run with $L = 5\,\mathrm{diag}(4, 3, 2, 1)$. The results are shown in Figure 4.3. The estimated set of control parameters $\hat{u}^*_{!:T}$ produced in this case is much more stable. However, this comes at a price on the tracking error. The average annualised tracking error with this setting is increased to 0.036386 over the 30 runs.

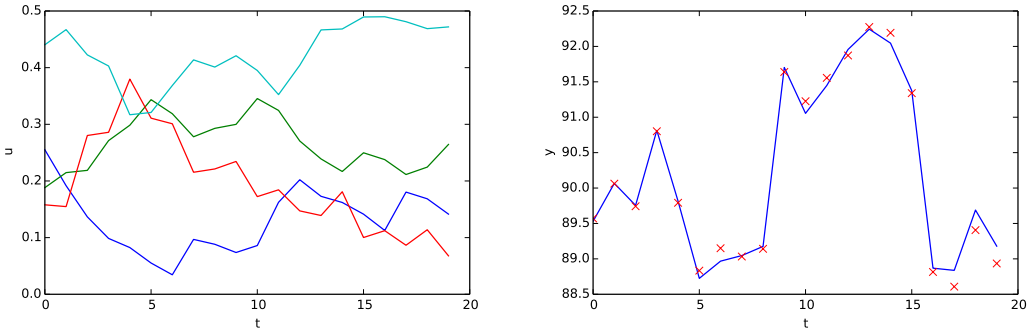

Figure 4.3: The estimated optimal control $u^*_{1:T}$ and the mean of the recursive likelihood $m_{t|t-1}(u^*_{1:T})$ obtained in one of the independent runs with the y^{DAX4} set to be the target reference using $L = 5\,\mathrm{diag}(4, 3, 2, 1)$.

[1] the annualisation factor is set to be $\sqrt{252}$ as the daily return is measured here.

4.1.2 Tracking the DAX index

Given the results of the experiment for DAX4 index look promising, we now attempt to track the original DAX index which consists of all 30 components using a similar model as follows:

$$X_t = X_{t-1} + F_t(U_t) + W_t \tag{4.7}$$

$$Y_t = 30U_t^T X_t + 0.01V_t \tag{4.8}$$

where $W_t \sim \mathcal{N}(\mu_{t_0}, \Sigma_{t_0})$, $V_t \sim \mathcal{N}(0, I)$, along with the same parameter settings as before except the following:

1. The target reference y^{ref} is the daily close level of the DAX index level.
2. The number of dimensions for U_t and X_t are now 30.
3. The L matrix is set to be $5I_{30\times 30}$, where $I_{n\times n}$ is the n dimensional identity matrix, i.e., the transaction cost of each component is all equal.

The experiment is carried out for 30 independent runs, each with sample size is fixed to 10000 and no Resample-Move step is used here.

As mentioned in Section 3.1.1, instead of tracking the index by fulling replicating all its components, sometimes it can be more efficient not only just in terms of computational, but also in terms of transaction cost and management perspective to track the benchmark index with only a subset of its component stocks.

To do this, only minimal changes are required to apply on the model proposed. We choose to use the top 6 highest weighted constituents of DAX index by including Allianz SE (ALV) and SAP SE (SAP), in which they all together constitute more than 50% of the weights for the DAX index level as of 2nd January 2014. Using only these 6 component stocks and a slight increase on the range of each component of U_t to within $[0, 1]$, the experiment is re-run for 30 times.

4.1.2.1 Results and discussion

As before, the estimated optimal control $u^*_{1:T}$ and the mean of the recursive likelihood $m_{t|t-1}(u^*_{1:T})$ obtained in one of the independent runs with the DAX index level set to be the target reference for both the full replication setting and partial replication setting are shown in Figure 4.4 and 4.5 respectively.

In the full replication setting, the average annualised tracking error over 30 runs is found to be 0.113118. However, the computational cost increases significantly in this

experiment due to the dimensionality increases in U and X. Using the computation cost of the previous $DAX4$ experiment as the baseline, the computational cost for this experiment is ≈ 7.2 times more per time step.

