Factor Stochastic Volatility Models for Portfolio Construction

Taylor Brown

B.A., University of Connecticut (2010)

M.S., University of Connecticut (2013)

A Dissertation Presented to the Graduate Faculty

of the University of Virginia in Candidacy for the Degree of

Doctor of Philosophy

Department of Statistics

University of Virginia

May, 2018

Abstract

We propose a new factor stochastic volatility model that increases the accuracy of short-term forecasts for financial assets. Our new model, called the Markov-Switching Loadings (MSL) model, extends previous models by including latent processes that control the mean vectors and covariance matrices of random sub-vectors of returns. In addition, we describe our estimation routine, a novel particle Markov chain Monte Carlo algorithm, which allows for efficient estimation of a wide range of models and requires little tuning or model-specific derivations. We give two specifications of the MSL model, and both are estimated and used to generate out-of-sample forecasts for weekly returns of Select Sector SPDR exchange-traded funds over a time window spanning the 2008 financial crisis. We examine these forecasts from a statistical perspective, as well as through a financial lens, by analyzing the returns of a hypothetical investment strategy.

Acknowledgments

To Clare, and to my family.

Contents

1 Introduction **1**

1.1 Objective and Outline 1

2 State Space Models and Particle Filtering **5**

2.1 Definitions and Background 5

2.2 SSMs for Financial Applications 6

2.2.1 Motivation: Portfolio Selection 6

2.2.2 Approximating the Forecast Distribution: Approach 1 7

2.2.3 Approximating the Forecast Distribution: Two More Approaches 8

2.2.4 Approximating the Forecast Distribution: Approach 4 9

2.3 Particle Filtering with Known Parameters 10

2.3.1 Sequential Importance Sampling (SIS) 12

2.3.2 Sequential Important Sampling with Resampling (SISR) . . . 15

2.3.3 The Auxiliary Particle Filter (APF) 18

2.3.4 The Rao-Blackwellized Particle Filter (RBPF) 19

2.3.5 The Rao-Blackwellized Auxiliary Particle Filter (RBAPF) . . 22

2.3.6 A Note on Computation 24

3 Parameter Estimation with Markov Chain Monte Carlo Techniques **29**

3.1 General MCMC 29

3.2 MCMC for SSMs 34

3.2.1 Closed-Form Gibbs Sampling 35

3.2.2 Advanced Gibbs Sampling 35

3.2.3 Particle Based Approximations to State Distributions 36
3.2.4 Particle Markov Chain Monte Carlo 37
3.2.5 Accelerating the PMMH Algorithm 42

4 Multivariate Stochastic Volatility Models 47

4.1 Introduction 47
4.2 An Initial Model 48
4.3 Identifiability of the Initial Model 49
4.4 Real Data Analysis with the Initial Model 51
4.4.1 Parameter Estimation 53
4.4.2 Out-of-Sample Forecasting 54
4.4.3 Alternative Models 58
4.5 The Markov-Switching Loadings Models 59
4.5.1 MSL1 59
4.5.2 MSL2 62
4.6 Estimation of MSL Models 64
4.6.1 Estimation of MSL1 64
4.6.2 Estimation of MSL2 67
4.7 Out-of-Sample Performance 69

5 Summary 73

5.1 Overview 73

A Appendix 75

A.1 Proofs 77
A.2 Deriving MSL Mean Forecasts 84
A.3 Deriving MSL Covariance Matrix Forecasts 85
A.4 Plots 89

List of Figures

2-1 Bootstrap filter resampling comparison 17

3-1 Noisy AR(1) PMMH scatterplot matrix 42

3-2 Noisy AR(1) PMMH traceplot 42

3-3 Noisy AR(1) PMMH correlograms 43

4-1 SPDR ETFs: returns and cumulative returns 52

4-2 Jacquier et al. out-of-sample performance 55

A-1 PMMH samples of **B** of Jacquier et al. model 89

A-2 PMMH samples ϕ, μ, σ^2 of Jacquier et al. model 90

A-3 PMMH samples of **R** of Jacquier et al. model 91

A-4 Jacquier et al. out-of-sample log volatilities and forecast scores 92

A-5 Jacquier et al. out-of-sample residuals 92

A-6 Jacquier et al. out-of-sample prewhitened residuals 93

A-7 Jacquier et al. out-of-sample naively whitened residuals 93

A-8 Cumulative returns from minimum variance portfolio strategy using Jacquier et al. model 94

A-9 PMMH samples of **B** of MSL1 model 94

A-10 PMMH samples of ϕ_i of MSL1 model 94

A-11 PMMH samples of μ_i of MSL1 model 95

A-12 PMMH samples of σ_i^2 of MSL1 model 95

A-13 PMMH samples of **R** of MSL1 model 95

A-14 PMMH samples of λ_i of MSL1 model 96

A-15 PMMH samples of p of MSL1 model 96
A-16 PMMH samples of $\mathbf{B}$ of MSL2 model 96
A-17 PMMH samples of μ_i Samples of MSL2 model 97
A-18 PMMH samples of ϕ_i of MSL2 model 97
A-19 PMMH samples of σ_i^2 of MSL2 model 97
A-20 PMMH samples of $\mathbf{R}$ of MSL2 model 98
A-21 PMMH samples of λ_i of MSL2 model 98
A-22 PMMH samples of p_i of MSL2 model 98
A-23 Comparison of 3 Models' Forecast Scores 99
A-24 Whitened Forecast Errors for MSL1 and MSL2 100
A-25 Standardized Forecast Errors for MSL1 and MSL2 101
A-26 Comparison of 3 Models' Filtered Log Volatilities 102
A-27 Comparison of 3 Models' Investment Performance 103

List of Tables

3.1 Noisy AR1 Parameter Estimates 43

4.1 Initial Model Beta Estimates 54
4.2 Initial Model State Process Estimates 54
4.3 Initial Model Error Variance Matrix Diagonal Estimates 54
4.4 MSL **B** Estimates 66
4.5 MSL State Process Parameter Estimates 66
4.6 MSL **R** Estimates 67
4.7 MSL λ_i and p_i Estimates $(i = 1, 2)$ 67
4.8 MSL2 **B** Estimates 68
4.9 MSL2 State Process Parameter Estimates 68
4.10 MSL2 **R** Estimates 68
4.11 MSL2 λ_i and p_i Estimates $(i = 1, 2)$ 69

Chapter 1

Introduction

Objective and Outline

Modeling a vector-valued time series of stock returns, or more generally the returns of any basket of financial instruments, is a very important and difficult task. The forecasts are important because they assist in asset pricing, constructing portfolios, and managing risk. The difficulty owes itself to many reasons.

First, each time's forecast mean deviates from the zero vector in very subtle and irregular ways. In other words, the mean is not likely a simple function of the process' past values. Also, this quantity is very small relative to the variance terms. In practice, estimation of the mean is often side-stepped: it is often assumed to be zero after suitable transformations of the data, or it is assumed to be some known constant.

The correlations and variances are difficult to estimate as well. The first difficulty is that a forecast's covariance matrix has many terms. If the return vector at each time is $y_t \in \mathbb{R}^{d_y}$, then there are $d_y(d_y+1)/2 = \mathcal{O}(d_y^2)$ terms in the forecast's covariance matrix. Second, these quantities are known to be time-varying as well, and any procedure used to estimate them should take into account that they evolve together on the space of positive-definite matrices.

Lastly, the type of data one uses constrains his choice of model. All financial data arrive in real time, so only "causal" models may be considered if one's primary goal is forecasting. This restriction guarantees that forecasting or prediction equations are available, and hopefully, the model's computational costs are low enough to allow its use in real time applications using data arriving at the desired sampling frequency.

Mindful of these constraints, this document pursues three goals:

1. to detail a practical portfolio managing framework that allows the use of a wide variety of models;

2. to establish the usefulness of a relatively recent class of estimation algorithms, and to increase the computational efficiency of some of its member algorithms;

3. to propose a new multivariate stochastic volatility model and to demonstrate its forecasting ability.

In light of these goals, this document is organized as follows. First, chapter (2) describes general state space models, along with several key results. The primary focus of this chapter is the description of algorithms that accomplish the task of "filtering." For many models, filtering is often impossible to do exactly, and it is involved both in parameter estimation and in developing recursive forecasting equations. These results will be used in all of the following chapters.

Chapter (3) reviews the literature surrounding Bayesian parmeter estimation. This chapter explains Markov chain Monte Carlo strategies generally, and it also details how they are used to estimate SSMs in particular. Historically, Gibbs-based strategies have been the most popular for the types of models that are considered in the last chapter. However, the estimation technique that this document emphasizes is a version of the relatively recent particle marginal Metropolis-Hastings algorithm. Using this method carries with it three important benefits: it allows for a wider array of models to be estimated and does not constrain the choice of prior distributions for these models, they are easier to program owing to the fact that there are no model-specific derivations of conditional posterior distributions, and there likely is no computational tradeoff when compared with Gibbs sampling.

Finally, in chapter (4), factor stochastic volatility models are introduced. These are a distinct subclass of state space models that meet all of the requirements mentioned above, and they are shown to have tremendous forecasting ability. Following a brief literature review, two new formulations of a model are proposed that are shown to have an increased capacity to represent some of the stylized features of financial returns: the Markov-Switching Loadings Model 1 and the Markov-Switching Loadings Model 2. These models, along with some of their competitors, are estimated, and their forecasting abilities are ranked using scoring rules and hypothetical performance on a widely-used investing strategy.

The primary motivation for the work presented in this document is to develop high-performance portfolio management strategies. Every investor, in addition to monitoring the performance of his existing models, faces the ongoing challenge of developing models that are more predictive, that can handle higher-dimensional data, and that can handle data arriving at higher sampling frequencies. Ideally this investor would pursue a top-down approach: as a first step, he would be able to specify a model without regard to any practical issues related to its use. This document also pursues a top-down approach. This will explain the overall preference for models that require sampling-based algorithms, as well as the preference for Metropolis-Hastings algorithms over Gibbs-based ones.

Chapter 2

State Space Models and Particle Filtering

Definitions and Background

A state space model (SSM) is defined by two things: a $\mathcal{Y} \subset \mathbb{R}^{d_y}$-valued observable time series $\{y_t\}_{t=1}^T \coloneqq y_{1:T}$, and an $\mathcal{X} \subset \mathbb{R}^{d_x}$-valued unobservable or latent time series $\{x_t\}_{t=1}^T \coloneqq x_{1:T}$. Specifying a SSM requires the selection of several probability distributions all specified by some parameter vector θ. For convenience and clarity of presentation, all probability measures in this paper will be assumed to be dominated by the Lebesgue measure, and thus admit density functions. What are chosen are $f(x_1 \mid \theta)$, the first time's state distribution, along with the state transition densities, $\{f(x_t \mid x_{t-1}, \theta)\}_{t=2}^T$, and the observation densities, $\{g(y_t \mid x_t, \theta)\}_{t=1}^T$.

Constructing the joint density of the complete data, which includes both the observed and unobserved portions, is straightforward because $y_{1:T}$ are conditionally independent given the states, and because the states are assumed to possess the Markov property. Specifically, this means

$$p(y_{1:T} \mid x_{1:T}, \theta) = \prod_{t=1}^{T} g(y_t \mid x_t, \theta) \tag{2.1}$$

and

$$p(x_{1:T} \mid \theta) \quad = \quad f(x_1 \mid \theta) \prod_{t=2}^{T} f(x_t \mid x_{t-1}, \theta). \tag{2.2}$$

The product of (2.1) and (2.2) yields the complete data likelihood: $p(x_{1:t}, y_{1:t} \mid \theta)$, and integrating out the unobserved $x_{1:t}$ from this yields the marginal likelihood $p(y_{1:t} \mid \theta)$.

Most quantities of interest at each time can be written as a conditional expectation involving integration with respect to the joint smoothing density, $p(x_{1:t} \mid y_{1:t}, \theta)$. Integration is generally difficult because this density is not tractable for general state space models; only in two scenarios is direct calculation possible. The first scenario is when $\mathcal{X}$ is finite, as in the case of what are usually referred to as Hidden Markov Models (HMMs). Integrating out $x_{1:t}$ from the complete data likelihood is expressed as a sum over all length t state sequences. The second scenario is when $p(x_{1:t} \mid \theta)$ is conjugate to $p(y_{1:t} \mid x_{1:t}, \theta)$, giving the smoothing density a known form. This arises in the case of linear-Gaussian state space models, wherein the use of the Kalman Filter and the Kalman Smoother is possible [Shumway and Stoffer, 2006].

SSMs for Financial Applications

Motivation: Portfolio Selection

Before particle filters are defined, consider as a motivating example the task of portfolio selection. An investor sits at time t and possesses the set of available information $y_{1:t}$. Using a decision variable w_t, he seeks to properly control or monitor the distribution of his future returns $p(w_t' y_{t+k} \mid y_{1:t})$. This w_t is a vector of weights, where each weight element describes the amount of his wealth used to buy a particular asset.

There are two examples where summary measures of this probability distribution are commonly used. First, techniques falling under the category of Mean-Variance optimization methods [Markowitz, 1952] will pick, in some sense decided upon by the investor, an optimal weight vector. The objective function that w_t is chosen to maximize will depend on the mean and variance of this distribution, and will also depend on how the investor will value expected returns compared with the variance. Another example includes what is called Value at Risk (VaR), which deals with estimating lower quantiles of this random variable. For example, the absolute value of the lower 1% quantile of the next day's return distribution would consitute another measure of investment risk. This one-number summary is highly reportable, and it is common for practitioners to place restrictions on how large this number can be.

The following sections describe how these quantities can be obtained using particle filtering approaches that work for almost any state space model. Afterwards, particle filters are defined, and several specific algorithms are described in detail.

Approximating the Forecast Distribution: Approach 1

In the case of no model or parameter uncertainty, the forecast distributions can be approximated in a timely manner using weighted samples from a particle filter. This assumption of known parameters may be justified for particular models, or if a MCMC algorithm yields very low-variance posterior distributions. Each time's approximation to the filtering distribution can be recursively computed using the weighted samples from the previous time period. Once one accomplishes the intermediate step of obtaining this filtering distribution $p(x_t \mid y_{1:t}, \theta)$, one may then use his weighted samples to approximate a k-step ahead forecast distribution for his portfolio's return. The following decomposition would be used:

$$p(w_t' y_{t+k} \mid y_{1:t}, \theta) = \int p(w_t' y_{t+k} \mid x_t, \theta) p(x_t \mid y_{1:t}, \theta) dx_t. \tag{2.3}$$

Recall that the weight vector w_t is either arbitarily chosen by the investor, or a function of one or both of θ and x_t. If $p(w_t' y_{t+k} \mid x_t, \theta)$ or some approximating moment of this distribution is tractable, then the samples and weights approximating the filtering distribution may be used to evaluate the mixture approximating (2.3). If $p(w_t' y_{t+k} \mid x_t, \theta)$ is not tractable, then the observation density $g(y_{t+k} | x_{t+k}, \theta)$ most likely is. Here, one may use the last resort of simulating forward k times the states, perhaps by using the model's transition density $f(x_t \mid x_{t-1}, \theta)$, and then use these samples distributed according to $p(x_{t+k} \mid y_{1:t}, \theta)$ to construct a mixture of the densities $\{g(y_{t+k} \mid x^i_{t+k}, \theta)\}$. This option obviously comes at the expense of higher Monte Carlo variance and slightly increased computation time.

Whichever option the investor chooses to target (2.3), the recursive nature of the particle filter allows him to revise his decisions efficiently upon receiving new information on the following day. First, he will revise his filtering approximation, and all of these computations will begin anew.

However, it is unusual for the parameters to be known. Indeed, this is often just an assumption made for practical convenience. The following sections draw on this line of reasoning to detail two complete approaches which do not underestimate the uncertainty of the parameters.

Approximating the Forecast Distribution: Two More Approaches

Approaches 2 and 3 for dealing with uncertainty about θ is suggested by the following decomposition:

$$p(w_t' y_{t+k} \mid y_{1:t}) = \iint p(w_t' y_{t+k} \mid x_t, \theta) p(x_t, \theta \mid y_{1:t}) dx_t d\theta. \tag{2.4}$$

This decomposition suggests two possible approaches. The first is to begin by simulating from $p(x_t, \theta \mid y_{1:t})$ at each time point, and then using these samples to calculate an expectation of $p(w_t' y_{t+k} \mid x_t, \theta)$. For example, this is similar to the approach taken in [Amisano and Geweke, 2008] when calculating approximations to the predictive log Bayes factors. Simulations can be obtained from different MCMC techniques for both the parameters and hidden states, or by using MCMC techniques to target the parameters alone, and using these parameters to run particle filters to estimate the terminal state. This approach has the highest potential for accuracy, but this comes at the expense of large and increasing computing times.

The second approach suggested by this decomposition is to rewrite the same model in an alternative form, a form that holds the unknown parameters as part of the state vector and features as a filtering distribution $p(x_t, \theta \mid y_{1:t})$. The static parameters, when thought of as a Markov chain, have known transition dynamics. At first glance, it appears that the previous knowledge about particle filters can be used for inference regarding θ in addition to the state. A particle filter would be set to run, and at any time, an up-to-date distribution for all of the unknown quantities would be given.

However, using a particle filter on this new model is more difficult owing to the lack of an "exponential forgetting property" of the filtering distributions [Kantas et al., 2014]. This property is a sufficient condition for the asymptotic variances to be bounded uniformly in time. When this property is not satisfied, it is often observed empirically that when weight vectors $\{w^i\}$ are calculated for each sample $\{x_t^i, \theta^i\}$, the samples used will be of extremely high variance. For large enough t, most, if not all of the samples will have the same value for the parameter. As a result, these samples will be unfit to approximate (2.4).

One way to combat this problem comes from assuming that these parameters are in fact not parameters at all, but are actually time-varying states. When this is done, often known-variance random walk dynamics are assumed for the new portions of the augmented state vector [Kitagawa, 1998]. Then the SSM's state transition will satisfy a sufficient condition (Assumption 9.4.9 [Cappé et al., 2005]) to guarantee exponential forgetting. This option is akin to Approach 1 in that the parameters (here they are tuning parameters) are assumed to be known. But this approach also has the disadvantage that there are no directly applicable asymptotic results: even though the augmented model might take advantage of central limit theorems targetting $p(\theta_t \mid y_{1:t})$, it is hard to see how this new working model's filtering distribution compares to the old model's posterior distribution $p(\theta \mid y_{1:t})$.

Approximating the Forecast Distribution: Approach 4

This document advocates the fourth approach. It starts with obtaining a marginal posterior distribution for the parameter using a stretch of historical data $y_{1:n}$. This distribution could be used for the k-step ahead forecasts starting at time n straightforwardly. But afterwards, once one is conditioning on historical information past time n, instead of re-simulating the newer posterior as in approach (2), the now outdated parameter posterior distribution is simply used again. This approximation is necessary if re-running the MCMC procedure is too computationally expensive. The decomposition that would be used here builds on (2.3). It is

$$\begin{aligned} p(w_t' y_{n+d+k} \mid y_{1:n+d}) &= \iint p(w_t' y_{n+d+k} \mid x_{n+d}, \theta) p(x_{n+d} \mid y_{1:n+d}, \theta) p(\theta \mid y_{1:n+d}) dx_{n+d} d\theta \\ &\approx \iint p(w_t' y_{n+d+k} \mid x_{n+d}, \theta) p(x_{n+d} \mid y_{1:n+d}, \theta) p(\theta \mid y_{1:n}) dx_{n+d} d\theta. \end{aligned} \tag{2.5}$$

where d is the number of days since the last MCMC batch posterior estimation. (2.5) suggests first drawing parameter values $\{\theta^i\}$ from the samples targetting $p(\theta \mid y_{1:n})$ at time n. Then, a particle filter is instantiated for each parameter, and from these, one will have samples from $p(x_{n+d} \mid y_{1:n+d}, \theta^i)$ for any $d \geq 0$. From there, the same mixture distribution choices detailed in the previous section will be available.

In practice, it is advisable to occasionally re-estimate the marginal posterior of the parameter vector on a recent stretch of historical data, not allowing d to get too big. It is natural to ask what the "sweetspot" for $d \geq 0$ is. The smaller it is, the more precise the forecast distribution will be, and the more expensive the calculations will become. On the other hand, if d is too large, then the posterior might be completely outdated. This is an issue worth considering; however, this paper does not make any attempts to answer this question.

Before these ideas can be discussed any further, particle filtering with known parameters must be explained. An overview of this is detailed in the next section.

Particle Filtering with Known Parameters

Even though the marginal filtering distribution of the states is usually the target distribution, particle filtering algorithms are more easily understood when the target density is the joint smoothing density $p(x_{1:t} \mid y_{1:t})$. After all, if one has samples from this density, $x_{1:t-1}$ can easily be integrated out. So there is nothing lost by taking this seemingly indirect route.

Observing a key decomposition suggests a recursive technique to approximate the densities at each time with weighted samples. As time passes, particles will be lengthened, and the weights corresponding to each of them will be adjusted. In this section, it is assumed that θ is known, so this dependence is not written in the notation for the densities.

Particle filtering algorithms for estimating $p(x_{1:t} \mid y_{1:t})$ are based on Bayes' rule and the following two-step recursion, which follows from the Markov assumption of a state space model:

$$p(x_{1:t} \mid y_{1:t-1}) = f(x_t \mid x_{t-1}) p(x_{1:t-1} \mid y_{1:t-1}) \tag{2.6}$$

$$p(x_{1:t} \mid y_{1:t}) = \frac{g(y_t \mid x_t) p(x_{1:t} \mid y_{1:t-1})}{p(y_t \mid y_{1:t-1})}. \tag{2.7}$$

As was mentioned in the previous section, computation of the denominator in (2.7) is impossible for all but two cases. If $p(x_t \mid x_{t-1}, y_{1:t-1})$ is conjugate to $g(y_t \mid x_t)$, then $p(x_{1:t} \mid y_{1:t})$ has a known distribution. Or, if $\mathcal{X}$ is finite, then the normalizing constant is a finite sum. The downside to designing a state space model with this in mind is that the models are often too simplistic. They do not always fit the data well, and this is likely to be true in particular for financial returns data.

Theoretically, particle filtering can be applied to a state space model of any form. At its core, it is best thought of as the combination of two things: importance sampling and resampling. This document's synopsis closely follows the description of [Doucet and Johansen, 2011], except for its explanation of auxiliary particle filters. The importance sampling portion is sequential in nature, targeting at each time t the smoothing density $p(x_{1:t} \mid y_{1:t})$. At time $t-1$ the user will have N samples of the path $x_{1:t-1}$ from some importance density and weights corresponding with each of these samples. These weights are calculated based on how "close" the importance density is to the complete data likelihood, which is proportional to the target.

The creation of this class of algorithms is usually credited to [Gordon et al., 1993] or [Kong et al., 1994], who were in turn inspired by the nonsequential "factored sampling" algorithm of [Grenander et al., 1991]. Both deal with the situation of missing data, but only the former deals explicitly with state space models. What follows is a more general explanation of the basic Sequential Monte Carlo algorithm: Sequential Importance Sampling. From here, more recent strategies can be seen as being based on the same underlying principle.