In the partial replication setting, the average annualised tracking error over 30 runs is found to be 0.062650. The estimated optimal control $u^*_{1:T}$ found in this setting is also much stable. This means smaller changes in holding positions and therefore lower transaction costs. In terms of computational cost, the relative computational cost factor in comparison to the baseline is approximately 1.1. This translates to approximately 6.5× speed-up in comparison to the full replication setting.

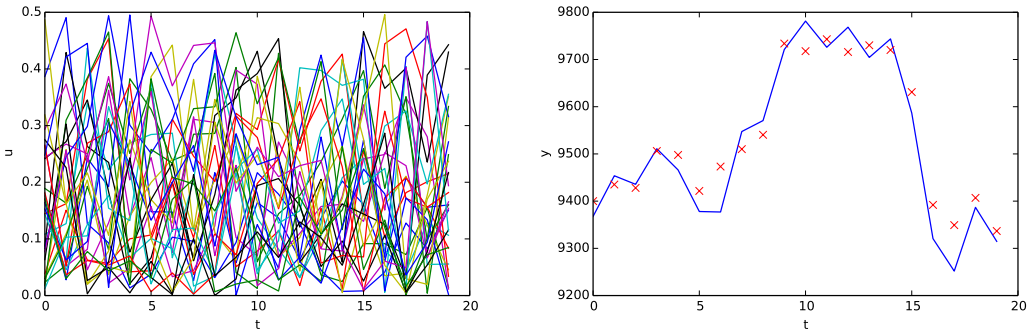

Figure 4.4: The estimated optimal control $u^*_{1:T}$ and the mean of the recursive likelihood $m_{t|t-1}(u^*_{1:T})$ obtained in one of the independent runs with the DAX index level set to be the target reference using all 30 component stocks.

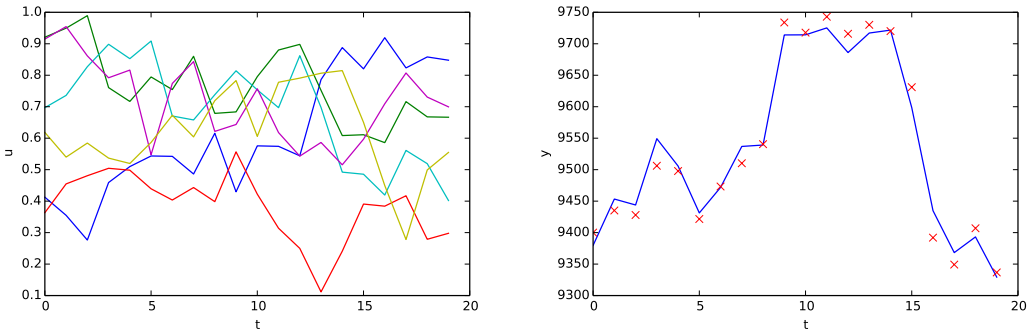

Figure 4.5: The estimated optimal control $u^*_{1:T}$ and the mean of the recursive likelihood $m_{t|t-1}(u^*_{1:T})$ obtained in one of the independent runs with the DAX index level set to be the target reference using only the top 6 highest weighted component stocks.

4.2 Model Predictive Control (MPC)

Up to this point, the discussion has been focussed on searching the optimal control parameters for a given time horizon $0 : T$ at time step 0. There are two issues here:

1. It is assumed that the future target reference signal $y^{ref}_{t'}$, $t' > 0$ is available for optimisation. In this application, but these are in fact not observable as of time step 0.

2. The optimisation is *only* performed at the beginning of the time horizon, i.e., the problem is treated as an open loop problem. Any new observation of the target reference signal, any of the control used and any of the output produced by the model over the time horizon are not fed back to the system.