Sequential Importance Sampling (SIS)

Start by picking a proposal distribution[1] that factors as

$$q(x_{1:t} \mid y_{1:t}) = q_t(x_t \mid x_{1:t-1}, y_{1:t}) q(x_{1:t-1} \mid y_{1:t-1}), \tag{2.8}$$

that dominates the complete data likelihood, is tractable, and can generate samples. In practice, this amounts to choosing q_t at each time point. Then, it can be seen that

$$\begin{aligned} p(x_{1:t} \mid y_{1:t}) &= C_t^{-1} \frac{p(x_{1:t}, y_{1:t})}{q(x_{1:t} \mid y_{1:t})} q(x_{1:t} \mid y_{1:t}) \\ &= C_t^{-1} \frac{g(y_t \mid x_t) f(x_t \mid x_{t-1})}{q_t(x_t \mid y_{1:t}, x_{1:t-1})} \frac{p(x_{1:t-1}, y_{1:t-1})}{q(x_{1:t-1} \mid y_{1:t-1})} q_t(x_t \mid y_{1:t}, x_{1:t-1}) q(x_{1:t-1} \mid y_{1:t-1}). \end{aligned}$$

This decomposition suggests the subsequent algorithm. Define $w^i_{a:b} = p(x^i_{a:b}, y_{a:b})/q(x^i_{a:b} \mid y_{a:b})$ and admit a couple of conventions: if $a > b$, then $w^i_{a:b} = 1$ and $q(x^i_{a:b} \mid y_{a:b}) = 1$. Superscripts refer to the particle index, not time. Capitalizations emphasize which quantities are random. For clarity, redundant subscripts may be dropped by setting $w^i_{a:a} := w^i_a$. One can see that $w^i_{1:t} = p(x^i_{1:t}, y_{1:t})/q(x^i_{1:t} \mid y_{1:t})$ is tractable. To go from time $t-1$ to time t, you first sample from $q_t(x_t \mid x_{1:t-1}, y_{1:t})$ and append this to the particle's path, then update the old path's unnormalized weights $w^i_{1:t-1} = p(x^i_{1:t-1}, y_{1:t-1})/q(x^i_{1:t-1} \mid y_{1:t-1})$ by multiplying them by a quantity consisting of evaluations of the model's densities and the proposal density, $g(y_t \mid x^i_t) f(x^i_t \mid x^i_{t-1})/q_t(x^i_t \mid y_{1:t}, x^i_{1:t-1})$. These are evaluated using the newest samples, the most recent portion of your pre-existing particles, and the newest data point. Here is the

[1]This conditional density notation reflects the set of information that it is *possible* to condition on. If the state space model is true, then some of the random variables conditioned on are redundant due to conditional independence.

algorithm made explicit. Note that the outer loop over time is not pictured.

Algorithm 1 SIS

```
procedure SIS($g_t, f_t, q_t$)
    if $t$ equals 1 then
        for $i = 1, \ldots, N$ do
            samples[1,i] $\leftarrow X_1^i \sim q_1(x_1 \mid y_1)$
            uNrmWts[i] $\leftarrow w_1^i = \frac{g(y_1|X_1^i) f(X_1^i)}{q_1(X_1^i|y_1)}$
        end for
    else
        for $i = 1, \ldots N$ do
            samples[t,i] $\leftarrow X_t^i \sim q_t(x_t \mid x_{t-1}^i, y_t)$
            uNrmWts[i] $*= \frac{g(y_t|X_t^i) f(X_t^i|x_{t-1}^i)}{q_t(X_t^i|y_t, x_{t-1}^i)}$
        end for
    end if
end procedure
```

Unless portions of the paths are discarded, this algorithm will yield samples from $q(x_{1:t} \mid y_{1:t})$ at any time point t. With each particle path sample $x_{1:t}^i$, there are the associated unnormalized scalar weights $w_{1:t}^i$, and the normalized weights $\tilde{w}_{1:t}^i = w_{1:t}^i / \sum_{j=1}^N w_{1:t}^j$. Later, these may be written in uppercase if the randomness of the quantity is being emphasized.

An estimator of the general smoothed functional, $E[h(x_{1:t}) \mid y_{1:t}]$, will require the normalized weights. It will be computed as the following weighted average:

$$\hat{E}[h(x_{1:t}) \mid y_{1:t}] = \sum_i \tilde{w}_{1:t}^i h(x_{1:t}^i). \tag{2.9}$$

This estimator is biased, yet consistent. The estimator of the likelihood, on the other hand, is consistent and unbiased:

$$\widehat{p}(y_{1:t}) = N^{-1} \sum_{i=1}^N w_{1:t}^i. \tag{2.10}$$

Unfortunately, the variance of the SIS estimator (2.9) increases very quickly in t. How quickly the increase depends on the choice of proposal distributions and the model being studied. How should the sequence of q_ts be chosen? In the particle filtering literature, generally the minimization of the conditional variance of the scalar path weights is discussed, $\text{Var}(w_{1:t}^i \mid w_{1:t-1}^i)$, instead of a particular smoothed functional's variance or MSE[2]. In this way, results about how "good" a proposal distribution is will not be contingent on any particular quantity of interest. This was justified by [Liu, 1996]. He approximates the MSE of a general smoothed functional estimator using the delta method, and what results is an expression that is only a function of the weights' variance. This will only depend on the model and the proposal distribution. With this criterion, the optimal proposal distribution would be

$$q_t^{\text{opt}}(x_t \mid x_{1:t-1}, y_{1:t}) = p(x_t \mid y_t, x_{t-1}) \propto g(y_t \mid x_t) f(x_t \mid x_{t-1}). \tag{2.11}$$

On account of the conditional independence assumption of any state space model, this is what $p(x_t|x_{1:t-1}, y_{1:t})$ simplifies to if the model is true. If one is able to sample from this distribution, then one is able to sample from the smoothing distribution exactly. The weight update is nonrandom because it is constant, and this means the conditional variance will be zero. However, in the usual cases where it is possible to sample from this distribution, sampling is unnecessary because one can compute the distribution exactly (e.g. Kalman filter, HMMs). In the case of models that require particle filtering, (2.11) can only serve as a guideline for a proposal distribution.

If care is not taken in choosing the proposal, paths will be suggested that do not tightly hug the mode of $p(x_{1:t} \mid y_{1:t})$. Errors can build up over time as the latest samples drift farther away from where they should be. Thinking of the paths like this provides an intuitive understanding of why this happens, but it also can be looked at from the point of view of the scalar path weights. As t progresses, more of the weights get pulled closer to zero, spreading out the distribution and enlarging the variance. From this point of view, the problem is called *weight degeneracy* [Cappé et al., 2005] (chapter 7.3).

[2]This is equivalent to minimizing any of the final weights, $w^i_{1:t}$, or minimizing the likelihood estimate, $\hat{p}(y_{1:t})$.

Sequential Important Sampling with Resampling (SISR)

One partial solution to this problem is called resampling. [Gordon et al., 1993] cites [Smith and Gelfand, 1992] to justify their use of this technique. At any time in the generic SIS algorithm, one has weighted samples $\{(\tilde{w}^i_{1:t}, x^i_{1:t})\}_{i=1}^N$. Resampling consists of revising these by discarding paths with low weights, and duplicating paths with large weights. This "resetting" comes with the hope that subsequent draws will have less overall variance. The simplest method to achieve this consists of drawing revised samples' counts from a multinomial distribution, although there are other methods that are more efficient [Douc, 2005].

A draw consists of a path index number, so denoting the number of times path i is drawn by M_i, then $(M_1, \ldots, M_N) \sim \text{Multinomial}[(\tilde{w}^1_{1:t}, \ldots, \tilde{w}^N_{1:t}), N]$. Time t Monte Carlo variance is added with this resampling, but overall path weight variance is sometimes held to be low. The algorithm is suggested by the following factorization:

$$\begin{aligned} p(x_{1:t} \mid y_{1:t}) &= C^{-1}_{r+1:t} p(x_{r+1:t}, y_{r+1:t} \mid x_r) p(x_{1:r} \mid y_{1:r}) \\ &= C^{-1}_{r+1:t} \frac{p(x_{r+1:t}, y_{r+1:t} \mid x_r)}{q_{r+1:t}(x_{r+1:t} \mid y_{r+1:t}, x_r)} q_{r+1:t}(x_{r+1:t} \mid y_{r+1:t}, x_r) p(x_{1:r} \mid y_{1:r}) \\ &= C^{-1}_{r+1:t} \frac{g(y_t \mid x_t) f(x_t \mid x_{t-1})}{q_t(x_t \mid x_{t-1}, y_t)} \frac{p(x_{r+1:t-1}, y_{r+1:t-1} \mid x_r)}{q(x_{r+1:t-1} \mid y_{r+1:t-1}, x_r)} \times \\ &\qquad q_t(x_t \mid x_{t-1}, y_t) q(x_{r+1:t-1} \mid y_{r+1:t-1}, x_r) p(x_{1:r} \mid y_{1:r}) \end{aligned}$$

where r is the last time resampling was performed. This suggests that after resampling, the paths follow the posterior distribution exactly with uniform weights. Keep in mind by the aforementioned conventions, if $r + 1 > t - 1$, then the last fraction disappears, as well as $q(x_{r+1:t-1} \mid y_{r+1:t-1}, x_r)$. Many papers omit this description by assuming that resampling is performed at every time point. That convention is not followed here. Below is the algorithm made explicit.

Algorithm 2 SISR

```
procedure SISR($g_t, f_t, q_t$)
    if $t$ equals 1 then
        for $i = 1, \ldots, N$ do
            samples[1,i] ← $X_1^i \sim q_1(x_1 \mid y_1)$
            uNrmWts[i] ← $\frac{g(y_1|X_1^i)f(X_1^i)}{q_1(X_1^i|y_1)}$
        end for
        lastLogCondLike ← $\log\left[\hat{p}(y_1)\right]$
        if resampling criterion true then
            resample(samples, uNrmWts)                    ▷ sets weights to uniform
        end if
    else                                                  ▷ $t > 1$
        for $i = 1, \ldots N$ do
            samples[t,i] ← $X_t^i \sim q_t(x_t \mid x_{t-1}, y_t)$
            uNrmWts[i] $*=$ $\frac{g(y_t|X_t^i)f(X_t^i|x_{t-1}^i)}{q_t(X_t^i|y_t,x_{t-1}^i)}$
        end for
        lastLogCondLike ← $\log\left[\hat{p}(y_t \mid y_{1:t-1})\right]$
        if resampling criterion satisfied then
            resample(samples, uNrmWts)                    ▷ sets weights to uniform
        end if
    end if
end procedure
```

The conditional likelihood formulas are:[3]

$$\hat{p}(y_1) = \frac{1}{N}\sum_i W_1^i$$

$$\hat{p}(y_t \mid y_{1:t-1}) = \sum_i \widetilde{W}^i_{r+1:t-1} \frac{g(y_t \mid X_t^i) f(X_t^i \mid x_{t-1}^i)}{q_t(X_t^i \mid y_t, x_{t-1}^i)}. \tag{2.12}$$

Resampling is not without other problems as well. One downside is a problem called "sample impoverishment," and this occurs because resampling successfully adjusts the weights, but damages the paths. Whenever resampling is performed, path portions that represent early stages of the chain are probabilistically discarded. Once they are discarded, they can never return, because samples are only appended to one side of the stored particles. For this reason, the particle filter is unsuitable to esimate things of, say, the form $E[h(x_1) \mid y_{1:t}]$, if $t \gg 1$. When particle filtering is being used for obtaining the forecasting density in a financial application, this usually does not matter. When it is not, resampling should not be performed too often.

Below is a demonstration of the weight degeneracy/sample impoverishment trade-off (2-1). The image on the left shows samples from a SISR algorithm with resampling conducted at each time step. On the right, resampling is never performed. The data

[3]This formula is generally not written in papers because we do not assume resampling is being conducted at every time step, which is a simplification. Despite being slightly more complicated, understanding it this way allows for easier programming of the algorithm.

are simulated from the univariate stochastic volatility model of [Taylor, 1982], and the exact parameters are assumed to be known by the user of the particle filter. This model's state process is a univariate autoregressive process of the first order. It represents the returns' "log volatility," or more exactly, the logarithm of a factor of the variance of a return series. For each time t:

$$y_t \mid x_t \sim \text{Normal}(0, \beta^2 \exp(x_t)) \tag{2.13}$$

$$x_t \mid x_{t-1} \sim \text{Normal}(\alpha x_{t-1}, \sigma_x^2), \tag{2.14}$$

and the first state random variable is assumed to be distributed according to the state chain's stationary distribution, $\text{Normal}(0, \sigma_x^2/(1-\alpha^2))$.

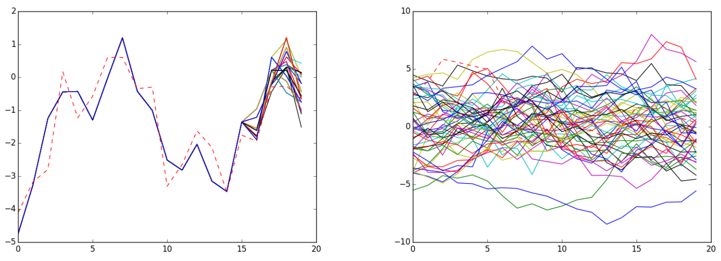


Figure 2-1: *Samples of $\{p(x_t|y_{1:20})\}_{t=1}^{20}$ from a basic bootstrap filter. On the left resampling is performed at every time step. On the right, resampling is never peformed.*

One difficulty with resampling is that it complicates asymptotic results; however, solutions to this problem exist. Going from the SIS algorithm to the SISR algorithm, the paths $\{x_{1:t}^i\}_{i=1}^N$ are no longer sampled independently. Instead, these paths "interact" via the resampling scheme. The first paper to prove convergence of estimators from Sequential Monte Carlo with resampling schemes was [Moral, 1996], but this is also discussed in [Crisan and Doucet, 2002] and [Chopin, 2004]. The monograph [Cappé et al., 2005] describes proofs of a law of large numbers and a central limit theorem in some detail, describing the extension from the case of the nonsequential algorithm of [Grenander et al., 1991].

The Auxiliary Particle Filter (APF)

A different partial solution, usually used in conjunction with resampling, is to take better care in picking a proposal distribution. A better proposal will mitigate the problem of weight degeneracy, just as resampling does. In the original bootstrap filter [Gordon et al., 1993], the model's state transition $f(x_t \mid x_{t-1})$ density is used as a proposal. This seems natural at first, because the model assumes that that is how the state evolves unconditionally of the observed data. It even simplifies the weight update routine, and is the only possible choice if evaluating the transition density is impossible. However, it is not the optimal proposal (2.11); there is nothing pulling the samples close to the filtering distribution $p(x_t \mid y_{1:t})$ and thereby keeping the variance of the weights low.

In the language of [Pitt and Shephard, 1999a], the naive proposal distribution is not "adapted" to the most recent data point. They propose a generic scheme for sampling that uses this information called "auxiliary particle filtering" (APF). The APF ameliorates this problem by automatically choosing a proposal distribution that more closely resembles the optimal one. The scheme makes use of an auxiliary random variable at each time point t, call it $k_{t-1} \in \{1, \ldots, N\}$. Its probability mass function is given by the stored path weights $\{\tilde{w}^1_{1:t-1}, \ldots, \tilde{w}^N_{1:t-1}\}$. This variable represents which recent sample should be chosen to propogate forward. The following procedure is based on the observation that

$$
\begin{aligned}
p(x_{1:t}, k_{t-1} \mid y_{1:t}) &= C^{-1}_{r+1:t} \frac{p(x_{r+1:t}, k_{t-1}, y_{r+1:t} \mid x_r)}{q(x_{r+1:t}, k_{t-1} \mid y_{1:t})} q(x_{r+1:t}, k_{t-1} \mid y_{1:t}) p(x_{1:r} \mid y_{1:r}) \\
&= C^{-1}_{r+1:t} \frac{g(y_t \mid x_t) f(x_t \mid x_{t-1}, k_{t-1}) p(k_{t-1} \mid x_{1:t-1}, y_{1:t-1})}{q_t(x_t, k_{t-1} \mid y_t, x_{t-1})} \frac{p(x_{r+1:t-1}, y_{r+:t-1} \mid x_r)}{q(x_{r+1:t-1} \mid y_{r+1:t-1})} \times \\
&\qquad q_t(x_t, k_{t-1} \mid y_{1:t}, x_{1:t-1}) q(x_{r+1:t-1} \mid y_{r+1:t-1}) p(x_{1:r} \mid y_{1:r}),
\end{aligned}
$$

where adjustments are made to the underlying state space model because the random index is part of the state process now.

To understand the APF, one must understand the choice of q_t. The optimal proposal, $p(x_t, k_{t-1} \mid y_{1:t})$, is proportional to $f(x_t \mid x_{t-1}, k_{t-1}) p(y_t \mid x_{t-1}, k_{t-1}) p(k_{t-1} \mid$

$x_{1:t-1}, y_{1:t-1})$, but generally $p(y_t \mid x_{t-1}, k_{t-1})$ is not available in closed-form. The following factorization suggests an approximation based on a user-chosen function $\mu_t(\cdot)$:

$$\begin{aligned}
p(x_t, k_{t-1} \mid y_{1:t}, x_{1:t-1}) &\propto f(x_t \mid x_{t-1}, k_{t-1}) p(y_t \mid x_{t-1}, k_{t-1}) p(k_{t-1} \mid x_{1:t-1}, y_{1:t-1}) \\
&= f(x_t \mid x_{t-1}, k_{t-1}) \left[\int g(y_t \mid x'_t) f(x'_t \mid x_{t-1}, k_{t-1}) dx'_t \right] p(k_{t-1} \mid x_{1:t-1}, y_{1:t-1}) \\
&\approx f(x_t \mid x_{t-1}, k_{t-1}) g(y_t \mid \mu_t[x_{t-1}, k_{t-1}]) p(k_{t-1} \mid x_{1:t-1}, y_{1:t-1}).
\end{aligned}$$

So, the normalized proposal distribution becomes

$$q_t^{\text{apf}}(x_t, k_{t-1} \mid y_{1:t}, x_{1:t-1}) = \frac{f(x_t \mid x_{t-1}, k_{t-1}) g(y_t \mid \mu_t[x_{t-1}, k_{t-1}]) p(k_{t-1} \mid x_{1:t-1}, y_{1:t-1})}{\sum_{k'_{t-1}} g(y_t \mid \mu_t[x_{t-1}, k'_{t-1}]) p(k'_{t-1} \mid x_{1:t-1}, y_{1:t-1})}.$$

The most common choice for μ_t is the mode of $f(x_t \mid x_{t-1})$, but many other choices have been studied [Whiteley and Johansen, 2011]. In words, the algorithm is as follows: all of the particles are weighted by the first stage weights, which are proportional to $g(y_t \mid \mu_t[x_{t-1}, k_{t-1}]) p(k_{t-1} \mid x_{1:t-1}, y_{1:t-1})$. Then, based on these weights, particle indices k_{t-1} are sampled[4], and the samples selected by these indices are then propogated further via the state transition density. The algorithm is as follows.

The formulas for the conditional likelihood approximations are the same as the SISR algorithms if one makes the appropriate substitution for q_t.

The Rao-Blackwellized Particle Filter (RBPF)

Another technique to reduce variance is called "Rao-Blackwellization" [Liu et al., 2001], [Chen and Liu, 2000], [Andrieu and Doucet, 2002], however this is only useful for models of a certain form. Let $x_t = (x'_{1,t}, x'_{2,t})'$, where $x_{1,t}$ and $x_{2,t}$ are subvectors for any time's state. The necessary form for a model of interest involves being able

[4]Confusingly, the literature on particle filtering will sometimes refer to the first stage of the APF proposal as resampling, and sometimes explain the general particle filter as having the true resampling step as coming first, instead of last. This document does not follow either of these conventions.

Algorithm 3 APF

```
procedure APF($g_t, f_t, \mu_t$)
    if $t$ equals 1 then                                                    ▷ same as SISR
        for $i = 1, \ldots, N$ do
            samples[1,i] $\leftarrow X_1^i \sim q_1(x_1 \mid y_1)$
            uNrmWts[i] $\leftarrow \frac{g(y_1|X_1^i)f(X_1^i)}{q_1(X_1^i|y_1)}$
        end for
        lastLogCondLike $\leftarrow \log\left[\hat{p}(y_1)\right]$
        if resampling criterion true then
            resample(samples, uNrmWts)
        end if
    else                                                                    ▷ $t > 1$
        for $i = 1, \ldots N$ do
            firstStageUnNormalizedWeights[i] $\leftarrow g(y_t \mid \mu_t[x_{t-1}, k_{t-1}^i])p(k_{t-1}^i \mid x_{1:t-1}, y_{1:t-1})$
        end for
        kTMinusOneArray = sample(1:N, firstStageUnNormalizedWeights)
        for $i = 1, \ldots N$ do
            samples[t,i] $\leftarrow X_t^i \sim f_t(x_t \mid x_{t-1}, K_{t-1}^i)$
            uNrmWts[i] $* = \frac{g(y_t|X_t^i)}{g(y_t|\mu_t[x_{t-1}, K_{t-1}^i])} \sum_j g(y_t \mid \mu_t[x_{t-1}, K_{t-1}^j])p(K_{t-1}^j \mid x_{1:t-1}, y_{1:t-1})$
        end for
        lastLogCondLike $\leftarrow \log\left[\hat{p}(y_t|y_{1:t-1})\right]$
        if resampling criterion satisfied then
            resample(samples, uNrmWts)
        end if
    end if
end procedure
```

to decompose the smoothing density as

$$p(x_{1:t} \mid y_{1:t}) = p(x_{1,1:t} \mid x_{2,1:t}, y_{1:t})p(x_{2,1:t} \mid y_{1:t}). \tag{2.15}$$

If $p(x_{1,1:t} \mid x_{2,1:t}, y_{1:t})$ is tractable, and $p(x_{2,1:t} \mid y_{1:t})$ is not, then samples are only needed for the latter. The particles are comprised of $\{w_{1:t}^i, x_{1,1:t}^i, x_{2,1:t}^i\}$, where the $x_{1,1:t}^i$ are nonrandom summaries of the distribution $p(x_{1,1:t} \mid x_{2,1:t}^i, y_{1:t})$, and the $x_{2,1:t}^i$ are "ordinary" samples in the sense they represent realizations of the random variable or vector. For example, if the first distribution on the right hand side in (2.15) is multivariate normal, then each $x_{1,1:t}^i$ is a sequence of mean vector/covariance matrix pairs coupled with both a corresponding $x_{2,1:t}^i$ sample, and a weight $w_{1:t}^i$. Consider as another example a model where the tractable distribution is a finite state space Markov chain. In that case, each particle element $x_{1,1:t}^i$ is a probability vector.

The primary decomposition expressing the workings of this algorithm involves writing the right hand side of (2.15) as

$$C_{r+1:t}^{-1}p(x_{1,1:t} \mid x_{2,1:t}, y_{1:t})a_{t-1:t}w_{r+1:t-1}\tilde{p}(x_{2,1:t} \mid y_{1:t}) \tag{2.16}$$

where $a_{t-1:t} = f(x_{2,t} \mid x_{2,t-1})p(y_t \mid y_{1:t-1}, x_{2,1:t})/q_t(x_{2,t} \mid x_{2,t-1}, y_t)$ is the unnormalized weight adjustment, $w_{r+1:t-1} = p(x_{2,r+1:t-1}, y_{r+1:t-1} \mid x_{2,r})/q(x_{2,r+1:t-1} \mid y_{r+1:t-1}, x_{2,r})$ is the previous unnormalized weight, and

$$\tilde{p}(x_{2,1:t} \mid y_{1:t}) = q_t(x_{2,t} \mid x_{2,t-1}, y_t)q(x_{2,r+1:t-1} \mid y_{r+1:t-1}, x_{2,r})p(x_{2,1:r} \mid y_{1:r}) \tag{2.17}$$

denotes the current distribution of the samples for $x_{2,1:t}$, assuming resampling was last performed at time r.

When conditioning on the most recent particles, one is able to use his inner[5] closed form model to accomplish the following update:

$$p(x_{1,1:t} \mid x_{2,1:t}, y_{1:t}) = \frac{p(y_t \mid x_{1,t}, x_{2,1:t})p(x_{1,t} \mid x_{1,t-1}, x_{2,1:t})p(x_{1,1:t-1} \mid y_{1:t-1}, x_{2,1:t})}{p(y_t \mid x_{2,1:t}, y_{1:t-1})} \tag{2.18}$$

Also, the first time's filtering distribution is:

$$p(x_{1,1}, x_{2,1} \mid y_1) = C_1^{-1} p(x_{1,1} \mid x_{2,1}, y_1)\frac{p(y_1 \mid x_{2,1})f(x_{2,1})}{q_1(x_{2,1} \mid y_1)}q_1(x_{2,1} \mid y_1) \tag{2.19}$$

This algorithm (4) is written below explicitly. Note that $p(y_1 \mid x_{2,1})$ and each $p(y_t \mid x_{2,1:t}, y_{1:t-1})$ are available in closed form.

The likelihood estimates are similar to the ones before:

$$\hat{p}(y_t \mid y_{1:t-1}) = \sum_{i=1}^{N} \tilde{W}^i_{r+1:t-1} \frac{f(X^i_{2,t} \mid x^i_{2,t-1})p(y_t \mid y_{1:t-1}, X^i_{2,1:t})}{q_t(X^i_{2,t} \mid x^i_{2,t-1}, y_t)}, \tag{2.20}$$

and

$$\hat{p}(y_1) = \frac{1}{N}\sum_{i=1}^{N} \frac{p(y_1 \mid X^i_{2,1})f(X^i_{2,1})}{q_1(X^i_{2,1}|y_1)}. \tag{2.21}$$

The formulas for the approximations to the expectations of functions of the state or based on the Law of Total Expectation. The quantity $E[h(x_{1t}, x_{2t}) \mid y_{1:t}] =$

[5]It is recommended to program a RBPF in an object-oriented manner. An instance of a Kalman Filter or HMM class can be held and manipulated inside a RBPF class. To obtain this value, each particle can have its, say, Kalman Filter class call the method that returns the most recent conditional likelihood.