To overcome these problems, instead of tracking the true y^{ref}, an estimate of y^{ref} is used as the target reference signal whenever the y^{ref} is not (yet) available. A simple estimator of y^{ref}_{t+s} at time step t is defined to be as follows:

$$\hat{y}^{ref}_{t+s} = y^{ref}_{t} + \beta s \tag{4.9}$$

where $s > 0$, β is the EWMA estimate ($\lambda = 0.94$) of the mean of daily changes of y^{ref}_t calculated at time step t. This estimator is chosen here for the sake of simplicity. Other estimators are possible. To simplify notation, we will define $\tilde{y}^{ref}_{t'}$ at any time step t to be as follows:

$$\begin{aligned} \tilde{y}^{ref}_{t'} &= y^{ref}_{t'} && \text{if } t' \leq t, \\ &= \hat{y}^{ref}_{t'} && \text{otherwise} \end{aligned} \tag{4.10}$$

To take into account the any new observable information into account over the time horizon, we employ the Model Predictive Control (MPC) framework. MPC is a receding horizon control algorithm. At time step t, the model defined is used to search for the optimal controls, $u^*_{t:t+H}$ for a finite horizon for $t : t + H$, where H is a parameter that defines the size of the short finite horizon. Then, only the first set of controls, i.e., u^*_t is applied to the system. At time step $t + 1$, the model is updated with new observed information. Using the updated model, the optimisation process is repeated for the same finite horizon, but shifted by 1 time step. This process is repeated until $t = T$. The MPC framework is summarised in Algorithm 7. MPC has achieved significant success in many applications, see [16, 19] for further details on MPC.

It is straight forward to apply the MPC framework for this problem. At every time step t, the SMC optimisation technique proposed can be executed to search for the

Algorithm 7 Model Predictive Control

```
function ModelPredictiveControl(T,H)
    Set t = 1.
    while t ≤ T do
        Search the optimal u*_{t:t+H} for the control problem of time horizon t : t + H.
        Apply the first set of the optimal control u*_t to the problem.
        Update the model states with any new information.
        Set t = t + 1.
    end while
end function
```

optimal $u^*_{t:t+H}$, but only the first set of controls, u^*_t is applied the model to produce the output y. Using this approach, the reward function that we attempt to optimise at each time step t is as follows:

$$J(u_{t:t+H}, \tilde{y}^{ref}_{t:t+H}, x_t) = \mathbb{E}_{x_0}\left[\exp\left(-\frac{1}{2}\sum_{s=t}^{t+H}\left(||\tilde{y}^{ref}_s - u_s^T x_s||^2_{Q_s(u_s)^{-1}} + ||u_s - u_{s-1}||^2_{L_s}\right)\right)\right] \tag{4.11}$$

This reward function is different from the previous ones as it does not depend any future information beyond time step t.

To demonstrate this idea, we re-run the experiment on the DAX index tracking with partial replication setting using this MPC framework with $H = 5$. The experiment is repeated for 30 times. Figure 4.6 shows the estimated optimal control $u^*_{1:T}$ and the mean of the recursive likelihood $m_{t|t-1}(u^*_{1:T})$ obtained in one of the independent runs. The tracking performance looks very good, with an average annualised tracking error of 0.062364 over the 30 runs. The estimated optimal control $u^*_{1:T}$ is also found to be reasonably stable.

In terms of computational cost, the empirical computational cost factor in relative to the baseline is approximately ≈ 5.2 per run. This makes sense as the complexity of each run here is $\mathcal{O}(NTH)$ as opposed to $\mathcal{O}(NT)$ in as in the previous case[1].

In the last experiment, we attempt to demonstrate the scalability of the technique by considering the same problem, but with an extended time period of $T = 125$, i.e., ≈ 6 months period. Figure 4.7 shows the estimated optimal control $u^*_{1:T}$ and the mean of the recursive likelihood $m_{t|t-1}(u^*_{1:T})$ obtained in one of the independent runs. Again, the results look promising, with an average annualised tracking error of 0.062364 over the 30 runs.