Algorithm 4 RBPF

```
procedure RBPF($p_t, f_t$)
    if $t$ equals 1 then
        for $i = 1, \ldots, N$ do
            particles.samples[1,i] $\leftarrow X_{2,1}^i \sim q_1(x_{2,1}|y_1)$
            particles.innerMods[i].prior $= p(x_{1,1}^i \mid X_{2,1}^i)$
            particles.innerMods[i].posterior $= p(x_{1,1}^i \mid X_{2,1}^i, y_1)$    ▷ This step and the last step are the first iteration of the closed-form filter.
            tempInnerCondLike $= p(y_1 \mid X_{2,1}^i) \leftarrow$ particles.innerMod[i].getCondLike()    ▷ New likelihood is by-product of updated posterior.
            particles.uNrmWts[i] $\leftarrow \frac{p(y_1|X_{2,1}^i)f(X_{2,1}^i)}{q_1(X_{2,1}^i|y_1)}$
        end for
        lastLogCondLike $\leftarrow \log\left[\hat{p}(y_1)\right]$
        if resampling criterion true then
            resample(particles)
        end if
    else    ▷ $t > 1$
        for $i = 1, \ldots N$ do
            particles.samples[t,i] $\leftarrow X_{2,t}^i \sim q_t(x_{2,t} \mid x_{2,t-1}^i, y_t)$
            particles.innerMods[i].prior.update()
            particles.innerMods[i].posterior.update($y_t$)
            tempInnerCondLike $= p(y_t \mid y_{1:t-1}, X_{2,1:t}^i) \leftarrow$ particles.innerMod[i].getCondLike()
            particles.uNrmWts[i] $*= \frac{f(X_{2,t}^i|x_{2,t-1}^i)p(y_t|y_{1:t-1},X_{2,1:t}^i)}{q_t(X_{2,t}^i|x_{2,t-1}^i,y_t)}$
        end for
        lastLogCondLike $\leftarrow \log\left[\hat{p}(y_t \mid y_{1:t-1})\right]$
        if resampling criterion satisfied then
            resample(particles)
        end if
    end if
end procedure
```

$E(E[h(x_{1t}, x_{2t}) \mid x_{2t}, y_{1:t}] \mid y_{1:t})$ is approximately

$$\sum_{i=1}^{N} \tilde{W}_{r+1:t-1}^{i} E[h(x_{1t}, x_{2t}^i) \mid x_{2t}^i, y_{1:t}]. \tag{2.22}$$

Note that an expression for the inner conditional expectation will often be available in closed form.

The Rao-Blackwellized Auxiliary Particle Filter (RBAPF)

Indeed, the ideas of the last two particle filters can be combined into one algorithm; it is possible to construct a "Rao-Blackwellized Auxiliary Particle Filter," or a "Marginal Rao-Blackwellized Particle Filter" [Nordh, 2014], [Fritsche et al., 2009], [Jianjuna et al., 2007]. This algorithm utilizes (2.15) just as before, however, it makes use of an auxiliary random variable, just as the APF does. The samples target the second factor on the right hand side with an importance distribution motivated by the reasoning described in the description of the APF.

This following formula is analagous to equation (2.16):

$$\begin{aligned}&p(x_{1,1:t} \mid x_{2,1:t}, k_{t-1}, y_{1:t})p(x_{2,1:t}, k_{t-1} \mid y_{1:t}) \\ &\qquad = C_{r+1:t}^{-1} p(x_{1,1:t} \mid x_{2,1:t}, k_{t-1}, y_{1:t}) a_{t-1:t} w_{r+1:t-1} \tilde{p}(x_{2,1:t} \mid y_{1:t})\end{aligned}$$

where $a_{t-1:t} = f(x_{2,t} \mid x_{2,t-1}, k_{t-1}) p(y_t \mid y_{1:t-1}, x_{2,1:t}, k_{t-1}) / q_t(x_{2,t}, k_{t-1} \mid x_{2,t-1}, y_t)$ is the unnormalized weight adjustment, $w_{r+1:t-1} = p(x_{2,r+1:t-1}, y_{r+1:t-1}, k_{t-1} \mid x_{2,r}) / q(x_{2,r+1:t-1} \mid y_{r+1:t-1}, x_{2,r})$ is the previous unnormalized weight, and

$$\tilde{p}(x_{2,1:t} \mid y_{1:t}) = q_t(x_{2,t}, k_{t-1} \mid x_{2,t-1}, y_t) q(x_{2,r+1:t-1} \mid y_{r+1:t-1}, x_{2,r}) p(x_{2,1:r} \mid y_{1:r})$$

denotes the current distribution of the samples for $x_{2,t-1}$, assuming resampling was last performed at time r. This decomposition again shows the multiplicative weight update, multiplying $w_{r+1:t-1}$ by $a_{t-1:t}$. It also shows the two places where the inner closed-form model is being used: the successive updates of $p(x_{1,1:t} \mid x_{2,1:t}, k_{t-1}, y_{1:t})$, and nonrandom evaluations of $p(y_t \mid y_{1:t-1}, x_{2,1:t}, k_{t-1})$. This algorithm shares (2.19) with the RBPF as well. Regarding the inner model updating, this model assumes a slightly modified version of (2.18):

$$\begin{aligned}&p(x_{1,1:t} \mid x_{2,1:t}, k_{t-1}, y_{1:t}) \\ &\qquad = \frac{p(y_t \mid x_{1,t}, x_{2,1:t}, k_{t-1}) p(x_{1,t} \mid x_{1,t-1}, k_{t-1}, x_{2,1:t}) p(x_{1,1:t-1} \mid y_{1:t-1}, k_{t-1}, x_{2,1:t})}{p(y_t \mid x_{2,1:t}, k_{t-1}, y_{1:t-1})}.\end{aligned}$$

The proposal distribution at each time point $q_t(x_{2,t}, k_{t-1} \mid x_{2,t-1}, y_t)$ is chosen to mimic the optimal proposal $p(x_{2,t}, k_{t-1} \mid x_{2,t-1}, y_{1:t})$:

$$\begin{aligned}p(x_{2,t}, k_{t-1} \mid x_{2,t-1}, y_{1:t}) &\propto f(x_{2,t} \mid x_{2,t-1}, k_{t-1}) p(y_t \mid x_{2,t-1}, k_{t-1}) p(k_{t-1} \mid x_{2,t-1}, y_{1:t-1}) \\ &= f(x_{2,t} \mid x_{2,t-1}, k_{t-1}) \left[\int g(y_t \mid x'_{1,t}, x'_{2,t}) p(x'_{1,t}, x'_{2,t} \mid x_{2,t-1}, k_{t-1}) dx'_t \right] \times \\ &\qquad p(k_{t-1} \mid x_{2,t-1}, y_{1:t-1}) \\ &\approx f(x_{2,t} \mid x_{2,t-1}, k_{t-1}) g(y_t \mid \mu_t[x_{t-1}, k_{t-1}]) p(k_{t-1} \mid x_{1:t-1}, y_{1:t-1}).\end{aligned}$$

The rest of the algorithmic details follow straightforwardly.

A Note on Computation

A primary computational difficulty with implementing these particle filters is the problem of *arithmetic underflow.* Given a base b and a precision p, a floating point number consists of three numbers: a significand, an exponent and a sign bit. Every floating point number stored inside a computer will be represented as a signicand times the base raised to the exponent, all multiplied by a power of -1. This significand is written in base b, has p digits, and its leading digit is nonzero (except in the case of subnormal numbers, which are not discussed here). Typically, the base is two, although other bases such as 10 and 16 are frequently used as well.

In the IEEE Standard for Floating Point Arithmetic (IEEE 754) a double precision floating point number consists of one bit to store the sign of the number (positive or negative), 11 bits for the signed exponent integer e (ranging from -1024 to 1023), and 52 bits for the significand $(1.b_{51}b_{50}\cdots b_0)_2$. In base ten, the significand provides approximately 15-17 significant digits because $2^{-53} \approx 1.11 \times 10^{-16}$. The formula for such a double precision floating point number is

$$(-1)^{\text{sign}}(1.b_{51}b_{50}\cdots b_0)_2 \times 2^e. \tag{2.23}$$

There are two ways that numerical error can enter into a program. Either a real number is not exactly representable by a floating point number, or the real number exceeds a lower or upper bound. Frequently, both problems arise in programs implementing particle filtering algorithms; however, this document primarily concerns itself with numerical underflow, which is an example of the second situation. This occurs when the target (positive) real number is below the smallest positive representable number. In this case, the ideal exponent of this number is smaller than the minimum that is allowed by the computer's floating point representation. In particle filtering, this problem often appears in calculations using the particle weights.

How often this happens depends on the specifics of the model and the algorithm under consideration. For example, many models of interest in this document consider Gaussian densities. In this case, evaluating these densities consists of exponentiating negative quadratic forms. If this negative power is smaller than -1024, the most extreme allowed exponent, then the evaluation will underflow, and the floating point number will be rounded to 0. This can cause a wide variety of numerical problems. Theoretically, a division-by-0 error might be guaranteed to be impossible; however, numerically, it may be likely, especially if preventative measures are not taken. Consider another example: a particle filter that resamples infrequently and experiences weight degeneracy (see 2.3), or possesses a large number of particles. These will increase its chances of getting false 0s for unnormalized weights. After normalizing these, the probabilities could be very far off from the true theoretical ones.

These 0 or near-0 unnormalized weights cause errors in the calculations of the approximations to expectations, likelihoods, and distribution functions. To circumvent these problems, one might consider keeping track of unnormalized log weights. After examining expression (2.23), it can be seen that using the log of the unnormalized weights and evaluating log densities is much safer. This is a good start. However, exponentations are not always unavoidable. When they are necessary, exponentiating these small (negative numbers that are large in absolute value) can potentially cause underflow. For example, if one has unnormalized log weights $\{z_i\}$, then the expectations must be approximated with the formula $\sum_i h(x_i)e^{z_i} / \sum_j e^{z_j}$. Underflow from these exponentials will incorrectly cause some samples to have 0 weight, and so will harm the output of these calculations. One way to address this issue is to divide the numerator and denominator of this fraction by some well-chosen number. It's easy to see the expectation formula is algebraically equivalent to

$$\frac{\sum_i h(x_i)e^{z_i - m}}{\sum_j e^{z_j - m}}, \tag{2.24}$$

where m is usually chosen to be the maximum of the log of the unnormalized weights. If m is a negative number, all of the unnormalized weights are negative, so it will shift all z_i upward and fewer numbers will underflow. If m is positive, then not all of the unnormalized weights are at risk for underflow, and shifting all of them downward will ensure the extremely negative numbers will round to zero after being exponentiated.

Likelihood calculations are also possibly affected by issues of underflow. A particle filter's approximations to each observation's conditional likelihood involves a computation with many (log)weights. It is at risk of causing underflow because exponentiation of the log weights is unavoidable. For example, if one is using a SISR algorithm, the log likelihood update formula for $t \geq 2$ is given by (2.12). The following formula, which is equivalent algebraically, is less prone to numerical errors:

$$\log \hat{p}(y_t|y_{1:t-1}) = m_1 - m_2 - \log \sum_i \exp\left\{\log W^i_{r+1:t-1} - m_2\right\} + \\ \log \sum_i \exp\left\{\log W^i_{r+1:t-1} + \log g(y_t|X^i_t) + \log f(X^i_t|x^i_{t-1}) - \log q_t(X^i_t|y_t, x^i_{t-1}) - m_1\right\} \tag{2.25}$$

where $m_1 = \max_i\{\log W^i_{r+1:t-1} + \log g(y_t|X^i_t) + \log f(X^i_t|x^i_{t-1}) - \log q_t(X^i_t|y_t, x^i_{t-1})\}$ and $m_2 = \max_i\{\log W^i_{r+1:t-1}\}$. If these summands are at risk of being too large, subtracting their maximum diminishes the risk of overflow into positive numbers. If these summands are at risk of being too small (large in absolute value, but negative), then subtracting the maximum from each summand has the same benefits that were mentioned above.

Adjusting formulas like these in this manner is applying what isknown as the "log-sum-exp trick." As another illustrative example, consider the formula for an Auxiliary Particle Filter's log conditional likelihood update. It applies the same trick, only in more places, subtracting a maximum before exponentiating:

$$\begin{aligned}\log \hat{p}(y_t|y_{1:t-1}) = m_1 + \log \sum_i \exp\left[\log W^i_{r+1:t-1} + \log g(y_t|X^i_t) - \log g(y_t|\mu_t[x_{t-1}, K^i_{t-1}]) - m_1\right] + \\ m_2 + \log \sum_j \exp\left[\log g(y_t|\mu_t[x_{t-1}, K^j_{t-1}]) + \log p(K^j_{t-1}|x_{1:t-1}, y_{1:t-1}) - m_2\right] - \\ m_3 - \log \sum_i \exp\left[W^i_{r+1:t-1} - m_3\right].\end{aligned} \tag{2.26}$$

Despite being slightly more computationally demanding, these two tricks are applied whenever possible in the software used for this document.

Chapter 3

Parameter Estimation with Markov Chain Monte Carlo Techniques

General MCMC

The previous chapter assumes the vector of unknown parameters $\theta \in \Theta \subset R^{d_p}$ is completely known, a situation that is unusual in practice. In order to deal with parameter uncertainty, this chapter details how to estimate this quantity in a Bayesian framework. Samples from either $p(x_{1:T}, \theta \mid y_{1:T})$ or $p(\theta \mid y_{1:T})$ are desired. Simulating indepenent samples from this distribution is difficult, so Markov chain Monte Carlo (MCMC) techniques construct a Markov chain evolving on either the state space $\mathcal{X}^T \times \Theta$ or the space Θ, which possesses the target posterior as an invariant/stationary distribution. Under certain conditions, a law of large numbers and a central limit theorem will hold, allowing one to estimate functionals of the chain by taking pseudo-time averages. Two very informative tutorials on MCMC techniques from the general point of view are [Geyer, 2005], [Roberts and Rosenthal, 2004].

Denote $\pi(dx)$ as the target distribution of the MCMC sampling scheme. As a reminder, x will no longer denote a realization of a state random variable. Rather, it will denote a realization of any Markov chain in any of the parameter spaces that were mentioned before; to avoid confusion, the state space of this chain will be denoted $\mathcal{X}$, and its σ-algebra $\mathcal{A}$. The letter "d" is used as a reminder that π is a measure.

Writing it this way follows the same convention followed by many MCMC resources.

Next let $K(x, dy) : \mathcal{X} \times \mathcal{A} \to [0, 1]$ be a transition kernel. Not all measures that are mentioned are absolutely continuous with respect to Lebesgue measure, and so do not admit densities. The Metropolis-Hastings sampler, for example, will not fit this description. Also, transition kernels are written in a left to right manner to match the notation of other MCMC tutorials. This is a break from the notation style being used to describe particle filters in the previous sections.

General MCMC algorithms will be dealt with before addressing estimation of SSMs. Several definitions are given, then two main theorems are discussed. First, the definition of a "stationary distribution" is given. The Markov chains discussed throughout this section will always be assumed to possess a stationary distribution.

Definition 1. A Markov chain with transition kernel K has *stationary* distribution π if for all $x, y \in \mathcal{X}$

$$\pi(dy) = \int_{\mathcal{X}} \pi(dx)K(x, dy). \tag{3.1}$$

In other words, applying the transition kernel does not change the marginal distribution of the chain. Practically speaking, the task for a Bayesian statistician using a MCMC technique is to design a chain that possesses the posterior distribution of interest as its stationary distribution. In this way, even though draws may be correlated with each other, individually, they are all still samples from the desired distribution.

Instead of verifying stationarity of a chain, it is often easier to verify the condition of "reversibility," which is a sufficient condition for stationarity.

Definition 2. A Markov chain with stationary distribution π is *reversible* if

$$\int_A \int_B \pi(dx)K(x, dy) = \int_B \int_A \pi(dx)K(x, dy) \tag{3.2}$$

for all $A, B \in \mathcal{A}$.

In other words, the joint distribution of the chain at two consecutive time points is "symmetric."

For a law of large numbers to hold, Markov chains need two other conditions, in addition to the possession of a stationary distribution. First, they need to be "irreducible."

Definition 3. Let ϕ be a nonzero σ-finite measure on the Markov chain's measure space $(\mathcal{X}, \mathcal{A})$. The Markov chain is called ϕ-irreducible if for every $x \in \mathcal{X}$ and every $A \in \mathcal{A}$, $\phi(A) > 0$ implies that there exists an $n \in \mathbb{N}$ such that $K^n(x, A) > 0$.

This condition guarantees that certain sets are reachable (eventually) from everywhere on the state space. There are many measures that work with this definition. In particular, ϕ may be chosen in a way that puts positive measure on very few sets. This is less informative than a "maximal" irreducibility measure. The unique maximal irreducibility measure ψ can be constructed by taking a weighted average of the chain's transition kernels; the resulting kernel is called the resolvent [Meyn and Tweedie, 2009]. This measure, ψ, gives positive measure to the maximum number of sets, and has nullsets that are essentially unreachable. When using Markov chains to sample from a stationary distribution π, π-irreducibility is of interest. Intuitively, if some set has positive measure under π, then it would be undesirable for the chain never to have any chance of getting there.

In addition to being irreducible, it is desirable that the chain is "aperiodic," or not "periodic."

Definition 4. A Markov chain with stationary distribution π is periodic with period d if there exists $d \geq 2$ disjoint subsets $\mathcal{X}_1, \ldots, \mathcal{X}_d \subset \mathcal{X}$ such that $\pi(\mathcal{X}_i) > 0$ for any i, and

$$K(x, \mathcal{X}_{i+1}) = 1$$

for any $x \in \mathcal{X}_i$, $1 \leq i < d$, along with $K(x, \mathcal{X}_1)$ for all $x \in \mathcal{X}_d$.

It is rare for MCMC chains to be aperiodic. This document favors versions of the Metropolis-Hastings algorithms, which produce chains that have positive probability of not moving over one time period. This property immediately eliminates the possibility of aperiodicity.

The following is the first theorem justifying the use of MCMC techniques for posterior simulation. Using the definitions given so far, this theorem implies that the chain is "ergodic," which is a sufficient, but not necessary, condition for a strong law of large numbers.

Theorem 1. *Let $X_1, X_2, \ldots$ be a ϕ-irreducible and aperiodic Markov equipped with a countably generated σ-field, and a stationary distribution π. Then the chain is ergodic, or in other words,*

$$\lim_{n \to \infty} \left\| P^n(x, \cdot) - \pi(\cdot) \right\| = 0 \tag{3.3}$$

for π-almost all x.

Now that a convergent estimator can be obtained by taking psuedo-time averages of a Markov chain, the next issue is estimation of an estimator's standard error. For this task, a central limit theorem is desired. Unlike central limit theorems for iid random samples, a central limit theorem for Markov chains is not guaranteed by the existence of second moments of g under π. Most central limit theorems need additional assumptions of stronger types of ergodicity.

Definition 5. A Markov chain with stationary distribution π is *uniformly ergodic* if

$$\left\| K^n(x, \cdot) - \pi(\cdot) \right\|_{TV} \leq M\rho^n, \qquad n = 1, 2, \ldots \tag{3.4}$$

for some $\rho < 1$ and $M < \infty$.

Definition 6. A Markov chain with stationary distribution π is *geometrically ergodic* if

$$\left\| K^n(x, \cdot) - \pi(\cdot) \right\|_{TV} \leq M(x)\rho^n, \qquad n = 1, 2, \ldots \tag{3.5}$$

for some $\rho < 1$ and $M(x) < \infty$ π-almost surely.

Both of these require that the transition distributions of the chain approach the stationary distribution of the chain exponentially fast. The difference between the two is that the same rate applies to any starting point x for the former, and it may vary for the latter. Here are four central limit theorems, all assuming different things about not only the rate of convergence, but also how many moments exist for the functional that is being averaged.

Theorem 2. *Let $X_1, X_2, \ldots$ be a uniformly ergodic Markov chain with stationary distribution π. Also assume that $E_\pi[g^2] < \infty$. Then*

$$\sqrt{n}\left(\sum_{i=1}^{n} \frac{g(X_i)}{n} - E_\pi[g]\right) \xrightarrow{D} Normal(0, \sigma^2) \tag{3.6}$$

for some σ^2 as $n \to \infty$.

Unfortunately, uniform ergodicity is difficult to verify for the Metropolis-Hastings algorithm.

Theorem 3. *Let $X_1, X_2, \ldots$ be a geometrically ergodic Markov chain with stationary distribution π. Also assume that for some $\delta > 0$ $E_\pi[g^{2+\delta}] < \infty$. Then*

$$\sqrt{n}\left(\sum_{i=1}^{n} \frac{g(X_i)}{n} - E_\pi[g]\right) \xrightarrow{D} Normal(0, \sigma^2) \tag{3.7}$$

for some σ^2 as $n \to \infty$.

Theorem 4. *Let $X_1, X_2, \ldots$ be a geometrically ergodic and reversible Markov chain with stationary distribution π. Also assume that $E_\pi[g^2] < \infty$. Then*

$$\sqrt{n}\left(\sum_{i=1}^{n} \frac{g(X_i)}{n} - E_\pi[g]\right) \xrightarrow{D} Normal(0, \sigma^2) \tag{3.8}$$

for some σ^2 as $n \to \infty$.

The central limit theorem of [Kipnis and Varadhan, 1986] does not require any ergodicity assumption.

Theorem 5. *Let $X_1, X_2, \ldots$ be a ϕ-irreducible, aperiodic and reversible Markov chain with stationary distribution π. Then*

$$\sqrt{n}\left(\sum_{i=1}^{n}\frac{g(X_i)}{n} - E_\pi[g]\right) \xrightarrow{D} Normal(0, \sigma^2) \tag{3.9}$$

for some σ^2 as $n \to \infty$.

Given these results, how does one design the Markov chain to sample from the posterior? The most popular choice of K is known as the Metropolis-Hastings algorithm

$$K(x, A) = \left[1 - \int_{\mathcal{Y}} q(x, y)r(x, y)dy\right] I(x, A) + \int_A q(x, y)r(x, y)dy \tag{3.10}$$

where $q(x, y)$ is a user-defined proposal density, π_u is the unnormalized stationary distribution, $I(x, A) = \delta_x(A)$ is the identity kernel and $r(x, y) = \min\left\{1, \frac{\pi_u(y)q(y,x)}{\pi_u(x)q(x,y)}\right\}$ is the Hastings' ratio. Special cases of this algorithm include Gibbs sampling, where $q(x, y) = \pi(y \mid x)$, independent Metropolis-Hastings, where $q(x, y) = q(y)$, random-walk Metropolis-Hastings, where $q(x, y) = q(x - y)$, and when q is symmetric (the Metropolis algorithm). In any of these cases, this Markov chain is reversible (and hence stationary), π-irreducible, and as was mentioned before, aperiodic.

MCMC for SSMs

After choosing a prior distribution for his parameters, $p(\theta)$, the investor is primarily interested in the posterior of the unknown parameters $p(\theta \mid y_{1:t})$. He might also be interested in the full posterior, $p(x_{1:t}, \theta \mid y_{1:t})$, particularly if he wishes to conduct a retrospective analysis of the hidden states.

In principle, MCMC algorithms will allow him to sample from either of these distributions. However, it is more common to simulate from the latter, as it is proportional to the "complete-data" likelihood (2.1). Even though it is lower-dimensional, the marginal posterior is rarely proportional to anything that can be evaluated in general SSMs; whenever samples from this distribution are required, they are usually obtained by integrating out undesired portions of samples from the higher-dimensional posterior.

Closed-Form Gibbs Sampling

Block-wise Gibbs sampling is one tool used to accomplish sampling from the full posterior, wherein draws are taken from $p(\theta \mid x_{1:t}, y_{1:t})$ and $p(x_{1:t} \mid \theta, y_{1:t})$ in an alternating fashion. Simulating from the former is relatively straightforward when using priors with a certain form. However, simulating from the smoothing distribution is only possible for a small class of SSMs. For example, with Linear-Gaussian state space models, the procedure of [Carter and Kohn, 1994] is available.

Component-wise Gibbs sampling can also be used with certain models, which alternates between sampling parameters and sampling each time's state variable. This is available, for instance, with discrete-valued HMMs [Fearnhead, 2011]. By taking advantage of conditional independence, and the finiteness of the state space, normalizing constants can be evaluated for each state component's conditional distribution. However, this is not available for the types of models considered in this paper, and it is generally desirable to block together correlated state samples to increase sampling efficiency.

Advanced Gibbs Sampling

With more complex models, Gibbs sampling is still possible. However, the conditional distributions will not always be tractable. For these particularly problematic distributions, Metropolis-Hastings samplers, or accept-reject strategies are often used.

Say, for example, component-wise Gibbs algorithms are to be used. Because every state space model makes the same assumptions about conditional independence, it is always true that

$$\begin{aligned} p(x_i \mid x_{1:i-1}, x_{i+1:T}, y_{1:T}, \theta) &\propto p(\theta) f(x_1 \mid \theta) g(y_1 \mid x_1, \theta) \prod_{t=2}^{T} f(x_t \mid x_{t-1}, \theta) g(y_t \mid x_t, \theta) \\ &\propto f(x_{i+1} \mid x_i, \theta) f(x_i \mid x_{i-1}, \theta) g(y_i \mid x_i, \theta). \end{aligned}$$

for $2 \leq i \leq n-1$. In the case where these three densities can only be evaluated, independence MH samplers can be used [Jacquier et al., 1994], [Shephard and Pitt, 1997], as well as accept-reject schemes [Kim et al., 1998].

There is no reason the block sizes have to be one of the two extremes–the state paths can be sampled in blocks larger than 1 and smaller than n. Using larger block sizes will likely improve mixing; however, it will also increase the difficulty of designing proposal distributions. For example, [Shephard and Pitt, 1997] also use a Gibbs sampler targeting blocks of states.

Particle Based Approximations to State Distributions

An alternative algorithm for sampling from the state smoothing distribution of general SSMs is the Forward-Filtering-Backward-Sampling algorithm (FFBSa)[1]. Instead of iterating between components or blocks of states, these algorithms can be used to sample all states at once within a Gibbs scheme alternating between the two conditional posteriors [Niemi and West, 2010].

This is an approximate method that is suggested by the decomposition

$$\begin{aligned} p(x_{1:T} \mid y_{1:T}, \theta) &= p(x_T \mid y_{1:T}, \theta) \prod_{t=1}^{T-1} p(x_t \mid x_{t+1:T}, y_{1:T}, \theta) \\ &= p(x_T \mid y_{1:T}, \theta) \prod_{t=1}^{T-1} p(x_t \mid x_{t+1}, y_{1:T}, \theta) \\ &= p(x_T \mid y_{1:T}, \theta) \prod_{t=1}^{T-1} p(x_t \mid x_{t+1}, y_{1:t}, \theta). \end{aligned}$$

[1]These are not to be confused with closed-form Forward-Backward algorithms for Linear-Gaussian models or discrete state space HMMs.

FFBSa first runs a particle filter forward and saves the particles and weights from each marginal filtering distribution $p(x_t \mid y_{1:t})$. This is the "forward" part. The "backward" part consists of recursively (backwards in time) sampling from the distributions $p(x_t \mid x_{t+1}, y_{1:t}, \theta)$. This is done by first sampling from the last filtering distribution $p(x_T \mid y_{1:T}, \theta)$, then iterating backwards in time, revising the saved filtering distributions. That is, for $t < T$, sample from $p(x_t \mid x_{t+1}, y_{1:t})$, where

$$p(x_t \mid x_{t+1}, y_{1:t}, \theta) = \frac{f(x_{t+1} \mid x_t, \theta) p(x_t \mid y_{1:t}, \theta)}{\int f(x_{t+1} \mid x_t, \theta) p(x_t \mid y_{1:t}, \theta) dx_t},$$

by reweighting the weights associated with $p(x_t \mid y_{1:t}, \theta)$ according to how well they cohere with x_{t+1}^i, and then sampling from the marginal's samples according to those new weights. Note that this FFBSa algorithm is different from the forward-filtering backward smoothing algorithm (FFBSm), which yields only marginal distributions $p(x_t \mid y_{1:T}, \theta)$ [Doucet and Johansen, 2011].

Particle Markov Chain Monte Carlo

[Andrieu et al., 2010] detail two approaches to sample from either of the target posterior distributions that were discussed above. They call these methods Particle Marginal Metropolis Hastings (PMMH), and Particle Gibbs (PG). These algorithms are based on the more general algorithm discussed in [Andrieu and Roberts, 2009], which is not explicitly designed to make use of particle filters. PMMH and PG, on the other hand, are, and they both can target either the full posterior of states and parameters, or the marginal posterior of just parameters. This section will focus on the PMMH algorithm.

To motivate the description of this algorithm, first consider the estimation of a relatively simple state space model whose likelihood is tractable. In this case, targeting the marginal posterior would involve the use of the following Hastings' ratio:

$$\frac{p(\theta')p(y_{1:t} \mid \theta')q(\theta \mid \theta')}{p(\theta)p(y_{1:t} \mid \theta)q(\theta' \mid \theta)}.$$

However, because evaluating the likelihood is rarely possible, one needs to target the full joint posterior, and integrate out the undesired state samples. If one is able to sample directly from the model's smoothing distribution $p(x_{1:T} \mid y_{1:T}, \theta)$, he might consider using the proposal distribution $p(x'_{1:T} \mid y_{1:T}, \theta')q(\theta' \mid \theta)$. The Hastings' ratio for this algorithm would be

$$\frac{p(x'_{1:T}, y_{1:T} \mid \theta')p(\theta')p(x_{1:T} \mid y_{1:T}, \theta)q(\theta \mid \theta')}{p(x_{1:T}, y_{1:T} \mid \theta)p(\theta)p(x'_{1:T} \mid y_{1:T}, \theta')q(\theta' \mid \theta)} = \frac{p(\theta')p(y_{1:t} \mid \theta')q(\theta \mid \theta')}{p(\theta)p(y_{1:t} \mid \theta)q(\theta' \mid \theta)}. \tag{3.11}$$

Again, though, the tractability of the likelihood is required. Moreover, sampling from the smoothing distribution is of the same difficulty as evaluating the likelihood.

Recall from section (2.3) that particle filters offer approximate samples from the smoothing distribution and estimations of the likelihood. One might consider approximating the above ideal algorithm by drawing one path from the approximation of the smoothing distribution, and using the estimated likelihood when calculating the Hastings' ratio. Even though this might be an approximate strategy, it would be tempting to consider because designing the low-dimensional proposal distribution $q(\theta' \mid \theta)$ and the low-dimensional proposal distributions for the particle filter is relatively easy. Removing the assumption of computationally convenient priors comes with the cost of design difficulty of the proposal, but this would be mitigated as the state samples "cohere" with the proposed parameters. Last, it is also much simpler to program. Assuming the code for a particle filter is already written, a program implementing the PMMH algorithm would strongly resemble the program implementing a regular MH algorithm.

Fortunately, this seemingly approximate strategy turns out to be exact with only a minimal set of assumptions [Andrieu et al., 2010](section 4.1). This can be seen if the acceptance probability is written in terms of the *exact* target and proposal distributions. Both of these are defined on a much higher-dimensional space, as they describe all of the randomly-generated output from a run of the particle filter. From this perspective, the acceptance ratio that was calculated previously is not random because the likelihood estimate is a deterministic function of the particle filter's output. Also, because the distribution for all the particle filter output is featured in both the target and proposal, these expressions cancel. This explains why the acceptance probability is a relatively simple expression.

Before the densities for the high-dimensional random output are written, consider how a particle filter generates all of its output. First, the parameter is proposed from a portion of the MH proposal distribution. Then, using that parameter, a particle filter is run through the entire dataset. Afterwards, one of the particle paths is selected with probability proportional to its final unnormalized weight.

For $t = 1, \ldots, T-1$ define $\mathbf{a}_t = (a_t^1, \ldots, a_t^N)$ to be the N indices resampled at the end of time t; also rewrite the particle filter's N state samples at time t as $\mathbf{x}_t$, and denote by k' a realization of the index of the final particle chosen. Then, the exact MH proposal distribution on the higher dimensional space can be written as

$$q(k', u', \theta' \mid k, u, \theta, y_{1:T}) = p(k' \mid u')\psi(u' \mid y_{1:T}, \theta')q(\theta' \mid \theta), \tag{3.12}$$

where $u = (\mathbf{x}_{1:T}, \mathbf{a}_{1:T-1})$ is the set of all random variables generated by a particle filter. The random variable K' is similar to the ancestor indices in that they are both indices; however it is different in the sense that there is only one that is chosen. Conditional on u', K' is drawn from the set $\{1, \ldots, N\}$ with probability masses $\tilde{w}_T^1, \ldots, \tilde{w}_T^N$[2].

The exact target distribution is

$$\frac{l(u, \theta, y_{1:T})p(k \mid u)\psi(u \mid y_{1:T}, \theta)p(\theta \mid y_{1:T})}{p(y_{1:T} \mid \theta)}, \tag{3.13}$$

where $l(u, \theta, y_{1:T})$ is the particle filter's likelihood estimate. Evaluating the Hastings' ratio with (3.12) and (3.13) simplifies to a Hastings' ratio that looks like the ideal model's, only with the particle filter's estimated likelihood:

$$\frac{l(u', \theta', y_{1:T})p(k' \mid u')\psi(u' \mid y_{1:T}, \theta')p(\theta')}{l(u, \theta, y_{1:T})p(k \mid u)\psi(u \mid y_{1:T}, \theta)p(\theta)} \frac{p(k \mid u)\psi(u \mid y_{1:T}, \theta)q(\theta \mid \theta')}{p(k' \mid u')\psi(u' \mid y_{1:T}, \theta)q(\theta' \mid \theta)}. \tag{3.14}$$

Two conditions remain to be checked. First, this unnormalized target density needs

[2]This notation uses abbreviated subscripts. That is $\tilde{w}_T^i = \tilde{w}_{1:T}^i$.

to be integrable. Second, this high-dimensional joint distribution needs to have as its marginal the correct target: $p(x_{1:T}, \theta \mid y_{1:T})$. These both follow from the following Lemma (proof in appendix).

Lemma 1. *Let $u, \theta, k, y_{1:T}, x_{1:T}$ be defined as they are above. Also, assume that multinomial resampling is being performed at every time step. Then*

$$\int l(u, \theta, y_{1:T}) p(k \mid u, \theta) \psi(u \mid y_{1:T}, \theta) d[u \setminus x_{1:T}^{k}] = p(x_{1:T}^{k}, y_{1:T} \mid \theta).$$

The integration is done with respect to all of the auxiliary random variables with the exception of the index k' and its associated path sample. Note that this may be an abuse of notation, because the index/ancestor random variables are integrated with respect to the counting measure.

It should be noted that the proof simplifies when the target is reduced to the marginal posterior $p(\theta \mid y_{1:T})$. Because neither state paths, nor samples of k need to be retained, the proposal distribution simplifies to $\psi(u' \mid y_{1:T}, \theta) q(\theta' \mid \theta)$. A reference to a particle filter's "unbiased" likelihood estimate refers to the fact that the expectation of $l(u, \theta, y_{1:T})$, taken with respect to the density $\psi(u \mid y_{1:T}, \theta)$ is equal to $p(y_{1:T} \mid \theta)$. With this property, the target distribution

$$\frac{l(u, \theta, y_{1:T}) \psi(u \mid y_{1:T}, \theta) p(\theta \mid y_{1:T})}{p(y_{1:T} \mid \theta)} \tag{3.15}$$

has as its marginal the marginal posterior, and is integrable as long as the smaller-dimensional posterior is. Although their description of particle filters differs from the description given in this document, [Pitt et al., 2010] prove that many popular particle filters feature unbiased likelihood estimates. The PMMH algorithm used to estimate the models in chapter (4) uses a Rao-Blackwellized particle filter, and its proof of unbiasedness is given in the appendix.

Below is a demonstration of the PMMH technique using simulated data from the

Algorithm 5 Particle Marginal Metropolis-Hastings

```
procedure PMMH($q_t$)
    if $t$ equals 1 then
        theta[1] ← $\theta^1$
        run particle filter through $y_{1:T}$ using $\theta^1$
        likes[1] ← $\hat{p}(y_{1:T} \mid \theta^1)$
        states[1] ← $X^K_{1:T}$ where $K \sim$ Multinomial$[1, \tilde{w}^1_T, \ldots, \tilde{w}^N_T]$        ▷ optional
    else
        draw $\theta' \sim q_t(\theta' \mid \theta_{t-1})$
        run particle filter through $y_{1:T}$ using $\theta'$
        tempState ← $X^L_{1:T}$ where $L \sim$ Multinomial$[1, \tilde{w}^{1'}_{1:T}, \ldots, \tilde{w}^{N'}_{1:T}]$        ▷ optional
        $p \leftarrow 1 \wedge p(\theta')\hat{p}(y_{1:t} \mid \theta')q(\theta \mid \theta')/p(\theta)\hat{p}(y_{1:t} \mid \theta)q(\theta' \mid \theta)$
        $u \leftarrow U \sim$ Uniform$(0, 1]$
        if u< p then
            theta[t] ← $\theta'$
            states[t] ←tempState        ▷ optional
        else
            theta[t] ← theta[t-1]
            states[t] ← states[t-1]        ▷ optional
        end if
    end if
end procedure
```

following model:

$$y_t = x_t + v_t, \qquad v_t \overset{iid}{\sim} \text{Normal}(0, \sigma^2_v) \tag{3.16}$$

$$x_t = \alpha x_{t-1} + w_t, \qquad v_t \overset{iid}{\sim} \text{Normal}(0, \sigma^2_w). \tag{3.17}$$

A time series of length 400 was generated with true parameters $\alpha = .91$, $\sigma^2_w = 1$ and $\sigma^2_v = 1.5^2$. $10,000$ iterations were performed, and on each iteration, a particle filter with 750 particles was used. The chain was initialized with $\alpha^0 = 0$, $\sigma^{2,0}_w = 4.8$ and $\sigma^{2,0}_v = .2$.

For the proposal distribution, a random walk was chosen for the transformed parameters (so one must take into account the Jacobian). α was transformed into $\text{logit}((1+\alpha)/2) = \log((1+\alpha)/(1-\alpha))$, and the two variance parameters were transformed by taking their logarithm. The covariance matrix for this proposal distribution was diagonal, with variances .04, 0.004, 0.06. Last, the priors were chosen to be uninformative. A Uniform(-1,1) prior was chosen for α, and for both of the variance parameters, an InverseGamma$(.001, .001)$ prior was chosen.

The program took 1.04 hours to run. (3-1) shows a scatterplot matrix of the draws, (3-2) the draws versus pseudo-time, and (3-3) shows a correlogram for the three chains. In order to better show convergence, no burn-in was discarded and no subsampling was performed. However, burn-in draws were discarded for parameter estimation. The parameter estimates, along with their standard errors, as computed by the R package `mcmcse` [Flegal et al., 2017], are shown in (3.1).

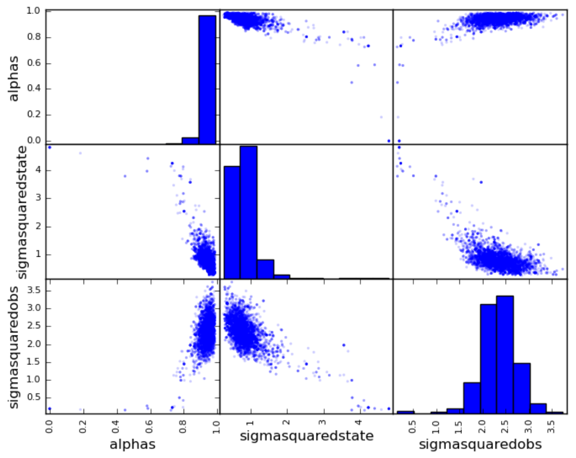


Figure 3-1: PMMH scatterplot matrix

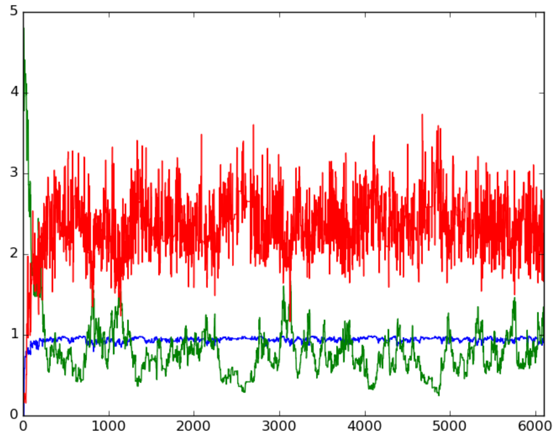


Figure 3-2: PMMH draws

Accelerating the PMMH Algorithm

For more complex models, it becomes much more difficult to choose the proposal distribution's covariance matrix. In order to properly take into account the scale and shape of the posterior, [Gelman et al., 2013] suggest that the covariance matrix be set to a multiple of the posterior's true covariance matrix (if the parameters are suitably

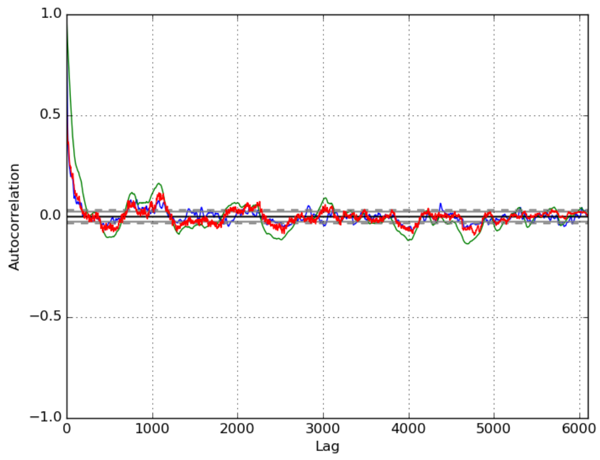


Figure 3-3: PMMH correlograms

	alpha	sigmaSquaredW	sigmaSquaredV
est	0.9352618205	0.8284527762	2.3384284056
se	0.0044350049	0.0486459661	0.0289953628

Table 3.1: Noisy AR1 Parameter Estimates

transformed); if Σ is the true d-dimensional covariance matrix, then the optimal scale for a random walk proposal is $\Sigma(2.4^2/d)$.

Unfortunately, this posterior covariance is rarely, if ever, known. Tuning the algorithm's settings to something that is close to this covariance matrix becomes more difficult as the dimension of the parameter space increases. Moreover, in the case of the PMMH algorithm, the situation is even more severe. It is not clear whether a working d should be set closer to the dimension of the ideal target, or closer to the dimension of the actual target, taking into account all of the auxiliary random variables. Interestingly, under certain conditions, as the number of particles grows, this effective dimension reduces, as the algorithm more closely approximates the ideal algorithm [Andrieu et al., 2010]. However, increasing the number of particles comes at a computational expense. For these reasons, d is tuned to be some percentage of the dimension of the full parameter space.

In addition to its obscuring of standard Metropolis-Hastings tuning advice, the PMMH is much more computationally demanding than standard Metropolis-Hastings samplers. If M samples are desired from the posterior of $p(\theta \mid y_{1:T})$, then using the PMMH algorithm described in the previous section with N particles requires TNM state samples and weight updates, and $(T-1)NM$ ancestor samples. Unfortunately, this cost is difficult to parallelize away. MCMC iterations cannot easily be performed in parallel, otherwise they will not form a Markov chain. Particle filter samples cannot easily be sampled in parallel, either, due to the resampling mechanism. Taking all these difficulties into account, and with the aim of maximizing the number of effective samples per unit time, this section describes a number of techniques that are used in the estimation of all models in chapter (4).

Following along the lines of [Haario et al., 2001], after some chosen iteration number t_0, the parameter proposal is allowed to adapt at every iteration. At each adaptation point, the proposal's covariance matrix is set to a scaled multiple of a matrix "close to" the empirical covariance matrix. Before the scaling is done, a small ϵ is added to all of the diagonal elements to ensure that the covariance matrix remains positive definite. In other words, $q_i(\theta' \mid \theta) = \mathrm{N}(\theta'; \theta, \Sigma_i)$, where

$$\Sigma_i = \begin{cases} \Sigma_0 & i \le t_0 \\ \frac{2.4^2}{d}\left[S^2_{i-1} + \epsilon I_{d_\theta}\right] & t_1 \ge i > t_0 \\ \Sigma_{i-1} & \text{else.} \end{cases} \tag{3.18}$$

Recursive formulas for the sample covariance matrix allow for more efficient programs.

Unlike their algorithm, however, this document picks some time t_1 where adaptation ends. In this way, after time t_1, the sampler regains the Markov property, and the results from earlier in this chapter can still be used to derive properties of the estimators. Samples before this time are discarded along with any additional burn-in that is chosen at the user's discretion.

Another beneficial technique is to, when possible, use Rao-Blackwellized particle filters inside this PMMH algorithm. Comparing the weight update formulas in section (2.3), it can be seen that Rao-Blackwellized particle filters have smaller variance compared to regular SISR filters. This is owing to the fact that RBPFs do not need to sample the entire state vector.

This idea is not completely novel. It was anticpated by [Jacob et al., 2009] and implemented in [Kokkala and Särkkä, 2014] for a model that is much different than the ones considered in this document. However, to the best of the author's knowledge, no proof has previously been given that the estimate of the RBPF likelihood is unbiased, which is the main requirement for this sampler to be exact. A proof of the following lemma is given in the appendix.

Lemma 2. *The Rao-Blackwellized particle filter's likelihood estimate*

$$\hat{p}(y_{1:T} \mid \theta) = \hat{p}(y_1 \mid \theta) \prod_{t=2}^{T} \hat{p}(y_t \mid y_{1:t-1}, \theta)$$

is unbiased, when multinomial resampling is conducted at every time point.

The formulas for $\hat{p}(y_t \mid y_{1:t-1})$ and $\hat{p}(y_1)$ are given by equations (2.20) and (2.21).

Another practical "trick" that can be used is to run $n_p > 1$ particle filters in parallel at every iteration of the chain. After they are all able to yield their estimate for the likelihood, the average of their approximations may be used to calculate the acceptance probability at the current iteration. The formula for the reduced-variance estimator is

$$\tilde{p}(y_{1:T} \mid \theta) = n_p^{-1} \sum_{i=1}^{n_p} \hat{p}^i(y_{1:T} \mid \theta).$$

The fact that this PMMH algorithm continues to be exact follows trivially from the linearity of the expectation operator. Also, because these particle filters do not interact, it is "pleasingly parallel" and thus relatively simple to implement. The number of particle filters running at each iteration (n_p) may be set to the number of available threads on a computer. At the time of writing this, the author is using an Intel Xeon Processor E3-1241 v3, where the number of available threads is 8. It is very likely that the performance of this algorithm could be drastically improved if this same strategy was implemented on a graphics processing unit (GPU) where the number of available threads is much higher. This development is left for future research.

This adaptive version of the PMMH algorithm is given in detail below. This

Algorithm 6 Adaptive Particle Marginal Metropolis-Hastings

```
procedure ADAPMMH
    set $d$ to be some percentage of the full parameter space
    if $t$ equals 1 then
        theta[1] $\leftarrow \theta^1$
        run $n_p$ (possibly Rao-Blackwellized )particle filters through $y_{1:T}$ in parallel each using $\theta^1$
        likes[1] $\leftarrow n_p^{-1} \sum_{i=1}^{n_p} \hat{p}^i(y_{1:T} \mid \theta^1)$
    else if $t_0 \geq t > 1$ or $t > t_1$ then
        draw $\theta' \sim q_t(\theta' \mid \theta_{t-1})$
        run $n_p$ Rao-Blackwellized particle filters through $y_{1:T}$ in parallel each using $\theta'$
        likes[t] $\leftarrow n_p^{-1} \sum_{i=1}^{n_p} \hat{p}^i(y_{1:T} \mid \theta')$
        $p \leftarrow 1 \wedge p(\theta')\hat{p}(y_{1:t} \mid \theta')q(\theta \mid \theta')/p(\theta)\hat{p}(y_{1:t} \mid \theta)q(\theta' \mid \theta)$
        $u \leftarrow U \sim$ Uniform$(0, 1]$
        if u< p then
            theta[t] $\leftarrow \theta'$
        else
            theta[t] $\leftarrow$ theta[t-1]
        end if
    else if $t_1 \geq$ t $> t_0$ then
        change covariance of $q_t$ using 3.18
        draw $\theta' \sim q_t(\theta' \mid \theta_{t-1})$
        run $n_p$ (possibly Rao-Blackwellized )particle filters through $y_{1:T}$ in parallel each using $\theta'$
        likes[t] $\leftarrow n_p^{-1} \sum_{i=1}^{n_p} \hat{p}^i(y_{1:T} \mid \theta')$
        $p \leftarrow 1 \wedge p(\theta')\hat{p}(y_{1:t} \mid \theta')q(\theta \mid \theta')/p(\theta)\hat{p}(y_{1:t} \mid \theta)q(\theta' \mid \theta)$
        $u \leftarrow U \sim$ Uniform$(0, 1]$
        if u< p then
            theta[t] $\leftarrow \theta'$
        else
            theta[t] $\leftarrow$ theta[t-1]
        end if
    end if
end procedure
```

algorithm is used in the following chapter to estimate the models of interest that guide investment decisions.