[1]We assume here the Resample-Move step is disabled. With Resample-Move step enabled, the complexity of this technique is $\mathcal{O}(NTH^2)$ as opposed to $\mathcal{O}(NT^2)$.

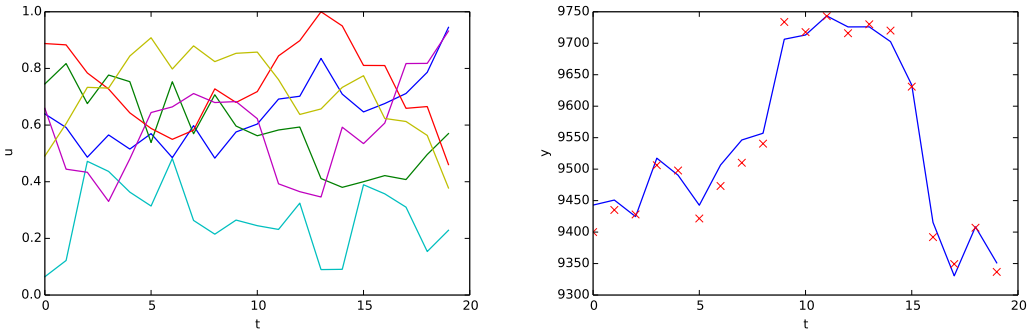

Figure 4.6: The estimated optimal control $u^*_{1:T}$ and the mean of the recursive likelihood $m_{t|t-1}(u^*_{1:T})$ obtained in one of the independent runs with the estimated DAX index level ($\hat{y}^{ref}_{t:t+H}$) set to be the target reference using the MPC framework with $H = 5$ and $T = 20$.

4.3 Conclusions

This chapter presents several experiments in apply the SMC technique to track a real-world financial index — the DAX index. It begins with a simple experiment in which a synthetic sub-index constructed with only 4 components to proof the concept. It then uses the same technique to demonstrate how the DAX index can be tracked with full replication as well as partial replication. Lastly, it introduces Model Predictive Control (MPC) framework and demonstrates how this technique can be integrated into the MPC framework easily. In all cases, the results show the SMC technique proposed is able to search for the optimal controls that produce an output that tracks the given reference signal well.