Chapter 4

Multivariate Stochastic Volatility Models

Introduction

[Lopes and Polson, 2010] and [Chib et al., 2009] review multivariate stochastic volatility models, separating this large grouping into several categories. This proposal chooses to focus on factor stochastic volatility (FSV) models, which resemble the univariate stochastic volatility model described in section (2.3) in that they feature a latent process that directly controls the conditional covariance matrix.

The name for these factor stochastic volatility models comes from the assumption that there exists a lower-dimensional random "factor" vector, denoted by $\mathbf{f}_t$, "driving" the higher-dimensional returns. These factors are unobserved, and they are usually assumed to be, conditionally on the state process, distributed independently across time, but not identically. The mean of the factor is usually assumed to be fixed at zero, however the variances of each factor are assumed to change stochastically over time.

An Initial Model

As a starting point, consider the following model from [Jacquier et al., 1999]. The d_x-dimensional unobserved state process is assumed to be a vector autoregressive process with a nonzero mean. The initial state's distribution, as well as the state transition equations for $t = 2, \ldots, T$, are

$$\mathbf{x}_t - \boldsymbol{\mu} = \boldsymbol{\Phi}\left[\mathbf{x}_{t-1} - \boldsymbol{\mu}\right] + \mathbf{w}_t \tag{4.1}$$

$$\mathbf{x}_1 \sim \mathrm{N}\left(\boldsymbol{\mu}, \mathrm{diag}\left[\frac{\sigma_1^2}{(1-\phi_1^2)}, \ldots, \frac{\sigma_{d_x}^2}{(1-\phi_{d_x}^2)}\right]\right) \tag{4.2}$$

where $\boldsymbol{\Phi} = \mathrm{diag}(\phi_1, \ldots, \phi_{d_x})$ is composed of the autoregressive parameters for the states. The sequence of state noise vectors $\{\mathbf{w_t}\}$ are iid Gaussian with a covariance matrix $\mathbf{Q} = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_{d_x}^2)$. This parametrization is frequently referred to as the "centered" parametrization, because the mean of the state process is not assumed to be the zero vector.

The initial model's observation equation is

$$\mathbf{y}_t = \mathbf{B}\mathbf{f}_t + \mathbf{v}_t, \tag{4.3}$$

where each $\mathbf{y}_t$ is a vector of returns of dimension $d_y > d_x$, $\mathbf{B} \in \mathbf{R}^{d_y \times d_x}$ is the matrix of constant factor loadings, $\{\mathbf{v}_t\}$ is iid Gaussian noise with a diagonal covariance matrix $\mathbf{R}$, $\{\mathbf{z}_t\}$ is iid Gaussian noise with an identity covariance matrix, and each $\mathbf{f}_t = \exp(\mathrm{diag}[\mathbf{x}_t/2])\mathbf{z}_t$ is a latent factor vector at time t. Here $\exp(\cdot)$ represents the matrix exponential. Moreover, the noise vectors $\{\mathbf{z}_t\}$ and $\{\mathbf{v}_t\}$ are assumed to be independent of each other.

Equation (2.5) suggests how to obtain forecasts. This means what are needed are expectations of the observations one step in the future, conditional on the current state and parameter value. One can verify that $E[\mathbf{y}_{t+1} \mid \mathbf{x}_t, \theta] = \mathbf{0}$, and

$$\mathrm{Var}[\mathbf{y}_{t+1} \mid \mathbf{x}_t, \theta] = \mathbf{B}\mathbf{E}_t\mathbf{B}' + \mathbf{R} \tag{4.4}$$

$$\mathbf{E}_t = \exp\left\{\mathrm{diag}[\boldsymbol{\mu} + \boldsymbol{\Phi}\left(\mathbf{x}_t - \boldsymbol{\mu}\right)] + \mathbf{Q}/2\right\}. \tag{4.5}$$

Using the law of total variance again, it can be seen that the one-step-ahead forecast moments are

$$E[\mathbf{y}_{t+1} \mid \mathbf{y}_{1:t}] = \mathbf{0} \tag{4.6}$$

$$\mathrm{Var}[\mathbf{y}_{t+1} \mid \mathbf{y}_{1:t}] = E\left[\mathbf{B}\mathbf{E}_t\mathbf{B}' + \mathbf{R} \mid \mathbf{y}_{1:t}\right]. \tag{4.7}$$

The second quantity can be approximated with the following formula:

$$\frac{1}{N_\theta}\sum_{i=1}^{N_\theta}\sum_{j_i=1}^{N_x}\tilde{w}_t^{j_i}\mathbf{B}^i\exp\left\{\mathrm{diag}[\boldsymbol{\mu}^i + \boldsymbol{\Phi}^i\left(\mathbf{x}_t^{j_i} - \boldsymbol{\mu}^i\right)] + \mathbf{Q}^i/2\right\}\mathbf{B}^{i'} + \mathbf{R}^i. \tag{4.8}$$

This average takes parameter values drawn from a collection of MCMC samples, and for each one of these parameter values, the weighted average is taken over the particle filter's samples. If this is too computationally expensive, then the outer sum can be removed by using only one particle filter. In this case, a suitable one-number summary of the posterior distribution would be chosen as the parameter vector used to instantiate a particle filter, for example the mean or the mode.

Identifiability of the Initial Model

An understanding of any issues of a model's nonidentifiability is important for choosing among different parametrizations of the model and restrictions on the parameter space. These choices, in turn, directly impact posterior interpretability, prior choice, as well as efficiency of MCMC algorithms [Kastner and Frühwirth-Schnatter, 2014].

Identifiability is a common issue impacting factor stochastic volatility models

[Aguilar and West, 2000]. Many (but not all) of the issues of static factor models carry over to the case of dynamic factor models. Moreover, many of these issues may or may not apply, depending on what kind of assumptions are made about the model structure. To clarify this particular situation, this section offers a more detailed description of all instances of nonidentifiability that may arise in the case of the initial model of [Jacquier et al., 1999]. With this complete understanding it will be possible to understand how the extended models, introduced later in this chapter, handle their own issues of nonidentifiability.

First, consider post-multiplying the loadings matrix $\mathbf{B}$ by an invertible matrix $\mathbf{P}$. It is easy to see that that the observation equation satisfies

$$\mathbf{B}\mathbf{f}_t + \mathbf{v}_t = \mathbf{B}\mathbf{P}\mathbf{P}^{-1}\mathbf{f}_t + \mathbf{v}_t. \tag{4.9}$$

For an alternative state space model with a state process that corresponds with this transformed factor vector $\mathbf{P}^{-1}\mathbf{f}_t$, the alternative loadings matrix $\mathbf{BP}$ will yield the same likelihood function as the initial model.

There does not exist a corresponding state process for every choice of invertible matrix; however, consider as one example $\mathbf{P} = cI_{d_y}$. Then, by properties of the matrix exponential, the new state would correspond with the alternative factor vector $\exp(\text{diag}[\mathbf{x}_t - 1_{d_x} 2\log c]/2)\mathbf{z}_t$. This can be generalized with the following result.

Lemma 3. *Let a factor stochastic volatility model be defined by the collection of parameters* $\{\mathbf{B}, \mathbf{R}, \boldsymbol{\mu}, \boldsymbol{\Phi}, \sigma_1^2, \ldots, \sigma_{d_x}^2\}$*, and* $\boldsymbol{P}$ *be a diagonal matrix with all its diagonal entries positive. Then the factor stochastic volatility model defined by the collection of parameters* $\{\mathbf{BP}, \mathbf{R}, \boldsymbol{\mu} + \boldsymbol{u}, \boldsymbol{\Phi}, \mathbf{Q}\}$ *yields the same likelihood. Here* $\boldsymbol{u} = (2\log(\boldsymbol{P}^{-1})_{11}, \ldots, 2\log(\boldsymbol{P}^{-1})_{nn})'$*, and* $\log(\cdot)$ *denotes the matrix logarithm obtained by taking the log of all the diagonal elements.*

The second example of nonidentifiability comes from the possibility of "column-switching." The idea is that, if one permutes the rows of the state vector and the columns of the loadings matrix, the same likelihood will be obtained. Because this document only considers factors of a certain form, more general factor rotations beyond permutations do not present any identifiability issues.

Lemma 4. *Let a factor stochastic volatility model be defined by the collection of parameters* $\{\mathbf{B}, \mathbf{R}, \boldsymbol{\mu}, \boldsymbol{\Phi}, \mathbf{Q}\}$*, and let* $\boldsymbol{P}$ *be a permutation matrix. Then the factor stochastic volatility model associated with the parameters* $\{\boldsymbol{BP}', \mathbf{R}, \mathbf{P}\boldsymbol{\mu}, \mathbf{P}\boldsymbol{\Phi}\mathbf{P}', \mathbf{P}\boldsymbol{Q}\mathbf{P}'\}$ *yields the same likelihood.*

Last, the effects of "sign-switching" are considered. If one makes some of the columns of $\mathbf{B}$ negative, then the conditional covariance matrix is left unchanged.

Lemma 5. *Let a factor stochastic volatility model be defined by the collection of parameters* $\{\mathbf{B}, \mathbf{R}, \boldsymbol{\mu}, \boldsymbol{\Phi}, \mathbf{Q}\}$*, and let* $\mathbf{S}$ *be a diagonal matrix with all entries equal to either* ± 1*. Then the factor stochastic volatility model associated with the parameters* $\{\mathbf{BS}, \mathbf{R}, \boldsymbol{\mu}, \boldsymbol{\Phi}, \mathbf{Q}\}$ *yields the same likelihood.*

To prevent column-switching and scale ambiguity, this paper adopts the most common restriction on $\mathbf{B}$. The loadings matrix is assumed to be of the following form:

$$\mathbf{B} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ b_{2,1} & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & 0 \\ b_{d_x,1} & b_{d_x,2} & b_{d_x,3} & \cdots & 1 \\ b_{d_x+1,1} & b_{d_x+1,2} & b_{d_x+1,3} & \cdots & b_{d_x+1,d_x} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ b_{d_y,1} & b_{d_y,2} & b_{d_y,3} & \cdots & b_{d_y,d_x} \end{bmatrix}. \tag{4.10}$$

The third type of nonidentifiability, sign-switching, is dealt with by selecting a prior that favors positive $\mathbf{B}$ elements.

Real Data Analysis with the Initial Model

This section analyzes weekly adjusted closing prices for the nine Select Sector SPDR exchange traded funds (ETFs) spanning the time interval starting on 2005/12/23 and ending on 2014/5/1. This dataset was selected in part because it spans the financial crisis of 2008. Models will be estimated using data prior to the crisis, and then be used to generate out-of-sample forecasts on post-crisis data. In particular, it is highly desirable for a model that is estimated this way to still be able to produce accurate estimates of the covariance matrix throughout the more turbulent period. Only by having a model "run the gauntlet" will one be able to know how robust its measures of finacial risk are.

Each of these nine ETFs represents the nine sectors[1] of the Standard and Poor's 500 Index, a proxy for the American economy. The log returns, as well as the cumulative sum of these log returns, are plotted below. The vertical bar in (4-1) roughly marks the beginning of the financial crisis.

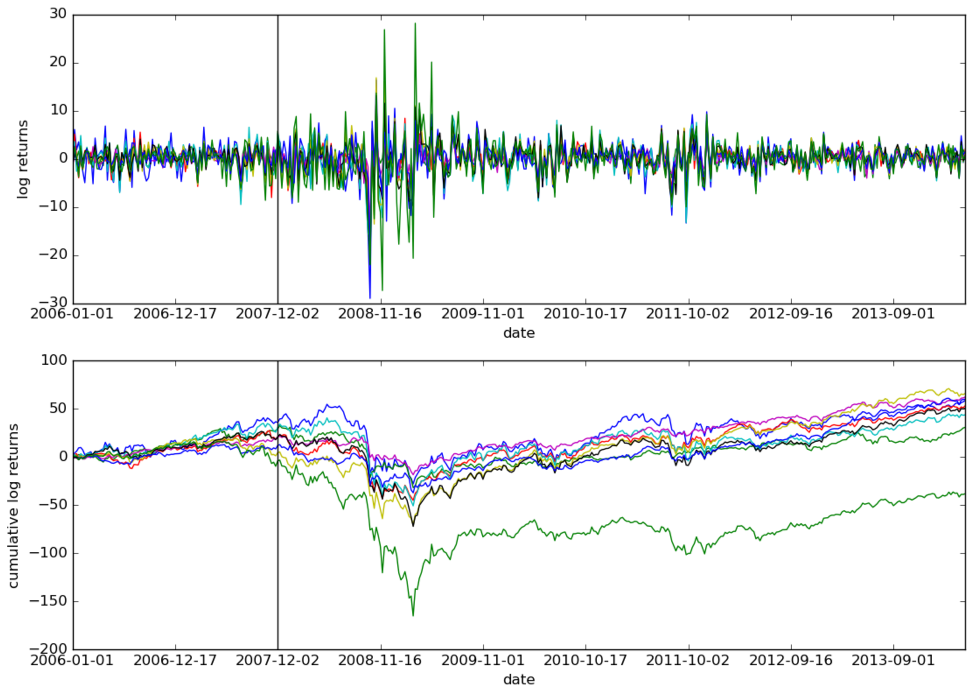


Figure 4-1: 9 SPDR ETFs: returns and cumulative returns

First, the model of [Jacquier et al., 1999] with one latent volatility process is estimated using a relatively small set of data prior to this cut-off point. The resulting

[1]On 10/8/2015 a tenth real estate sector ETF was added to this collection.

parameter estimates are used to generate forecasts on a larger out-of-sample dataset coming after the cut-off point. The forecast distributions are evaluated on the data, and then these evaluations are used to suggest an alternative model.

Parameter Estimation

Even though this model has been estimated in the literature, priors were specified without examining outside results. First, $(\phi + 1)/2$ was given a Beta(6.27, 2.07) prior. This makes the prior mean and variance for ϕ .9 and .025, respectively. The restriction that the log volatility process is stationary, i.e. that $-1 < \phi < 1$, is very common and economically reasonable. Setting the prior mean to .9 anticipates positive autocorrelation. This volatility persistence or "volatility clustering" is a well-known "stylized feature" of stock returns.

For all of the parameters in the vector $\mathbf{B}$ except the first, a Normal prior with mean $1_{d_y - 1}$ and variance $.125 I_{d_y - 1}$ was used. The top left element of $\mathbf{B}$ is set to 1, and this element corresponds with the returns for "XLE", the energy sector. So, the remaining coefficients have the interpretation of "relative market risk." A value higher (lower) than 1 indicates that the variances and covariances for this sector increase by a higher (lower) multiple.

An uninformative Normal(0,10) prior is put on μ, an InverseGamma(.5,.5) prior was put on σ^2, and last, it is assumed that the diagonal elements of $\mathbf{R}$ are all iid, each with an InverseGamma(1,1) prior. These are both "fat-tailed" as they don't have means; moreover, they have modes around what would be suspected.

Algorithm (6) is used used to draw 50,000 samples from the target posterior. Each particle filter uses 5 particles, and every iteration uses 8 particle filters in parallel. The program took .15 hours to run on an Intel Xeon Processor E3-1241 v3 and had an acceptance rate of 27.7%. As for the parameters themselves, their transformed versions are proposed with a Gaussian multivariate random walk. The variance parameters and ϕ are transformed as follows: the logarithm of half of each variance parameter is taken, and ϕ is transformed into $\log\left([1+\phi]/[1-\phi]\right)$. 500 samples were discarded as burn-in because the covariance was adapting for the first 450 iterations.

The initial values were taken from the last iteration of a trial run of the algorithm. More burn-in wasn't discarded because the initial parameters were likely drawn from a distribution very close to the stationary distribution.

Even though it does not generate state samples with the optimal proposal distribution, a regular Bootstrap Filter is used for this. The justification for this is computational efficiency. As long as the code is specialized enough to avoid the unnecessary density calculations for the weight update formula, a Bootstrap Filter is faster than the more elaborate particle filters such as the Auxiliary Particle Filter.

The parameter estimates, along with their batch means standard errors, are printed below in tables (4.1), (4.2), (4.3). The trace plots and histograms can be seen in (A-1), (A-2), and (A-3). The trace plots do not discard any burn-in samples; however, the histograms and scatterplots do discard the initial 500 samples as burn-in.

Table 4.1: Initial Model Beta Estimates

	xlu beta	xlk beta	xlb beta	xlp beta	xly beta	xli beta	xlv beta	xlf beta
est	0.80964	1.19713	1.34597	0.63260	1.19113	1.07208	0.75803	1.16886
se	0.00658	0.00732	0.00794	0.00393	0.00674	0.00621	0.00460	0.00719

Table 4.2: Initial Model State Process Estimates

	phi	mu	sig. sq.
est	0.28926	0.76999	0.32545
se	0.02119	0.01397	0.01084

Table 4.3: Initial Model Error Variance Matrix Diagonal Estimates

	xle r	xlu r	xlk r	xlb r	xlp r	xly r	xli r	xlv r	xlf r
est	6.52486	2.52616	1.31751	2.14624	0.80179	0.67280	0.75594	0.97397	2.05327
se	0.03263	0.01338	0.00816	0.01498	0.00442	0.00563	0.00581	0.00578	0.01305

Out-of-Sample Forecasting

Evaluating out-of-sample forecasts is done in three ways. First, (approximate) scoring rules are used to measure this model's forecasting ability. Scoring rules are

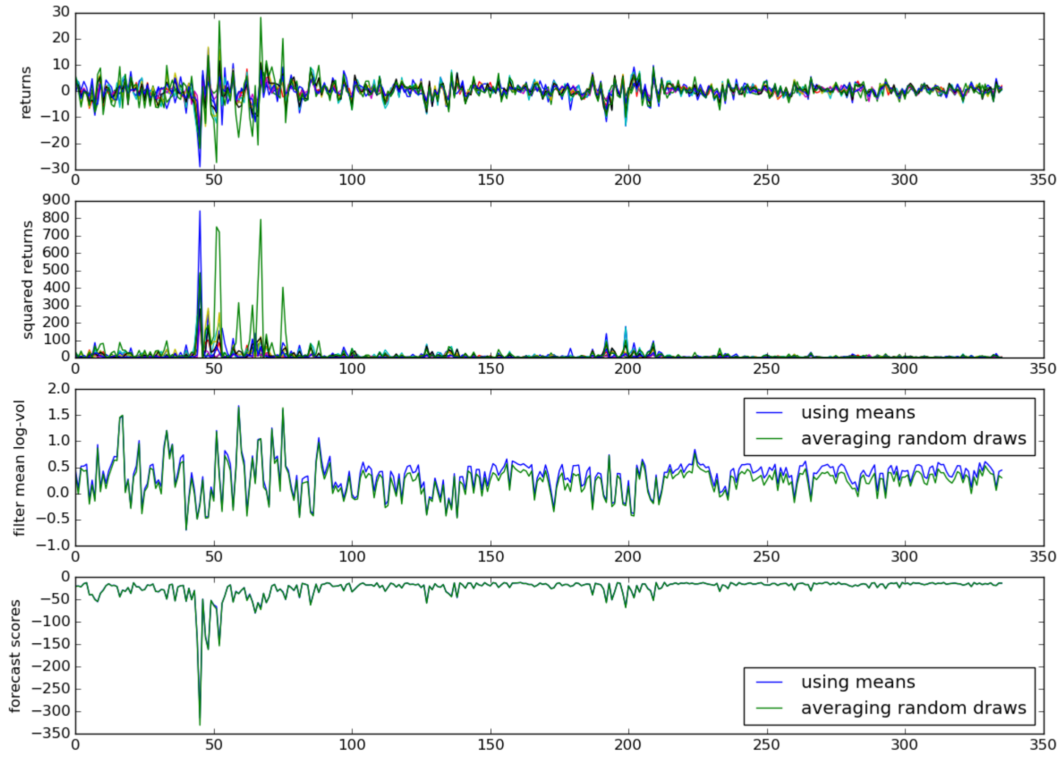


Figure 4-2: 9 SPDR ETFs: Out-of-sample returns, squared returns, filtered means, and log forecast densities

functions that take both a forecast distribution and a materialized value, and then return a real-valued number representing the accuracy of the forecast [Gneiting and Raftery, 2007]. The first scoring rule used here is motivated by the log predictive Bayes factor [Amisano and Geweke, 2008]. The log predictive Bayes factor in favor of model $\mathcal{M}_1$ over model $\mathcal{M}_2$ is

$$\begin{aligned}\log\left[\frac{p(\mathbf{y}_{101:436} \mid \mathbf{y}_{1:100}, \mathcal{M}_1)}{p(\mathbf{y}_{101:436} \mid \mathbf{y}_{1:100}, \mathcal{M}_2)}\right] &= \log\left[\frac{\prod_{i=1}^{336} p(\mathbf{y}_{100+i} \mid \mathbf{y}_{1:100+i-1}, \mathcal{M}_1)}{\prod_{i=1}^{336} p(\mathbf{y}_{100+i} \mid \mathbf{y}_{1:100+i-1}, \mathcal{M}_2)}\right] \\ &= \sum_{i=1}^{336} \log p(\mathbf{y}_{100+i} \mid \mathbf{y}_{1:100+i-1}, \mathcal{M}_1) \\ &- \sum_{i=1}^{336} \log p(\mathbf{y}_{100+i} \mid \mathbf{y}_{1:100+i-1}, \mathcal{M}_2).\end{aligned}$$

An approximation is used for each log forecast distribution:

$$\begin{aligned} p(\mathbf{y}_{100+i} \mid \mathbf{y}_{1:100+i-1}, \mathcal{M}_i) &= \int p(\mathbf{y}_{100+i} \mid \theta_i, \mathbf{y}_{1:100+i-1}, \mathcal{M}_i) p(\theta_i \mid \mathbf{y}_{1:100+i-1}, \mathcal{M}_i) d\theta_i \\ &\approx N_{\theta_i}^{-1} \sum_{j=1}^{N_{\theta_i}} \hat{p}(\mathbf{y}_{100+i} \mid \theta^j, \mathbf{y}_{1:100+i-1}, \mathcal{M}_i) \end{aligned}$$

where each θ^j is drawn from the samples of the posterior distribution $p(\theta \mid y_{1:100})$, and particle filter approximations to the conditional likelihoods are used. Evaluating approximations to the conditional likelihoods was extensively discussed in (2) wherein using (2.5) was advocated. Also, instead of computing this difference of sums for every model pair, the summands at each point in time are plotted. This technique is more informative as it reveals exactly which point in time a model's forecast failed.

Another approximation to this score function is also plotted: the sequence $\log p(y_{t+1} \mid y_{1:t}, \hat{\theta}_{\text{post}})$. These are approximate likelihood calculations using a particle filter instantiated with the posterior mean, where the posterior is calculated conditioning on the first 100 observations. At the cost of ignoring parameter uncertainty, the benefit of this second method is computational ease. Multiple particle filters are not used here, so a serial program would finish much more quickly, or more computational effort could be placed on increasing the number of particles to increase the precision of the state estimation. A parallel program would not be necessary, here.

The first method mentioned above is implemented using 20 Auxiliary Particle Filters, each with 250 particles, and the second method is implemented using one particle filter with 5000 particles. These two score functions illustrate the shortcomings of this model in plot (4-2). Data arriving during the financial crisis yield low evaluations of the log forecasting densities. (4-2) also shows that during periods of high volatility, the mean of the filtering distribution for the log volatility increases, as expected. This happens even though the parameters were estimated on a low-volatility dataset.