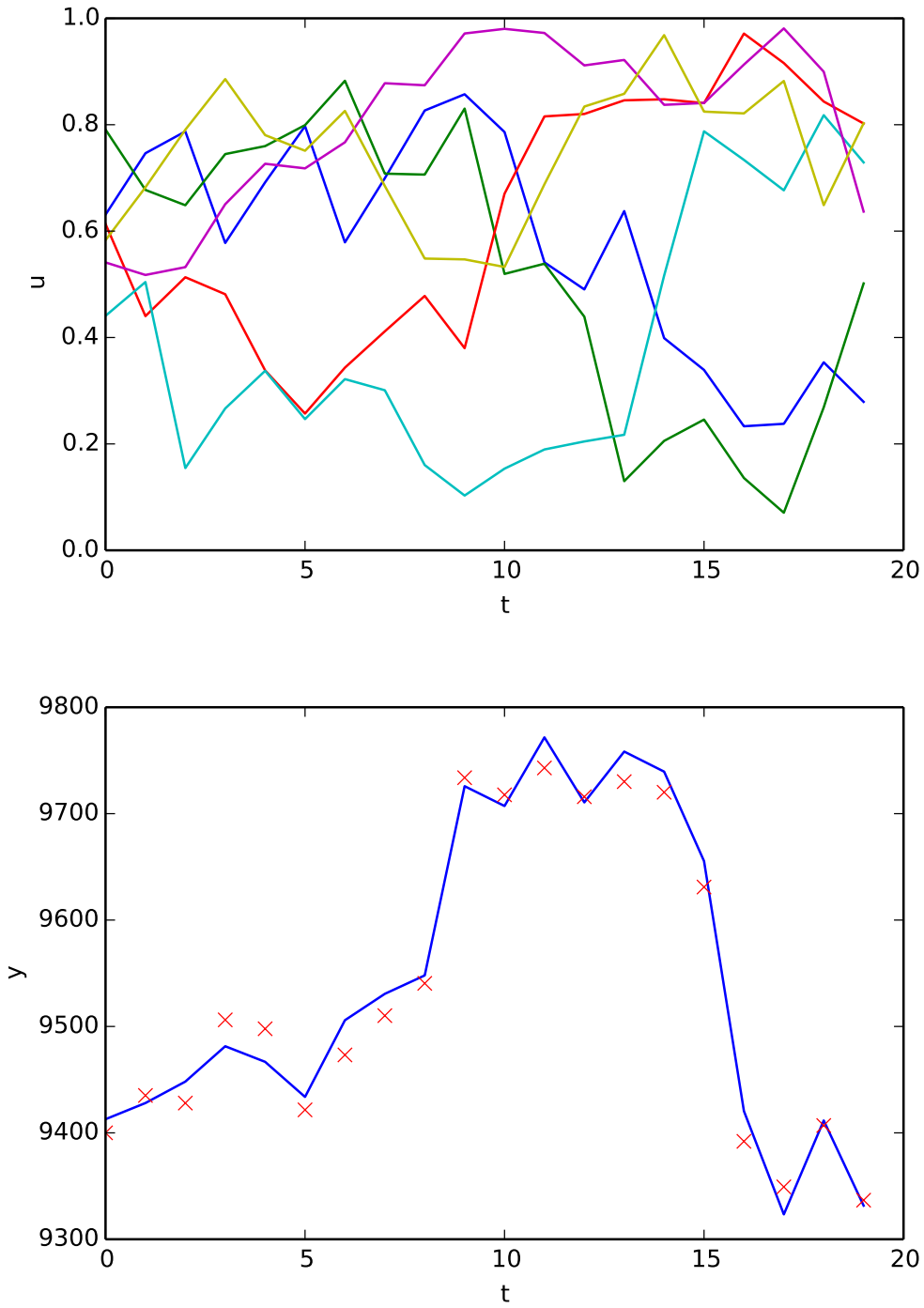

Figure 4.7: The estimated optimal control $u^*_{1:T}$ and the mean of the recursive likelihood $m_{t|t-1}(u^*_{1:T})$ obtained in one of the independent runs with the estimated DAX index level ($\hat{y}^{ref}_{t:t+H}$) set to be the target reference using the MPC framework with $H = 5$ and $T = 125$.

Chapter 5

Conclusions and Future Work

This thesis demonstrates how SMC can be used as an effective means of searching the optimal strategy for an index tracking fund in terms of minimising the tracking error and the transaction cost. The sensitivity of the technique with respect to the parameter settings and annealing techniques are explored. Several numerical examples using both synthetic and real-world data have been presented, showing how an index can be fully replicated or partially replicated with the transaction cost taken into account. It also presents the Model Predictive Control (MPC) framework and demonstrates how the SMC technique can be integrated into MPC framework. The experimental results show the SMC technique proposed is very promising.

5.1 Future Work

There are numerous possible directions for future work that have been identified during the course of this research.

1. More realistic models — The Arithmetic Brownian Model with drift used in this thesis is rather simple. A possible extension work is to consider a more advanced model, e.g., Geometric Brownian Model, Jump Diffusion model, etc. Moving away from a conditional Gaussian model is also a possibility. However, the inner Kalman Filter recursion is no longer optimal outsides the Gaussian assumption. A possible solution to this is to substitute the Kalman Filter with a nested SMC. This setup is commonly known as the SMC2 [5].

2. Parallel computation — The nested SMC setup inevitably adds consideration amount of computation requirements. A possible way to speed-up is to parallelise some of the steps in SMC. This is very straight-forward for many of the steps,

except the resampling step, which remains an interesting research topic on its own.