As a second means of forecast evaluation, the standardized forecast errors were plotted. These standardized errors were calculated in two ways. Taken together, these two methods show when the model's forecast variances and covariances lose accuracy. Figure (A-5) plots the original returns, and the de-correlated returns. The de-correlated returns are obtained by pre-multiplying each return vector by the inverse of the Cholesky decomposition of the forecast's covariance matrix. The bottom row plots the cumulative sums of these errors to better highlight the sign of their means. It appears during the crisis period, the errors tend to be negative, and after the crisis period, they tend to be positive.

Figure (A-6) shows the out-of-sample prewhitened errors as well. Each time series is given its own plot window. Figure (A-7) shows the standardized errors. These standardized errors are obtained by dividing each return element by its estimated standard deviation. A comparison of these plots suggests that this model has difficulty forecasting the changing correlations and variances during crisis events.

Finally, figure (A-8) shows the cumulative sums of returns from a minimum variance investment strategy. This strategy chooses each week's weight vector to minimize the overall portfolio variance, subject to the constraint that the elements of this weight vector sum to 1. One can use Lagrange multipliers to show that this weight vector is proportional to $\Sigma_t^{-1} 1_{d_y}$, where Σ_t is the forecast's covariance matrix. This "backtest" ignores the effect of costs of trading such as commissions and slippage, assumes that shorting is allowed, and that nonintegral amounts of shares are available for purchase. To compare, the cumulative sums of returns from holding the "SPY" ETF are also plotted. SPY tracks the S&P500 index, which has each company weighted by its market capitalization.

When the single log volatility state increases, the future covariance matrix increases by a multiple of $\mathbf{BB}'$. It is true that the lack of adaptability might be mitigated by adding more factors; indeed, the future covariance matrix could change in more complicated ways, and this modification would be relatively easy to implement. However, this mitigation is "dangerous" in the sense that there is no guarantee that the in-sample parameter estimates for $\mathbf{B}$ would yield accurate forecasts on out-of-sample data. For example, if the estimates of another column of $\mathbf{B}$ corresponded with a factor that could be interpreted as "interest rate shocks," then there is no guarantee that future volatility dynamics would be driven by uncertainty over interest rates.

Alternative Models

These results motivate the need for a refined model, and indeed, there have been many adjustments to this model proposed in the literature already. Both [Pitt and Shephard, 1999b] and [Aguilar and West, 2000] assume that the covariance matrix of $\mathbf{v}_t$, in addition to the factors, is driven by exponentiated latent state processes as well. In this way, individual elements along the diagonal of the forecast covariance matrix are allowed to change over time. This model has the advantage that any number of stocks could experience an increase in their return variances.

One downside to this approach, however, is that this addition increases the model's state dimension by d_y, which might be substantial. This is undesirable because it increases the variance of (or the requisite number of particles for) the approximation of the filtering distribution. Also, it might be inaccurate to assume that these idiosyncratic variance components change in this manner, both independently of each other and following AR(1) dynamics.

[Chib et al., 2006] propose a related and much more flexible model. They add jumps to the observation equation and assume the idiosyncratic errors follow a student-t distribution. This shares the same problems as the models above, and in addition to this, it can have an enormous amount of parameters.

Another extension is given in [Lopes et al., 2006]. The authors allow elements of the $\mathbf{B}$ matrix to follow independent first-order autoregressions and the mean of the state process to be a "regime-switching" term. The hope is that this model will capture slowly changing factor loadings, and the apparent jumping nature of the market's volatility. This has the same computational drawback as the previously mentioned approach, and moreover, the authors only allow for jumps in the mean of the overall volatility. The author agrees that abrupt shocks are a necessary and an intuitively reasonable adjustment to include in the model, but there does not seem to be any support for the hypothesis that the factor loadings should follow a random walk.

The Markov-Switching Loadings Models

It can be seen after looking at (A-6) and (A-7) that the industries whose variances are particularly underestimated are very likely to be the industries most affected by the financial crisis. "XLY", "XLV" and "XLF" represent the consumer discretionary, healthcare and financial sectors, respectively. These plots suggest not only that the number of active factors is time-varying, but also that the makeup of pertinent factors is economically situational. Here, a novel factor stochastic volatility model that exhibits a time-varying number of factors with random loading matrices is presented. Two formulations are given: the first and second Markov-Switching Loadings Models (MSL1) and (MSL2). Each is designed to better distinguish between market-wide increases in volatility, and increases in volatility for random subsets of stocks.

MSL1

The following model will have its state broken into three components: x_{1t}, $\mathbf{x}_{2t}$ and $\mathbf{x}_{3t}$. Its last two sets of states are the log volatilities of two sets of equally sized factor vectors $\mathbf{f}_{1t}$ and $\mathbf{f}_{2t}$. The first vector will always be "active" in the sense that in the observation equation, it is multiplied by the same nonrandom loadings matrix. On the other hand, the second factor vector is occasionally "inactive." This factor vector is pre-multiplied by the same nonrandom loadings matrix, but this loadings matrix is in turn multiplied by a random matrix $\mathbf{D}_{x_{1t}}$. The resulting loadings matrix is a randomly sparsified version of the original loadings matrix $\mathbf{B}$. Conditioning on the event that $\mathbf{D}_{x_{1t}} = \mathbf{0}_{d_y \times d_y}$, the second factor vector can be omitted from the observation equation.

The observation equation can be written as follows:

$$\mathbf{y}_t = \mathbf{B}\mathbf{f}_{1t} + \mathbf{D}_{x_{1t}}\mathbf{B}\mathbf{f}_{2t} + \mathbf{v}_t. \tag{4.11}$$

$\{\mathbf{v}_t\}$ are again iid Gaussian noise with a diagonal covariance matrix $\mathbf{R}$. The random loadings matrix $\mathbf{D}_{x_{1t}}$ is a matrix-valued transformation of the first state component;

it is diagonal and can have up to K elements that are 1. More detail on x_{1t} is given in the paragraph after the next.

Both factor vectors will also have nonzero means. This addition is motivated by figure (A-5). By examining the cumulative sums of forecast errors, one is able to see that these errors, which correspond with the one market factor, tend to have a positive mean (at least in tranquil times). It is anticipated that the first group of factors will have positive means. The positive mean of a factor is often referred to as its "risk premium" in the financial literature. The second group of factors, because its members are only active during market turmoil, is anticipated to have a negative mean vector. The factors can be written as follows:

$$\mathbf{f}_t = \begin{bmatrix} \mathbf{f}_{1t} \\ \mathbf{f}_{2t} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\lambda}_1 \\ \boldsymbol{\lambda}_2 \end{bmatrix} + \begin{bmatrix} \exp(\text{diag}\{\mathbf{x}_{2t}\}) & \mathbf{0} \\ \mathbf{0} & \exp(\text{diag}\{\mathbf{x}_{3t}\}) \end{bmatrix} \begin{bmatrix} \mathbf{z}_{1t} \\ \mathbf{z}_{2t} \end{bmatrix}. \tag{4.12}$$

The dynamics for all but the first state processes are independent AR(1) models. The disturbances $\{\mathbf{w}_t\} = \{\mathbf{w}_{1t}, \mathbf{w}_{2t}\}$ follow iid Gaussian noise with the diagonal covariance matrix $\mathbf{Q} = \text{diag}(\mathbf{Q}_1, \mathbf{Q}_2) = \text{diag}(\sigma_1^2, \sigma_2^2, \dots, \sigma_{d_x-1}^2)$. The state transition equations can be written as

$$\begin{bmatrix} \mathbf{x}_{2t} \\ \mathbf{x}_{3t} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix} + \begin{bmatrix} \boldsymbol{\Phi}_1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Phi}_2 \end{bmatrix} \left(\begin{bmatrix} \mathbf{x}_{2t-1} \\ \mathbf{x}_{3t-1} \end{bmatrix} - \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix} \right) + \begin{bmatrix} \mathbf{w}_{1t} \\ \mathbf{w}_{2t} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{x}_{21} \\ \mathbf{x}_{31} \end{bmatrix} \sim \text{N}\left(\begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix}, \begin{bmatrix} \mathbf{Q}_1(\mathbf{I} - \boldsymbol{\Phi}_1^2)^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_2(\mathbf{I} - \boldsymbol{\Phi}_2^2)^{-1} \end{bmatrix} \right)$$

where $\boldsymbol{\mu} = (\boldsymbol{\mu}_1', \boldsymbol{\mu}_2')' = (\mu_1, \dots, \mu_{d_x-1})'$ is the vector of state means, and $\boldsymbol{\Phi} = \text{diag}(\boldsymbol{\Phi}_1, \boldsymbol{\Phi}_2) = \text{diag}(\phi_1, \dots, \phi_{d_x-1})$ is the diagonal transition matrix.

K represents the maximum number of sectors that can be affected by a "contained panic." The panic is contained in the sense that it does not apply to all of the stocks being modeled. Or, in other words, $K < d_y$. This forces the first factor to pick up on a market-wide increase in volatility while the second factor only affects sectors involved in a contained panic factor. Care should be taken to not choose this number to be too low. If this is done, then the second factor will fail to explain any crisis event that affects more than K sectors.

$S_K = \sum_{k=0}^{K} \binom{d_y}{k}$ will denote the total number of subsets, each representing a unique collection of instruments that participate in a contained panic. In the following analysis, these subsets are ordered lexicographically, and x_{1t}, a discrete Markov chain evolving on $\{1, 2, \ldots, S_K\}$, will represent the order index of a particular subset. For example, if $d_y = 3$ and $K = 2$, then the lexicographically ordered subsets are

$$\{(0,0,0), (0,0,1), (0,1,0), (0,1,1), (1,0,0), (1,0,1), (1,1,0)\}.$$

These correspond, respectively, with the values $1, 2, \ldots, 7$.

The transition matrix is specified by $p\mathbf{I} + (\mathbf{1}_{S_K}\mathbf{1}'_{S_K} - \mathbf{I})(1-p)/(S_K - 1)$. At the first time period, x_{11} is assumed to be distributed according to the discrete uniform distribution. This transition matrix conveniently only adds one additional parameter to be estimated. It assumes all states have the same probability of remaining in that state (p), and that from any state, there is an equal probability of going to any different state.

For example, consider the situation when $d_x = 3$, and $\mathbf{D}_{x_{1t}} = \text{diag}(1, 1, 0, \cdots, 0)$. Then, the observation equation can be written as

$$\mathbf{y}_t = \mathbf{B}f_{11t} + \mathbf{D}_{x_{1t}}\mathbf{B}f_{21t} + \mathbf{v}_t. \tag{4.13}$$

The primary factor f_{11t} represents a typical market factor, has a positive mean and exhibits stochastic volatility. The secondary factor also exhibits stochastic volatility, but has a negative mean. Conditioning on the first component of the state, this secondary factor will represent the first two stocks' propensity to decrease their means, increase their conditional variances, and increase their shared covariance. This can be seen by noticing that $E[\mathbf{D}_{x_{1t}}\mathbf{B}f_{21t} \mid \mathbf{x}_t] = \lambda_2(B_1, B_2, 0, \dots, 0)'$ and that

$$\text{Var}\left[\mathbf{D}_{x_{1t}}\mathbf{B}f_{21t} \mid \mathbf{x}_t\right] = e^{x_{31t}} \begin{bmatrix} B_1^2 & B_1B_2 & 0 & \cdots & 0 \\ B_2B_1 & B_2^2 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix}.$$

It is important to remind the reader that this model possesses the structure required for a Rao-Blackwellized particle filter [Liu et al., 2001], [Andrieu and Doucet, 2002], and [Nordh, 2014]. This will reduce the requisite number of particles of a particle filter, and thus increase the efficiency of the PMMH algorithms, as well as decrease the computational cost of the rolling forecast estimates.

MSL2

The second Markow-Switching Loadings model is similar to the first. It has the same autoregressive dynamics for all but the first state processes, and it features the same observation equation. MSL2, like MSL1, also features a state component representing which of the S_K vectors should be chosen for the second column of $\mathbf{B}$. The only difference is that this Markov chain x_{1t} possesses a different, hierarchical formulation.

This hierarchical formulation can be seen if one writes this as two components, i.e. by setting $x_{1t} = (u_t, v_t)'$. The first component u_t is a binary random variable representing whether or not there is any contained panic taking place in the market. Its transition matrix can be written as

$$\begin{bmatrix} p_1 & 1 - p_1 \\ 1 - p_2 & p_2 \end{bmatrix}. \tag{4.14}$$

The probability of remaining in a state of no contained panic is p_1, while the probability of remaining in a contained panic is p_2.

The second component v_t represents which grouping of stocks is affected by the contained panic. It takes values in $\{1, 2, \ldots, S_K\}$ which are numbers that correspond to the elements of the lexicographically ordered subsets of $\{0, 1\}^{d_y}$ which have at most K 1s with the rest 0s.

The next task is to describe the conditional transitions of v_t. A $S_K \times S_K$ transition matrix is needed for each value of u_t. In this particular case, two 130×130 matrices are needed: $p(v_t \mid v_{t-1}, u_t = 0)$ and $p(v_t \mid v_{t-1}, u_t = 1)$. Fortunately, these matrices contain no unknown parameters. This is possible after making the following economically reasonable assumptions.

First, it is assumed that if $u_t = 0$, then transitioning into any state other than the first is impossible. This corresponds with the interpetation of u_t. The resulting conditional transition matrix has no free parameters; it is

$$p(v_t \mid v_{t-1}, u_t = 0) = \begin{bmatrix} \mathbf{1}_{S_K} & \mathbf{0}_{S_K} & \cdots & \mathbf{0}_{S_K} \end{bmatrix}.$$

On the other hand, when $u_t = 1$, it is assumed that

1. transitioning to a tranquil state is impossible from anywhere,
2. coming from a tranquil state, each configuration is an equally likely arrival location, and
3. it is impossible to change from one panic configuration to another.

The first assumption coheres with the interpretation of u_t. The second assumption is reasonable for this dataset because, to the best of the author's knowledge, there is no economic reason why one sector is more likely than another to be involved in a contained panic. The third assumption is reasonable for the dataset under consideration as well; markets are "efficient" in the sense that they likely take less than a week to determine which sectors are affected by unfavorable news. The resulting conditional transition matrix is specified as follows:

$$p(v_t \mid v_{t-1}, u_t = 1) = \begin{bmatrix} 0 & \frac{1}{S_K-1} & \frac{1}{S_K-1} & \cdots & \frac{1}{S_K-1} \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}.$$

Because these two matrices do not contain any unknown parameters, this implies that MSL2 only has one more parameter than MSL1.

To complete the specification of MSL2, the distribution for the first state component at time 1 must be specified. Just like in the specification of MSL1, it is again assumed that x_{11} is distributed according to the chain's stationary distribution: $p(x_{11}) = p(v_1 \mid u_1)p(u_1)$. It is straightforward to verify that the stationary distribution is $p(u_1) = \left(\frac{1-p_2}{2-p_1-p_2}, \frac{1-p_1}{2-p_1-p_2}\right)$, while $p(v_1 \mid u_1 = 0) = (1, 0, \ldots, 0)'$ and $p(v_t \mid u_1 = 1) = (0, 1/(S_K - 1), 1/(S_K - 1), \ldots, 1/(S_K - 1))'$.

Estimation of MSL Models

This section describes an estimation routine for the parameters of the two MSL models when d_x is set to 3 and K is set to 2. The two MSL models suffer from the same nonidentifiability concerns as the inital model. Scaling, column-switching, and sign-switching all possibly yield alternative state space models with an identical likelihood functions. In this section, the same measures to deal with nonidentifiability are taken.

Estimation of MSL1

Many of the priors for the initial model are used again for the estimation of the MSL models. The priors for all but the first elements of $\mathbf{B}$ remain the same as the previous model's: a Gaussian distribution with mean $\mathbf{1}$ and covariance matrix $.125\mathbf{I}$.

The priors for the diagonal elements of $\mathbf{R}$ remain the same as well. The previous priors for the single ϕ are given to the two ϕ_is describing the dynamics of x_{2t} and x_{3t}, and the same technique is used to assign priors to the vectors $\boldsymbol{\mu}$ and $(\sigma_1^2, \sigma_2^2)'$.

The two factor means λ_1 and λ_2, after being made positive, are each given independent Half-Normal priors. That is $\lambda_1 \sim$ Half-Normal(10) and $-\lambda_2 \sim$ Half-Normal(20). Restricting $\lambda_2 < 0$ reinforces the interpretation that the second factor is a contained panic factor. One of the stylized features of financial returns is that they are "asymmetric," which describes the inverse relationship between return means, and return dispersion. This prior/likelihood combination is very much in agreement with this empirical observation.

Last, p is given a Beta(6,1) prior. This Beta prior is chosen to reflect the pre-existing knowledge about volatility clustering. Even though the term "volatility clustering" usually refers to the positivity of the AR(1) coefficents of the state process (in this case these are ϕ_1 and ϕ_2), it is also reasonable that the set describing *which* sectors are involved in a contain panic is likely to remain the same from one time period to the next.

The same PMMH algorithm (6) is used to draw 50,000 samples from the target posterior. This time, however, a Rao-Blackwellized particle filter is used to approximate likelihood evaluations. Every iteration runs 8 particle filters in parallel, each with 5 particles. The transformed parameters are proposed with a multivariate Gaussian random walk, and then transformed back into the quantities of interest. This multivariate Gaussian proposal distribution begins its adaptation of its covariance matrix at iteration 150, and finishes its adaptation at iteration 500. The initial parameters were taken from the last iteration of a trial-run of the algorithm. After a burn-in portion of the 500 samples is discarded, the posterior mean estimates are calculated. These estimates, along with their batch means standard errors, are printed below in tables (4.4), (4.5), (4.6) and (4.7). The program took 2.78 hours to run on an Intel Xeon Processor E3-1241 v3, and had an acceptance rate of 7.4%.

Both the initial model and MSL1 produce similar estimates for $\mathbf{B}$. Comparing the estimates of the elements of $\mathbf{R}$, the MSL model's estimates are always smaller than

Table 4.4: MSL **B** Estimates

	xlu beta	xlk beta	xlb beta	xlp beta	xly beta	xli beta	xlv beta	xlf beta
est	0.821	1.171	1.323	0.664	1.076	0.990	0.775	1.208
se	0.006	0.008	0.009	0.005	0.007	0.007	0.006	0.008

Table 4.5: MSL State Process Parameter Estimates

	phi1	phi2	mu1	mu2	sigma1	sigma2
est	0.424	0.396	0.743	0.668	0.309	0.370
se	0.023	0.019	0.016	0.022	0.011	0.013

those of the initial model. This is to be expected as the MSL model tries to explain more of the variability of the returns.

The MSL model's estimates of the parameters governing the AR(1) processes are similar to those of the initial model as well; even though there is an additional factor, the persistence parameters, the means, and the variance of the log volatility are all very close to the corresponding estimates of the previous model.

The estimate of the overall market risk premium parameter λ_1 is .302. This has the interpretation that, assuming there will be no contained panic in the next time period, the vector of expected log returns for the ETFs is $\lambda_1 \mathbf{B}$. In other words, each stock, when it is known that there will be no market turmoil, makes about 30 basis points every week. The mean of the panic factor, on the other hand, is estimated to be $-.525$. When stock i is assumed to be involved in a contained panic, its expected return is $\mathbf{B}_i(.302 - .525)$. This means that when a stock is experiencing a contained panic, it will lose on average 22 basis points per week. Last, p's estimate is .84, which means that the vector of the ETFs' "panic configuration" has a strong tendency to stay in the same state from one week to the next.

It bears reminding that these estimates were obtained using data prior to the financial crisis. In the next section, these estimates are used to instantiate particle filters to be used for online forecasting. Every week, on a rolling basis, predictions for the following week are generated, analyzed, and used to make investment decisions.

Table 4.6: MSL **R** Estimates

	xle rsd	xlu rsd	xlk rsd	xlb rsd	xlp rsd	xly rsd	xli rsd	xlv rsd	xlf rsd
est	6.315	2.158	1.258	1.591	0.626	0.435	0.550	0.669	0.859
se	0.045	0.015	0.013	0.017	0.005	0.007	0.005	0.007	0.010

Table 4.7: MSL λ_i and p_i Estimates $(i = 1, 2)$

	lambda1	lambda2	p
est	0.302	-0.525	0.840
se	0.008	0.015	0.004

Estimation of MSL2

To estimate the second Markov-Switching Loadings model, only three parameter groups were given the same priors from the previous section. The elements of the **B** vector were given the same independent Normal$(1, .125)$ priors, the elements of $\boldsymbol{\mu}$ were given the same Normal$(0, 3.16)$ priors, and λ_1 and $-\lambda_2$ were each given Half-Normal(3) priors. The uncertainty over what the other parameter values are was expressed with a choice of uninformative priors. ϕ_1 and ϕ_2 were given Uniform$(-1, 1)$ priors, not favoring any volatility persistence, but still ensuring stationarity of the log volatility processes. All variance parameters, σ_1^2, σ_2^2 and **R** were all given independent Inverse-Gamma$(.001, .001)$ priors. Last, p_1 and p_2 were given Uniform$(0, 1)$ priors.

The same algorithm used to estimate MSL1 is also used to estimate MSL2 (6), although it is tuned differently. 50,000 samples were drawn from the posterior, and at every iteration, 8 Rao-Blackwellized particle filters, each with 10 particles, were used to approximate the likelihood. A similar proposal distribution was used to change the parameters. After each parameter was transformed to make its support unrestricted, a multivariate Normal distribution was used to propose new values. This multivariate Gaussian proposal started its adaptation at iteration 150 and finished its adaptation at iteration 500. The starting parameter values were chosen from a trial run of the program. The program took 16.69 hours to run on an Intel Xeon Processor E3-1241 v3, and achieved an acceptance rate of 6.9%. The trace plots in figures (A-16), (A-17), (A-18), (A-19), (A-20), (A-21) and (A-22) show all the samples obtained; they do not discard any burn-in samples. The histograms and scatterplots, on the other hand, do discard the intial 500 samples. The posterior means, along with their standard error estimates are displayed in tables (4.8), (4.9), and (4.11).

Table 4.8: MSL2 **B** Estimates

	xlu beta	xlk beta	xlb beta	xlp beta	xly beta	xli beta	xlv beta	xlf beta
est	1.372	1.304	0.881	0.257	0.416	0.261	0.191	0.350
se	0.012	0.016	0.010	0.005	0.005	0.003	0.003	0.004

Table 4.9: MSL2 State Process Parameter Estimates

	phi1	phi2	mu1	mu2	sigmaSq1	sigmaSq2
est	0.388	0.930	1.024	0.370	0.016	1.561
se	0.009	0.003	0.018	0.040	0.002	0.059

Table 4.10: MSL2 **R** Estimates

	xle rsd	xlu rsd	xlk rsd	xlb rsd	xlp rsd	xly rsd	xli rsd	xlv rsd	xlf rsd
est	7.752	0.936	1.868	1.371	0.873	0.974	0.693	1.002	1.802
se	0.078	0.045	0.023	0.022	0.009	0.011	0.006	0.009	0.018

It is interesting to note the change in the parameter estimates related to the second factor. In MSL2, more structure was given to the transition matrix for x_{1t}. This was done by assuming that many one-step transitions were impossible. As a result, a contained panic is more extreme; these contained panics are associated with a larger decrease in the forecast's mean ($\hat{\lambda}_2 = -2.872$), they last for a shorter amount of time ($\hat{p}_2 = 0.063$), they have a log volatility that is more persistent ($\hat{\phi}_2 = 0.93$), and there is more uncertainty about this log volatility's value ($\hat{\sigma}_2^2 = 1.561$).

Also note that the posterior mean for σ_1^2 is quite close to 0. If in fact σ_1^2 was 0, this would simplify the model; it would lead to $x_{2t} = \mu_1$ for all t, causing $\{f_{1t}\}$ to be iid Normal random variables with mean λ_1 and variance $\exp(\mu_1)$. Conditioning on the event that $x_{1t} = 1$, the returns would be Normally distributed, suggesting that the fatness in the tails of weekly returns mostly comes from rare contained panic events.

Table 4.11: MSL2 λ_i and p_i Estimates ($i = 1, 2$)

	lambda1	lambda2	p1	p2
est	0.296	-2.872	0.955	0.063
se	0.007	0.035	0.002	0.002

Out-of-Sample Performance

This section compares the out-of-sample forecasting performance of all the models discussed. This is done in the same three ways that were used in (4.4.2): by evaluating score functions, examining forecast errors, and backtesting a minimum variance portfolio investment strategy.