3. More complex financial indices — The benchmark index used in this thesis is rather simple. This can be potentially an issue. Having said that, this index is a simple, but by no means a "toy" index. It is one of the world's major financial indices. There are however indices with much large number of components, e.g., ≈ 1600 for MSCI World Index. This translates to a high dimensional problem in which may be difficult for the proposed model here. A possible way to cope with this is to just track the index with partial replication using pre-selected subset of components, perhaps using Principle Component Analysis (PCA). Alternatively, we could use a divide and conquer approach, tracking multiple sub-indices separately. There is still much to answer here, for example:

   - How to split an index into smaller components in a systematic manner?
   - How to deal with index components changes during announcement and implementation?

5.2 Closing remarks

The work reported in this thesis demonstrates a considerable degree of originality supported by extensive experimentation. The case studies are necessarily limited given the limited amount of time frame. However, the results demonstrate that portfolio optimisation using Sequential Monte Carlo techniques have very considerable promise. We recommend these approaches to the research community for further investigation.

Appendix A

Kalman Filter

In a conditional linear Gaussian state space as follows:

$$X_t^L = A_t(X_t^N)X_{t-1}^L + B_t(X_t^N)W_t + F_t(X_t^N) \tag{A.1}$$

$$Y_t = C_t(X_t^N)X_t^L + D_t(X_t^N)V_t + G_t(X_t^N) \tag{A.2}$$

where $\{X_t^N\}_{t\geq 0}$ is a non-linear Markov process, A_t, B_t, C_t, D_t, F_t, G_t are appropriate matrix/vector of X_t^N and $\{W_t\}_{t\geq 0}$ and $\{V_t\}_{t\geq 0}$ are independent sequences of standard Gaussian random variables, i.e., $W_t, V_t \sim \mathcal{N}(0, I)$. In such a case, the transition density and likelihood of this model are Gaussian distributions with centre lied at a point of a linear combination of the known x_t^N of the following form:

$$\begin{aligned} f_t(x_t^L \mid x_{t-1}^L, x_t^N) &= \mathcal{N}(A_t(x_t^N)x_{t-1}^L + F_t(x_t^N), B_t(x_t^N)B_t(x_t^N)^T) \\ g_t(y_t^L \mid x_t^L, x_t^N) &= \mathcal{N}(C_t(x_t^N)x_t^L + G_t(x_t^N), D_t(x_t^N)D_t(x_t^N)^T) \end{aligned} \tag{A.3}$$

Using the properties of Gaussian distribution, the integral involved can be resolved analytically. This leads to the widely used *Kalman Filter* [12] that has the following recursive solution as follows:

$$\mu_{t|t-1} = A_t(x_t^N)(\mu_{t-1|t-1})X_{t-1} + F_t(x_t^N) \tag{A.4}$$

$$\Sigma_{t|t-1} = A_t(x_t^N)\Sigma_{t-1|t-1}A_t(x_t^N)^T + B_t(x_t^N)B_t(x_t^N)^T \tag{A.5}$$

$$S_t = C_t(x_t^N)\Sigma_{t|t-1}C_t(x_t^N)^T + D_t(x_t^N)D_t(x_t^N)^T \tag{A.6}$$

$$m_{t|t-1} = C_t(x_t^N)\mu_{t|t-1} + G_t(x_t^N) \tag{A.7}$$

$$\mu_{t|t} = \mu_{t|t-1} + \Sigma_{t|t-1}C_t(x_t^N)S_t^{-1}(y_t - m_{t|t-1}) \tag{A.8}$$

$$\Sigma_{t|t} = \Sigma_{t|t-1} - \Sigma_{t|t-1}C_t(x_t^N)S_t^{-1}C_t(x_t^N)\Sigma_{t|t-1} \tag{A.9}$$

where $\mu_{t|t-1}$ and $\Sigma_{t|t-1}$ are the predicted mean and co-variance matrix of the state x_t^L, $m_{t|t-1}$ and S_t are the mean and co-variance matrix of the measurement at time t and

$\mu_{t|t}$ and $\Sigma_{t|t}$ are the estimated mean and co-variance matrix of the state x_t^L after seeing the observation y_t.