Figure (A-23) shows the one-step-ahead log conditional likelihoods. These are computed averaging over 50 parameter draws from the time 100 posterior. Each of these parameter draws instantiated a Rao-Blackwellized particle filter with 250 particles. It can be seen that MSL1 and MSL2, by this measure, have superior forecasts for nearly the entire time window.

Before discussing the forecast error plots, formulas are given for the first and second moments of the MSL forecasts. Proofs of these facts are located in the appendix. As opposed to the initial model, neither MSL1 nor MSL2 have constant forecast means. It can be seen that

$$E[\mathbf{y}_{t+1} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}, \theta] = \lambda_1 \mathbf{B} + \lambda_2 \overline{\mathbf{D}}_{t+1}^{t} \mathbf{B} \tag{4.15}$$

where $\overline{\mathbf{D}}_{t+1}^{t} = E[\mathbf{D}_{x_{1,t+1}} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}, \theta]$. This is very much in agreement with stylized features of financial returns. If $x_{1,t+1}$ is expected to be in the state associated with no panic, returns are expected to be slightly positive ($\lambda_1 \mathbf{B}$). On the other hand, if $x_{1,t+1}$ is expected to be in a state that signifies a panic, the affected sectors will have a reduced expected return.

The conditional forecast's variance, $\mathrm{Var}\left(\mathbf{y}_{t+1} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}, \theta\right)$, is equal to

$$\begin{aligned}
&\exp\left(\mu_1 + \phi_1(x_{2,t} - \mu_1) + \sigma_1^2/2\right)\mathbf{B}\mathbf{B}' + \mathbf{R} \\
&+ \exp\left(\mu_2 + \phi_1(x_{3,t} - \mu_2) + \sigma_2^2/2\right) E\left(\mathbf{D}_{x_{1,t+1}}\mathbf{B}\mathbf{B}'\mathbf{D}'_{x_{1,t+1}} \mid x_{2t}, x_{3t}\right) \qquad (4.16) \\
&+ \lambda_2^2\left\{E\left(\mathbf{D}_{x_{1t+1}}\mathbf{B}\mathbf{B}'\mathbf{D}_{x_{1t+1}} \mid x_{2t}, x_{3t}\right) - \overline{\mathbf{D}}^t_{t+1}\mathbf{B}\mathbf{B}'\overline{\mathbf{D}}^t_{t+1}\right\}.
\end{aligned}$$

The first row is the familiar piece. It has the same form as the previous model's forecast's variance (4.5). The rest of the expression allows only some of the terms of the covariance matrix to change.

Obtaining expressions for the forecast mean and covariance matrix is done in the same approach used to obtain (4.8). To obtain an approximation for the forecast mean, $E[\mathbf{y}_{t+1} \mid \mathbf{y}_{1:t}]$, simply average over parameter draws and weighted state samples the values associated with (4.15). The forecast's covariance matrix involves a more cumbersome expression. The law of total variance states

$$\mathrm{Var}\left[\mathbf{y}_{t+1} \mid \mathbf{y}_{1:t}\right] = \mathrm{Var}\left[E\left(\mathbf{y}_{t+1} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}, \theta\right) \mid \mathbf{y}_{1:t}\right] + E\left[\mathrm{Var}\left(\mathbf{y}_{t+1} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}, \theta\right) \mid \mathbf{y}_{1:t}\right].$$

To obtain an approximation for the forecast covariance matrix, write the right hand side of this expression in terms of expectations, and carry out the same procedure as mentioned above.

To whiten/standardize the errors, the same procedure as in section (4.4.2) is followed, only the observations are demeaned first. Figure (A-24) shows the whitened errors for both models, and figure (A-25) shows the standardized errors for both models. Comparing these errors with figures (A-6) and (A-7), more closely account for the extreme observations, particularly for the financial sector ETF (XLF) during the crisis period.

Figure (A-26) plots the filter means of the second and third state components of each MSL model, the filtered state means of the intial model, and the filtered probabilities that each MSL model is in some contained panic configuration. The separation between the two log volatilities is much more extreme in the second MSL model. In this model, the first log volatility process hugs its stationary mean closely for much of the time. The second log volatility process is probably under-specified as these filtered values do not look stationary. The process appears to have a higher mean and variance during the crisis period. It is also worth considering whether to add jumps to this state process. Around the time of the 240th week, both models pick up a jump in the contained panic log volatility. MSL2's filtered mean is much more pronounced, however.

Finally, figure (A-27) shows the results of two minimum variance portfolio investment strategies using each of the three models. The first strategy sets the investment weights at every week equal to the vector that minimizes forecasted variance. However, these weights will often be associated with a negative expected return. The second, adjusted strategy simply sets these weights to the zero vector when this happens. It can be seen that all strategies yield higher terminal wealth than the passive buy-and-hold strategy. Also, it appears that MSL2 is generally "pessimistic" in the sense that quite often the expected returns associated with the minimum variance portfolio yields a negative expected return. Again, these backtests do not account for costs of trading, they assume that shorting is allowed, and they assume nonintegral amounts of shares may be purchased or sold.

Chapter 5

Summary

Overview

This document has described a complete, top-down approach for portfolio construction that allows the use of a very wide and general class of nonlinear and non-Gaussian state space models, it has suggested enhancements to particle Markov chain Monte Carlo algorithms as well as demonstrated their use on real-world datasets with large and complex models, and it has proposed a novel factor stochastic volatility model. In addition to these contributions, particle filtering and Markov chain Monte Carlo algorithms for state space models have been reviewed in detail. The combination of these two techniques, in the opinion of the author, constitute a strong foundation upon which to build further developments.

Appendix A

Appendix

Proofs

Proof of Lemma 1. This proof adapts the proof given in [Andrieu et al., 2010] for the simpler IMH algorithm.

The target distribution needs to be proportional to

$$p(\theta)\hat{p}(y_{1:T}|\theta)p(k'|u')\psi(u'|y_{1:T},\theta). \tag{A.1}$$

Note that $\hat{p}(y_{1:T}|\theta)$ is not a density; it is a functional of random variables. Its calculation requires $x_1^{b_1^k},\ldots,x_T^{b_T^k}$.

Let q_t refer to the proposal density of x_t, and r_t refer to the pmf of the ancestor indices. Define the joint density of all particle filter output, as well as the probability of picking path k from this output:

1. $\psi(u|y_{1:T},\theta)=\prod_{i=1}^N q_1(x_1^i)r_1(a_1^i|\mathbf{x}_1)\prod_{t=2}^{T-1}\prod_{j=1}^N q_t(x_t^j|x_{t-1}^{a_{t-1}^j},y_t)r_t(a_t^j|\mathbf{x}_t)q_T(x_T^j|x_{T-1}^{a_{T-1}^j},y_T)$

2. $p(k|u)=r_T(k|\mathbf{x}_T)=\tilde{w}_T(x_T^k)\propto\frac{g(y_T|x_T^k)f(x_T^k|x_{T-1}^{b_{T-1}^k})}{q_T(x_T^k|x_{T-1}^{b_{T-1}^k})}$.

The joint density is:

$$\begin{aligned}
p(k|u,\theta)\psi(u|y_{1:T},\theta) &= p(k|u,\theta)\prod_{i=1}^N q_1(x_1^i)r_1(a_1^i|\mathbf{x}_1)\prod_{t=2}^{T-1}\prod_{j=1}^N q_t(x_t^j|x_{t-1}^{a_{t-1}^j},y_t)r_t(a_t^j|\mathbf{x}_t)q_T(x_T^j|x_{T-1}^{a_{T-1}^j},y_T)\\
&=\prod_{i=1}^N q_1(x_1^i)\tilde{w}_1(x_1^{a_1^i})\prod_{t=2}^{T-1}\prod_{j=1}^N q_t(x_t^j|x_{t-1}^{a_{t-1}^j},y_t)\tilde{w}_t(x_t^{a_t^j})q_T(x_T^j|x_{T-1}^{a_{T-1}^j},y_T)\tilde{w}_T(x_T^k)\\
&=\prod_{i=1,i\neq b_1^k}^N q_1(x_1^i)\tilde{w}_1(x_1^{a_1^i})\prod_{t=2}^{T}\prod_{j=1,j\neq b_t^k}^N q_t(x_t^j|x_{t-1}^{a_{t-1}^j},y_t)\tilde{w}_t(x_t^{a_t^j})\\
&\quad\times q_1(x_1^{b_1^k})\tilde{w}_1(x_1^{b_1^k})\prod_{t=2}^T q_t(x_t^{b_t^k}|x_{t-1}^{b_{t-1}^k},y_t)\tilde{w}_t(x_t^{b_t^k}).
\end{aligned}$$

So

$$
\begin{aligned}
&\hat{p}(y_{1:T})p(k|u,\theta)\psi(u|y_{1:T},\theta) \\
&= \hat{p}(y_1)\prod_{t=2}^{T}\hat{p}(y_t|y_{1:t-1}) \\
&\qquad \times \prod_{i=1,i\neq b_1^k}^{N} q_1(x_1^i)\tilde{w}_1(x_1^{a_1^i})\prod_{t=2}^{T}\prod_{j=1,j\neq b_t^k}^{N} q_t(x_t^j|x_{t-1}^{a_{t-1}^j},y_t)\tilde{w}_t(x_t^{a_t^j}) \\
&\qquad \times q_1(x_1^{b_1^k})\tilde{w}_1(x_1^{b_1^k})\prod_{t=2}^{T} q_t(x_t^{b_t^k}|x_{t-1}^{b_{t-1}^k},y_t)\tilde{w}_t(x_t^{b_t^k}) \\
&= N^{-1}\sum_{i=1}^{N} w_1(x_1^i)\prod_{t=2}^{T} N^{-1}\sum_{i=1}^{N} w_t(x_t^i) \\
&\qquad \times \prod_{i=1,i\neq b_1^k}^{N} q_1(x_1^i)\tilde{w}_1(x_1^{a_1^i})\prod_{t=2}^{T-1}\prod_{j=1,j\neq b_t^k}^{N} q_t(x_t^j|x_{t-1}^{a_{t-1}^j},y_t)\tilde{w}_t(x_t^{a_t^j})q_T(x_T^j|x_{T-1}^{a_{T-1}^j},y_T) \\
&\qquad \times q_1(x_1^{b_1^k})\tilde{w}_1(x_1^{b_1^k})\prod_{t=2}^{T} q_t(x_t^{b_t^k}|x_{t-1}^{b_{t-1}^k},y_t)\tilde{w}_t(x_t^{b_t^k}) \\
&= N^{-T}\prod_{t=1}^{T}\sum_{i=1}^{N} w_t(x_t^i) \\
&\qquad \times \prod_{i=1,i\neq b_1^k}^{N} q_1(x_1^i)\tilde{w}_1(x_1^{a_1^i})\prod_{t=2}^{T-1}\prod_{j=1,j\neq b_t^k}^{N} q_t(x_t^j|x_{t-1}^{a_{t-1}^j},y_t)\tilde{w}_t(x_t^{a_t^j})q_T(x_T^j|x_{T-1}^{a_{T-1}^j},y_T) \\
&\qquad \times q_1(x_1^{b_1^k})\frac{w_1(x_1^{b_1^k})}{\sum_i w_1(x_1^i)}\prod_{t=2}^{T} q_t(x_t^{b_t^k}|x_{t-1}^{b_{t-1}^k},y_t)\frac{w_t(x_t^{b_t^k})}{\sum_i w_t(x_t^i)} \\
&= N^{-T}\prod_{i=1,i\neq b_1^k}^{N} q_1(x_1^i)\tilde{w}_1(x_1^{a_1^i})\prod_{t=2}^{T-1}\prod_{j=1,j\neq b_t^k}^{N} q_t(x_t^j|x_{t-1}^{a_{t-1}^j},y_t)\tilde{w}_t(x_t^{a_t^j})q_T(x_T^j|x_{T-1}^{a_{T-1}^j},y_T) \\
&\qquad \times q_1(x_1^{b_1^k})w_1(x_1^{b_1^k})\prod_{t=2}^{T} q_t(x_t^{b_t^k}|x_{t-1}^{b_{t-1}^k},y_t)w_t(x_t^{b_t^k}) \\
&= N^{-T}\prod_{i=1,i\neq b_1^k}^{N} q_1(x_1^i)\tilde{w}_1(x_1^{a_1^i})\prod_{t=2}^{T-1}\prod_{j=1,j\neq b_t^k}^{N} q_t(x_t^j|x_{t-1}^{a_{t-1}^j},y_t)\tilde{w}_t(x_t^{a_t^j})q_T(x_T^j|x_{T-1}^{a_{T-1}^j},y_T) \\
&\qquad \times q_1(x_1^{b_1^k})\frac{g(y_1|x_1^{b_1^k})f_1(x_1^{b_1^k})}{q_1(x_1^{b_1^k})}\prod_{t=2}^{T} q_t(x_t^{b_t^k}|x_{t-1}^{b_{t-1}^k},y_t)\frac{g(y_t|x_t^{b_t^k})f(x_t^{b_t^k}|x_{t-1}^{b_{t-1}^k})}{q_t(x_t^{b_t^k}|x_{t-1}^{b_{t-1}^k},y_t)}
\end{aligned}
$$

$$
\begin{aligned}
&= N^{-T} \prod_{i=1, i\neq b_1^k}^{N} q_1(x_1^i)\tilde{w}_1(x_1^{a_1^i}) \prod_{t=2}^{T-1} \prod_{j=1, j\neq b_t^k}^{N} q_t(x_t^j | x_{t-1}^{a_{t-1}^j}, y_t)\tilde{w}_t(x_t^{a_t^j}) q_T(x_T^j | x_{T-1}^{a_{T-1}^j}, y_T) \\
&\quad \times g(y_1|x_1^{b_1^k}) f_1(x_1^{b_1^k}) \prod_{t=2}^{T} g(y_t|x_t^{b_t^k}) f(x_t^{b_t^k}|x_{t-1}^{b_{t-1}^k}) \\
&= N^{-T} \prod_{i=1, i\neq b_1^k}^{N} q_1(x_1^i)\tilde{w}_1(x_1^{a_1^i}) \prod_{t=2}^{T-1} \prod_{j=1, j\neq b_t^k}^{N} q_t(x_t^j | x_{t-1}^{a_{t-1}^j}, y_t)\tilde{w}_t(x_t^{a_t^j}) q_T(x_T^j | x_{T-1}^{a_{T-1}^j}, y_T) \\
&\quad \times p(y_{1:T}, x_{1:T}^k \mid \theta).
\end{aligned}
$$

Finish by integrating with respect to $u \setminus x_{1:T}^k$.

$\square$

Proof of lemma 2. This proof uses notation that is similar to the above. However, we need to rewrite the resampling pmfs and the likelihood estimate a little differently. They are:

1. $r(a_1^i \mid \mathbf{x}_1) = \frac{p(y_1|x_{2,1}^i)f(x_{2,1}^i)}{q_1(x_{2,1}^i|y_1)} \left[\sum_k \frac{p(y_1|x_{2,1}^k)f(x_{2,1}^k)}{q_1(x_{2,1}^k|y_1)}\right]^{-1}$

2. $r(a_t^i \mid \mathbf{x}_{t-1:t}, \mathbf{a}_{t-1}) = \frac{f(X_{2,t}^i|x_{2,t-1}^{a_{t-1}^i})p(y_t|y_{1:t-1},X_{2,1:t}^i)}{q_t(X_{2,t}^i|x_{2,t-1}^{a_{t-1}^i},y_t)} \left[\sum_k \frac{f(X_{2,t}^k|x_{2,t-1}^{a_{t-1}^k})p(y_t|y_{1:t-1},X_{2,1:t}^k)}{q_t(X_{2,t}^k|x_{2,t-1}^{a_{t-1}^k},y_t)}\right]^{-1}$

3. $\psi(u|y_{1:T},\theta) = \prod_{i=1}^N q_1(x_{2,1}^i \mid y_1) r(a_1^i|\mathbf{x}_1) \prod_{t=2}^{T-1}\prod_{j=1}^N q_t(x_{2,t}^j|x_{2,t-1}^{a_{t-1}^j}, y_t) r(a_t^j|\mathbf{x}_{t-1:t}, \mathbf{a}_{t-1}) q_T(x_{2,T}^j \mid x_{2,T-1}^{a_{T-1}^j}, y_T)$

4. $\hat{p}(y_{1:T} \mid \theta) = \left(N^{-1}\sum_{i=1}^N \frac{p(y_1|x_{2,1}^i)f(x_{2,1}^i)}{q_1(x_{2,1}^i|y_1)}\right) \prod_{t=2}^T \left(N^{-1}\sum_{i=1}^N \frac{f(X_{2,t}^i|x_{2,t-1}^i)p(y_t|y_{1:t-1},X_{2,1:t}^i)}{q_t(X_{2,t}^i|x_{2,t-1}^i,y_t)}\right)$.

Our goal is to show the the likelihood estimator is unbiased for the RBPF, or in other words, that

$$\int \hat{p}(y_{1:T} \mid \theta)\psi(u \mid y_{1:T}, \theta) du = p(y_{1:T} \mid \theta). \tag{A.2}$$

Because our functional is a product, our general proof strategy is to iterate the expectations. For three random variables x_1, x_2, x_3, this would look like

$$\begin{aligned} E[x_1x_2x_3] &= E[x_1E(x_2x_3 \mid x_1)] \\ &= E[x_1E(x_2E\{x_3 \mid x_2, x_1\} \mid x_1)]. \end{aligned}$$

Denote $m_{T-1} = E[\hat{p}(y_T \mid y_{1:T-1}) \mid \mathbf{x}_{1:T-1}, \mathbf{a}_{1:T-1}]$, and for $t < T-1$

$$m_t = E[\hat{p}(y_{t+1} \mid y_{1:t})E(m_{t+1} \mid \mathbf{x}_{1:t+1}) \mid \mathbf{x}_{1:t}, \mathbf{a}_{1:t}].$$

From the definition of the likelihood approximation above:

$$E\left[\hat{p}(y_1)\prod_{t=1}^{N}\hat{p}(y_{t+1} \mid y_{1:t})\right] = E\left[\hat{p}(y_1)E(m_1 \mid \mathbf{x}_1)\right].$$

Dropping dependence on theta,

$$\begin{aligned} m_{T-1} &= E[\hat{p}(y_T \mid y_{1:T-1}) \mid \mathbf{x}_{1:T-1}, \mathbf{a}_{1:T-1}] && \text{(defn.)} \\ &= E\left[\frac{f(x^i_{2,T} \mid x^{a^i_{T-1}}_{2,T-1})p(y_T \mid y_{1:T-1}, x^i_{2,1:T})}{q_T(x^i_{2,T} \mid x^{a^i_{T-1}}_{2,T-1}, y_T)}\middle|\mathbf{x}_{1:T-1}, \mathbf{a}_{1:T-1}\right] && \text{(linearity)} \\ &= \int f(x^i_{2,T} \mid x^{a^i_{T-1}}_{2,T-1})p(y_T \mid y_{1:T-1}, x^i_{2,1:T})dx^i_{2,T} && \text{(cancelling } q) \\ &= \int p(x^i_{2,T} \mid x^{a^i_{T-1}}_{2,1:T-1}, y_{1:T-1})p(y_T \mid y_{1:T-1}, x^i_{2,1:T})dx^i_{2,T} && \text{(cndtnl indep.)} \\ &= \int \frac{p(x^i_{2,1:T}, y_{1:T-1})}{p(x^{a^i_{T-1}}_{2,1:T-1}, y_{1:T-1})}\frac{p(y_{1:T}, x^i_{2,1:T})}{p(y_{1:T-1}, x^i_{2,1:T})}dx_{2,T} \\ &= \int \frac{p(y_{1:T}, X^i_{2,1:T})}{p(x^{a^i_{T-1}}_{2,1:T-1}, y_{1:T-1})}dx_{2,T} && \text{(notation)} \\ &= \int p(y_T, x_{2,T} \mid y_{1:T-1}, x^{a^i_{T-1}}_{2,1:T-1})dx_{2,T} \\ &= p(y_T \mid y_{1:T-1}, x^{a^i_{T-1}}_{2,1:T-1}). \end{aligned}$$

Then

$$E[m_{T-1} \mid \mathbf{x}_{T-1}] = \frac{\sum_i p(y_T \mid y_{1:T-1}, x^i_{2,1:T-1}) \frac{f(X^i_{2,T-1}|x^{a^i_{T-2}}_{2,T-2}) p(y_{T-1}|y_{1:T-2}, x^i_{2,1:T-1})}{q_{T-1}(x^i_{2,T-1}|x^{a^i_{T-2}}_{2,T-2}, y_{T-1})}}{\sum_j \frac{f(x^j_{2,T-1}|x^{a^j_{T-2}}_{2,T-2}) p(y_{T-1}|y_{1:T-2}, x^j_{2,1:T-1})}{q_{T-1}(x^j_{2,T-1}|x^{a^j_{T-2}}_{2,T-2}, y_{T-1})}}.$$

So

$$\begin{aligned}
m_{T-2} &= E[\hat{p}(y_{T-1} \mid y_{1:T-2}) E\left\{m_{T-1} \mid \mathbf{x}_{T-1}\right\} \mid \mathbf{x}_{1:T-2}, \mathbf{a}_{1:T-2}] \\
&= E\left[\frac{1}{N}\sum_i p(y_T \mid y_{1:T-1}, x^i_{2,1:T-1}) \frac{f(x^i_{2,T-1} \mid x^{a^i_{T-2}}_{2,T-2}) p(y_{T-1}|y_{1:T-2}, x^i_{2,1:T-1})}{q_{T-1}(X^i_{2,T-1} \mid x^{a^i_{T-2}}_{2,T-2}, y_{T-1})}\middle| \mathbf{x}_{1:T-2}, \mathbf{a}_{1:T-2}\right] \\
&= E\left[p(y_T \mid y_{1:T-1}, x^i_{2,1:T-1}) \frac{f(x^i_{2,T-1} \mid x^{a^i_{T-2}}_{2,T-2}) p(y_{T-1}|y_{1:T-2}, X^i_{2,1:T-1})}{q_{T-1}(x^i_{2,T-1} \mid x^{a^i_{T-2}}_{2,T-2}, y_{T-1})}\middle| \mathbf{x}_{T-2}, \mathbf{a}_{T-2}\right] \\
&= \int p(y_T \mid y_{1:T-1}, x^i_{2,1:T-1}) f(x^i_{2,T-1} \mid x^{a^i_{T-2}}_{2,T-2}) p(y_{T-1}|y_{1:T-2}, x^i_{2,1:T-1}) dx_{T-1} \\
&= \int p(y_T \mid y_{1:T-1}, x^i_{2,1:T-1}) p(x^i_{2,T-1} \mid x^{a^i_{T-2}}_{2,1:T-2}, y_{1:T-2}) p(y_{T-1}|y_{1:T-2}, x^i_{2,1:T-1}) dx_{T-1} \\
&= \int \frac{p(y_{1:T}, x^i_{2,1:T-1})}{p(y_{1:T-1}, x^i_{2,1:T-1})} \frac{p(x^i_{2,1:T-1}, y_{1:T-2})}{p(x^{a^i_{T-2}}_{2,1:T-2}, y_{1:T-2})} \frac{p(y_{1:T-1}, X^i_{2,1:T-1})}{p(y_{1:T-2}, x^i_{1:T-1})} dx_{T-1} \\
&= \int \frac{p(y_{1:T}, x^i_{2,1:T-1})}{p(x^{a^i_{T-2}}_{2,1:T-2}, y_{1:T-2})} dx_{T-1} \\
&= p(y_{T-1:T} \mid y_{1:T-2}, x^i_{2,1:T-2}).
\end{aligned}$$

Here's the inductive step. Assume that for $T-2 \geq t \geq 1$

$$m_t = p(y_{t+1:T} \mid y_{1:t}, x^{a^i_t}_{2,1:t}).$$

We want to show that

$$m_{t-1} = p(y_{t:T} \mid y_{1:t-1}, x^{a^i_{t-1}}_{2,1:t-1}).$$

First,

$$E[m_t \mid \mathbf{x}_t] = \frac{\sum_i p(y_{t+1:T} \mid y_{1:t}, x^i_{2,1:t}) \frac{f(x^i_{2,t} | x^{a^i_{t-1}}_{2,t-1}) p(y_t | y_{1:t-1}, x^i_{2,1:t})}{q_t(x^i_{2,t} | x^{a^i_{t-1}}_{2,t-1}, y_t)}}{\sum_j \frac{f(x^j_{2,t} | x^{a^j_{t-1}}_{2,t-1}) p(y_t | y_{1:t-1}, x^j_{2,1:t})}{q_t(x^j_{2,t} | x^{a^j_{t-1}}_{2,t-1}, y_t)}}.$$