There are various extensions have been developed on top of this approach. For example, the Extended Kalman Filter (EKF) which uses Taylor Series expansion to linearise at the conditional variables locally [23], Unscented Kalman Filter which further extend the idea in EKF by only using a minimal set of well chosen samples [22], etc.

References

[1] Factsheet DAX. Websites, June 2014. 32

[2] Andrieu, C., Doucet, A., and Holenstein, R. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society Series B 72*, 3 (2010), 269–342. 15

[3] Berzuini, C., and Gilks, W. RESAMPLE-MOVE Filtering with Cross-Model Jumps. In *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, Eds., Statistics for Engineering and Information Science. Springer New York, 2001, pp. 117–138. 13

[4] Cappé, O., Moulines, E., and Ryden, T. *Inference in Hidden Markov Models (Springer Series in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005. 7, 15

[5] Chopin, N., Jacob, P. E., and Papaspiliopoulos, O. SMC2: An efficient algorithm for sequential analysis of state space models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 75*, 3 (2013), 397–426. 42

[6] Douc, R., and Cappe, O. Comparison of resampling schemes for particle filtering. In *Image and Signal Processing and Analysis, 2005. ISPA 2005. Proceedings of the 4th International Symposium on*. 12

[7] Doucet, A., and Johansen, A. M. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of Nonlinear Filtering 12* (2009), 656–704. 5

[8] Gilks, W. R., and Berzuini, C. Following a moving targetMonte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 63*, 1 (2001), 127–146. 13

[9] Godsill, S., Doucet, A., and West, M. Maximum a Posteriori Sequence Estimation Using Monte Carlo Particle Filters. *Annals of the Institute of Statistical Mathematics 53*, 1 (2001), 82–96. 23

[10] Johansen, A. M., Evers, L., and Whiteley, N. Monte Carlo Methods. *Lecture Notes 200* (2010). 12, 13, 14

[11] Johnson, B., Bioy, H., Kellett, A., and Davidson, L. On the Right Track: Measuring Tracking Efficiency in ETFs. 17, 18

[12] Kalman, R. E. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME–Journal of Basic Engineering 82*, Series D (1960), 35–45. 44

[13] Kantas, N., Moral, P. D., and Vinter, R. Particle methods for Stochastic Regulation: towards an application for power scheduling. Tech. rep., Department of Electrical and Electronic Engineering, Imperial Colege, London, March 2011. 19, 21, 23, 27

[14] Kitagawa, G. Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models. *Journal of Computational and Graphical Statistics 5*, 1 (1996), 1–25. 12

[15] Liu, J. S., and Chen, R. Sequential Monte Carlo Methods for Dynamic Systems. *Journal of the American Statistical Association 93* (1998), 1032–1044. 12

[16] Maciejowski, J. *Predictive Control: With Constraints*. Pearson Education. Prentice Hall, 2002. 38

[17] Markowitz, H. Portfolio Selection. *The Journal of Finance 7*, 1 (1952), 77–91. 1, 16

[18] Rabiner, L. R. A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of IEEE* (1989), pp. 257–286. 15

[19] Rawlings, J., and Mayne, D. *Model Predictive Control: Theory and Design*. Nob Hill Pub., 2009. 38

[20] Robert, C. P., and Casella, G. *Monte Carlo Statistical Methods (Springer Texts in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005. 5

[21] Rockafellar, R. T., and Uryasev, S. Optimization of Conditional Value-at-Risk. *Journal of Risk 2* (2000), 21–41. 1

[22] Wan, E. A., and Merwe, R. V. D. The Unscented Kalman Filter. In *Kalman Filtering and Neural Networks* (2001), Wiley, pp. 221–280. 45

[23] Welch, G., and Bishop, G. An Introduction to the Kalman Filter. Tech. rep., University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, 1995. 45

[24] Whittle, P. *Risk-sensitive Optimal Control*. Wiley-Interscience series in systems and optimization. Wiley, 1990. 19