Then we plug that in to obtain

$$\begin{aligned}
m_{t-1} &= E[\hat{p}(y_t \mid y_{1:t-1}) E\left\{m_t \mid \mathbf{x}_t\right\} \mid \mathbf{x}_{t-1}, \mathbf{a}_{t-1}] \\
&= E\left[\frac{1}{N}\sum_i p(y_{t+1:T} \mid y_{1:t}, x^i_{2,1:t}) \frac{f(x^i_{2,t} \mid x^{a^i_{t-1}}_{2,t-1}) p(y_t | y_{1:t-1}, x^i_{2,1:t})}{q_t(x^i_{2,t} \mid x^{a^i_{t-1}}_{2,t-1}, y_t)} \middle| \mathbf{x}_{t-1}, \mathbf{a}_{t-1}\right] \\
&= E\left[p(y_{t+1:T} \mid y_{1:t}, x^i_{2,1:t}) \frac{f(x^i_{2,t} \mid x^{a^i_{t-1}}_{2,t-1}) p(y_t | y_{1:t-1}, x^i_{2,1:t})}{q_t(x^i_{2,t} \mid x^{a^i_{t-1}}_{2,t-1}, y_t)} \middle| \mathbf{x}_{t-1}, \mathbf{a}_{t-1}\right] \\
&= \int p(y_{t+1:T} \mid y_{1:t}, x^i_{2,1:t}) f(x^i_{2,t} \mid x^{a^i_{t-1}}_{2,t-1}) p(y_t | y_{1:t-1}, x^i_{2,1:t}) dx^i_{2,t} \\
&= \int \frac{p(y_{t+1:T}, y_{1:t}, x^i_{2,1:t})}{p(y_{1:t}, x^i_{2,1:t})} \frac{p(x^i_{2,1:t}, y_{1:t-1})}{p(x_{2,1:t-1}, y_{1:t-1})} \frac{p(y_{1:t}, x^i_{2,1:t})}{p(y_{1:t-1}, x^i_{2,1:t})} dx^i_{2,t} \\
&= p(y_{t:T} \mid y_{1:t-1}, x^{a^i_{t-1}}_{2,1:t-1}),
\end{aligned}$$

which means

$$
\begin{aligned}
E\left[\hat{p}(y_1)\prod_{t=1}^{N}\hat{p}(y_{t+1}\mid y_{1:t})\right] &= E\left[\hat{p}(y_1)E[m_1\mid \mathbf{x}_1]\right.\\
&= E\left[\left(N^{-1}\sum_{i=1}^{N}\frac{p(y_1|x_{2,1}^i)f(x_{2,1}^i)}{q_1(x_{2,1}^i|y_1)}\right)\left(\frac{\sum_i p(y_{2:T}\mid y_1, x_1^i)\frac{p(y_1|x_{2,1}^i)f(x_{2,1}^i)}{q_1(x_{2,1}^i|y_1)}}{\sum_{i=1}^{N}\frac{p(y_1|x_{2,1}^i)f(x_{2,1}^i)}{q_1(x_{2,1}^i|y_1)}}\right)\right]\\
&= E\left[\frac{1}{N}\sum_i p(y_{2:T}\mid y_1, x_1^i)\frac{p(y_1|x_{2,1}^i)f(x_{2,1}^i)}{q_1(x_{2,1}^i|y_1)}\right]\\
&= E\left[p(y_{2:T}\mid y_1, x_1^i)\frac{p(y_1|x_{2,1})f(x_{2,1})}{q_1(x_{2,1}\mid y_1)}\right]\\
&= \iint p(y_{2:T}\mid y_1, x_1^i)p(y_1|x_{2,1}^i)f(x_{2,1}^i)dx_1dx_2\\
&= p(y_{1:T}).
\end{aligned}
$$

□

Proof of Lemma 3. The matrix $\boldsymbol{P}$ is diagonal with positive diagonal elements if and only if there exists a diagonal matrix $\log(\boldsymbol{P}^{-1})$, where $\log(\boldsymbol{P}^{-1})$ is defined uniquely as the logarithm of each of the diagonal elements of the inverse matrix. This allows us to write

$$
\begin{aligned}
\boldsymbol{P}^{-1}\exp\left[\operatorname{diag}(x_t)/2\right] &= \exp\left[\log(\boldsymbol{P}^{-1})\right]\exp\left[\operatorname{diag}(x_t)/2\right]\\
&= \exp\left[\left\{2\log(\boldsymbol{P}^{-1}) + \operatorname{diag}(x_t)\right\}/2\right] \qquad (*)\\
&= \exp\left[\operatorname{diag}(u + x_t)/2\right].
\end{aligned}
$$

Note that $(*)$ uses a property of matrix exponentials that doesn't always hold. If the matrices commute, however, then the product of these matrix exponentials equals the exponential of the sum, and it is always true that $2\log(\boldsymbol{P}^{-1})$ commutes with $\operatorname{diag}(x_t)$, as they are both diagonal.

Clearly, the process $u + x_t$ is a vector autoregressive process of the same form but with new mean $\mu + u$. □

Proof of Lemma 4. Let $\mathbf{P}$ be the permutation that encodes the permutation $\pi : i \mapsto j$. Because every permutation matrix is orthogonal,

$$\boldsymbol{B}\text{Var}(\mathbf{f}_t \mid x_t)\boldsymbol{B}' = \boldsymbol{B}\boldsymbol{P}'\boldsymbol{P}\text{Var}(\mathbf{f}_t|x_t)\boldsymbol{P}'(\boldsymbol{B}\boldsymbol{P}')'.$$

Pre-multiplying both sides of equation 4.1 by $\mathbf{P}$ yields a system of state transition equations corresponding to a vector autoregressive progress with parameters $\mathbf{P}\boldsymbol{\mu}$, $\mathbf{P}\boldsymbol{\Phi}\mathbf{P}'$, $\mathbf{P}\boldsymbol{Q}\mathbf{P}'$. Also, because it is true that for any diagonal matrix $\mathbf{D}$, $\mathbf{PDP}'$ will be the diagonal matrix with D_{ii} in the jjth spot, we have

$$\text{Var}(\widetilde{\mathbf{f}}_t|x_t) = \boldsymbol{P}\text{Var}(\mathbf{f}_t|x_t)\boldsymbol{P}',$$

where $\widetilde{\mathbf{f}}_t = \exp\left[\text{diag}(\mathbf{P}\mathbf{x}_t)/2\right]\mathbf{z}_t$. □

Proof of Lemma 5. This follows from the diagonality of $\mathbf{S}$ and $\mathbf{D} = \text{diag}(e^{x_{1,t}}, e^{x_{2,t}})$.

$$\begin{aligned}\text{Var}(\mathbf{B}\mathbf{f}_t \mid \mathbf{x}_t) &= \mathbf{BDB}' \\ &= \mathbf{BSDSB}' \\ &= \text{Var}(\mathbf{BS}\mathbf{f}_t \mid \mathbf{x}_t).\end{aligned}$$

□

Deriving MSL Mean Forecasts

Recall that a particle filter can approximate expectations with respect to the filtering distribution. In the case of a Rao-Blackwellized particle filter, it can approximate expectations with respect to the distribution $p(\mathbf{x}_{2t}, \mathbf{x}_{3t} \mid \mathbf{y}_{1:t}, \theta)$. The general strategy here is to iterate expectations and end up with a function of $\mathbf{x}_{2t}, \mathbf{x}_{3t}$.

We temporarily drop from the notation the dependence on the parameter values. Here is the first step:

$$
\begin{aligned}
&E[\mathbf{y}_{t+1} \mid \mathbf{x}_{t+1}] \\
&= \mathbf{B}E[\mathbf{f}_{1,t+1} \mid \mathbf{x}_{t+1}] + \mathbf{D}_{x_{1,t+1}}\mathbf{B}E[\mathbf{f}_{2,t+1} \mid \mathbf{x}_{t+1}] \qquad (4.11) \\
&= \mathbf{B}E[\boldsymbol{\lambda}_1 + \exp(\text{diag}\{\mathbf{x}_{2,t+1}\})\mathbf{z}_{1,t+1} \mid \mathbf{x}_{t+1}] + \mathbf{D}_{x_{1,t+1}}\mathbf{B}E[\boldsymbol{\lambda}_2 + \exp(\text{diag}\{\mathbf{x}_{3,t+1}\})\mathbf{z}_{2,t+1} \mid \mathbf{x}_{t+1}] \qquad (4.12) \\
&= \mathbf{B}\boldsymbol{\lambda}_1 + \mathbf{D}_{x_{1,t+1}}\mathbf{B}\boldsymbol{\lambda}_2.
\end{aligned}
$$

Here is step 2:

$$
\begin{aligned}
E[\mathbf{y}_{t+1} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}] &= E[E\left(\mathbf{y}_{t+1} \mid \mathbf{x}_{t+1}\right) \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}] \\
&= E[\mathbf{B}\boldsymbol{\lambda}_1 + \mathbf{D}_{x_{1,t+1}}\mathbf{B}\boldsymbol{\lambda}_2 \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}] \\
&= \mathbf{B}\boldsymbol{\lambda}_1 + \overline{\mathbf{D}}^t_{t+1}\mathbf{B}\boldsymbol{\lambda}_2,
\end{aligned}
$$

where $\overline{\mathbf{D}}^t_{t+1} = E[\mathbf{D}_{x_{1,t+1}} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}]$ is an expectation taken with respect ot $p(x_{1,t+1} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t})$.

Deriving MSL Covariance Matrix Forecasts

Again, we temporarily drop from the notation the dependence on the parameter values. Here is step 1:

$$
\begin{aligned}
&\text{Var}[\mathbf{y}_{t+1} \mid \mathbf{x}_{t+1}] \\
&= \text{Var}[\mathbf{B}\mathbf{f}_{1,t+1} + \mathbf{D}_{x_{1,t+1}}\mathbf{B}\mathbf{f}_{2,t+1} + \mathbf{v}_{t+1} \mid \mathbf{x}_{t+1}] \\
&= \mathbf{B}\mathbf{E}_{1,t+1}\mathbf{B}' + \mathbf{D}_{x_{1,t+1}}\mathbf{B}\mathbf{E}_{2,t+1}\mathbf{B}'\mathbf{D}'_{x_{1t}} + \mathbf{R}
\end{aligned}
$$

where $\text{Var}[\mathbf{f}_{1,t+1} \mid \mathbf{x}_{t+1}] = \mathbf{E}_{1,t+1} = \text{diag}(e^{x_{2,t+1}}, \ldots, e^{x_{(d_x+1)/2,t+1}})$ and $\text{Var}[\mathbf{f}_{2(t+1)} \mid \mathbf{x}_{t+1}] = \mathbf{E}_{2,t+1} = \text{diag}(e^{x_{(d_x+3)/2,t+1}}, \ldots, e^{x_{(d_x,t+1}})$

To complete step 2, use the following fact:

$$
\begin{aligned}
&\overline{\mathbf{E}}^{t}_{1,t+1}\\
&= E(\mathbf{E}_{1,t+1} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t})\\
&= E\left(\begin{bmatrix} \exp(x_{2,t+1}) & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \exp(x_{(d_x+1)/2,t+1}) \end{bmatrix} \middle| \mathbf{x}_{2t}, \mathbf{x}_{3t} \right)\\
&= E\left(\begin{bmatrix} e^{\mu_1+\phi_1(x_{2,t}-\mu_1)+w_{1,t}} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & e^{\mu_{(d_x-1)/2}+\phi_{(d_x-1)/2}(x_{(d_x+1)/2,t}-\mu_{(d_x-1)/2})+w_{(d_x-1)/2,t}} \end{bmatrix} \middle| \mathbf{x}_{2t}, \mathbf{x}_{3t} \right)\\
&= \begin{bmatrix} e^{\mu_1+\phi_1(x_{2,t}-\mu_1)+\sigma_1^2/2} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & e^{\mu_{(d_x-1)/2}+\phi_{(d_x-1)/2}(x_{(d_x+1)/2,t}-\mu_{(d_x-1)/2})+\sigma^2_{(d_x+1)/2}/2} \end{bmatrix}.
\end{aligned}
$$

Similary $\overline{\mathbf{E}}^{t}_{2,t+1}$ equals

$$
\begin{bmatrix} e^{\mu_{(d_x+1)/2}+\phi_{(d_x+1)/2}(x_{(d_x+3)/2,t}-\mu_{(d_x+1)/2})+\sigma^2_{(d_x+1)/2}/2} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & e^{\mu_{d_x-1}+\phi_{d_x-1}(x_{d_x,t}-\mu_{d_x-1})+\sigma^2_{1,d_x-1}/2}. \end{bmatrix}
$$

So

$$
\begin{aligned}
&E\left(\mathrm{Var}[\mathbf{y}_{t+1} \mid \mathbf{x}_{t+1}] \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right)\\
&= E\left(\mathbf{B}\mathbf{E}_{1,t+1}\mathbf{B}' + \mathbf{D}_{x_{1,t+1}}\mathbf{B}\mathbf{E}_{2,t+1}\mathbf{B}'\mathbf{D}'_{x_{1,t+1}} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right) + \mathbf{R}\\
&= \mathbf{B}\overline{\mathbf{E}}^{t}_{1,t+1}\mathbf{B}' + E\left(\mathbf{D}_{x_{1,t+1}}\mathbf{B}\mathbf{E}_{2,t+1}\mathbf{B}'\mathbf{D}'_{x_{1,t+1}} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right) + \mathbf{R}\\
&= \mathbf{B}\overline{\mathbf{E}}^{t}_{1,t+1}\mathbf{B}' + E\left\{\mathbf{D}_{x_{1,t+1}} E\left(\mathbf{B}\mathbf{E}_{2,t+1}\mathbf{B}' \mid x_{1,t+1}, \mathbf{x}_{2t}, \mathbf{x}_{3t}\right)\mathbf{D}'_{x_{1,t+1}} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right\} + \mathbf{R}\\
&= \mathbf{B}\overline{\mathbf{E}}^{t}_{1,t+1}\mathbf{B}' + E\left\{\mathbf{D}_{x_{1,t+1}}\mathbf{B}\overline{\mathbf{E}}^{t}_{2,t+1}\mathbf{B}'\mathbf{D}'_{x_{1,t+1}} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right\} + \mathbf{R}
\end{aligned}
$$

and

$$
\begin{aligned}
&\operatorname{Var}\left(E\left[\mathbf{y}_{t+1} \mid \mathbf{x}_{t+1}\right] \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right) \\
&= \operatorname{Var}\left(\mathbf{B}\boldsymbol{\lambda}_1 + \mathbf{D}_{x_{1,t+1}}\mathbf{B}\boldsymbol{\lambda}_2 \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right) \\
&= \operatorname{Var}\left(\mathbf{D}_{x_{1,t+1}}\mathbf{B}\boldsymbol{\lambda}_2 \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right) \\
&= E\left(\mathbf{D}_{x_{1,t+1}}\mathbf{B}\boldsymbol{\lambda}_2\boldsymbol{\lambda}_2'\mathbf{B}'\mathbf{D}_{x_{1,t+1}}' \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right) - E\left(\mathbf{D}_{x_{1,t+1}}\mathbf{B}\boldsymbol{\lambda}_2 \mid \mathbf{x}_t\right) E\left(\mathbf{D}_{x_{1,t+1}}\mathbf{B}\boldsymbol{\lambda}_2 \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right)' \\
&= E\left(\mathbf{D}_{x_{1,t+1}}\mathbf{B}\boldsymbol{\lambda}_2\boldsymbol{\lambda}_2'\mathbf{B}'\mathbf{D}_{x_{1,t+1}}' \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right) - \overline{\mathbf{D}}_{t+1}^{t}\mathbf{B}\boldsymbol{\lambda}_2\boldsymbol{\lambda}_2'\mathbf{B}'\overline{\mathbf{D}}_{t+1}^{t\prime}
\end{aligned}
$$

Put these two together and $\operatorname{Var}\left(\mathbf{y}_{t+1} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right)$ equals

$$
\begin{aligned}
&\mathbf{B}\overline{\mathbf{E}}_{1,t+1}^{t}\mathbf{B}' + E\left\{\mathbf{D}_{x_{1,t+1}}\mathbf{B}\overline{\mathbf{E}}_{2,t+1}^{t}\mathbf{B}'\mathbf{D}_{x_{1,t+1}}' \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right\} + \mathbf{R} \\
&+ E\left(\mathbf{D}_{x_{1,t+1}}\mathbf{B}\boldsymbol{\lambda}_2\boldsymbol{\lambda}_2'\mathbf{B}'\mathbf{D}_{x_{1,t+1}} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right) - \overline{\mathbf{D}}_{t+1}^{t}\mathbf{B}\boldsymbol{\lambda}_2\boldsymbol{\lambda}_2'\mathbf{B}'\overline{\mathbf{D}}_{t+1}^{t\prime}.
\end{aligned}
\tag{A.3}
$$

Things simplify in the special case of MSL1 and MSL2 used in the applied section. In this case $\operatorname{Var}\left(\mathbf{y}_{t+1} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}\right)$ equals

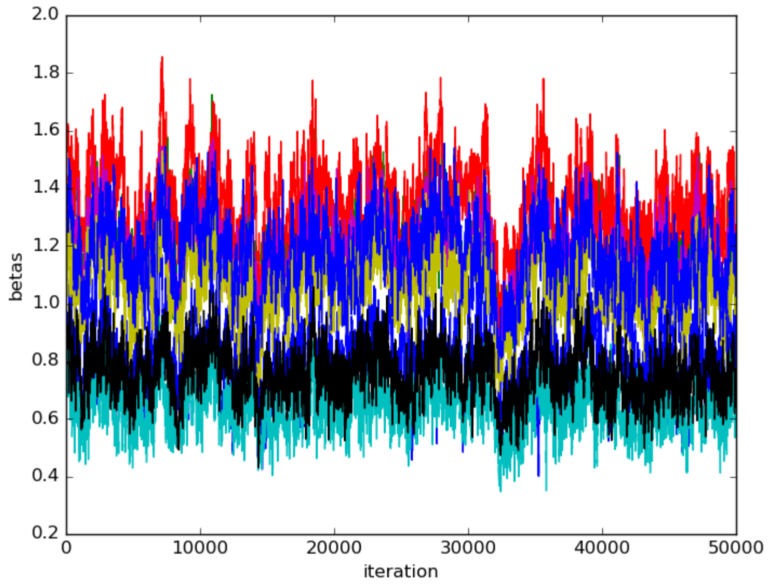
$$
\begin{aligned}
&\exp\left(\mu_1 + \phi_1(x_{2,t} - \mu_1) + \sigma_1^2/2\right)\mathbf{B}\mathbf{B}' \\
&+ \exp\left(\mu_2 + \phi_1(x_{3,t} - \mu_2) + \sigma_2^2/2\right) E\left(\mathbf{D}_{x_{1,t+1}}\mathbf{B}\mathbf{B}'\mathbf{D}_{x_{1,t+1}}' \mid x_{2t}, x_{3t}\right) \\
&+ \mathbf{R} + \lambda_2^2 E\left(\mathbf{D}_{x_{1,t+1}}\mathbf{B}\mathbf{B}'\mathbf{D}_{x_{1,t+1}} \mid x_{2t}, x_{3t}\right) \\
&- \lambda_2^2\overline{\mathbf{D}}_{t+1}^{t}\mathbf{B}\mathbf{B}'\overline{\mathbf{D}}_{t+1}^{t}.
\end{aligned}
\tag{A.4}
$$

We reintroduce to the notation the dependence on θ. Recall that our goal is an approximation of

$$
\operatorname{Var}\left[\mathbf{y}_{t+1} \mid \mathbf{y}_{1:t}\right] = \operatorname{Var}\left[E\left(\mathbf{y}_{t+1} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}, \theta\right) \mid \mathbf{y}_{1:t}\right] + E\left[\operatorname{Var}\left(\mathbf{y}_{t+1} \mid \mathbf{x}_{2t}, \mathbf{x}_{3t}, \theta\right) \mid \mathbf{y}_{1:t}\right].
$$

Write the above expression in terms of expectations, and average the quantities over several parameter samples, and for each parameter's particle filter, over the state samples with their associated weights.

Plots



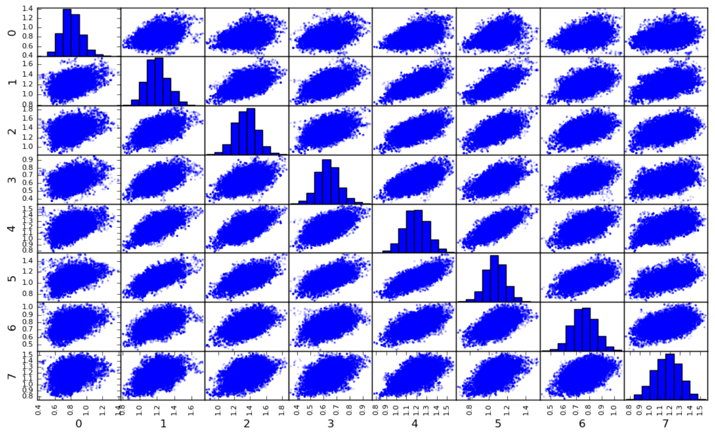
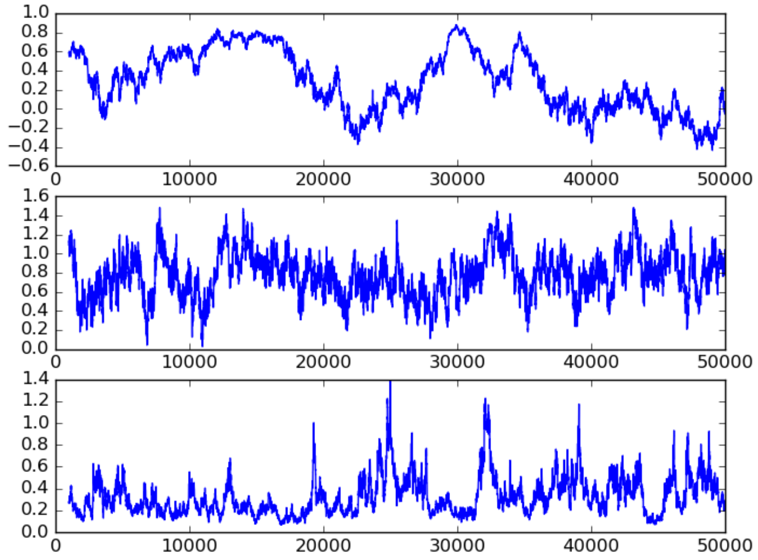


Figure A-1: PMMH samples of **B** of Jacquier et al. model



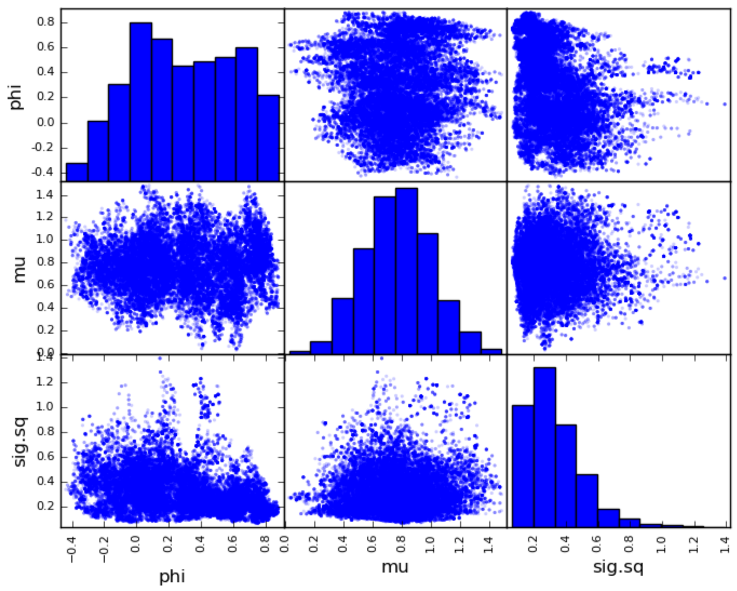
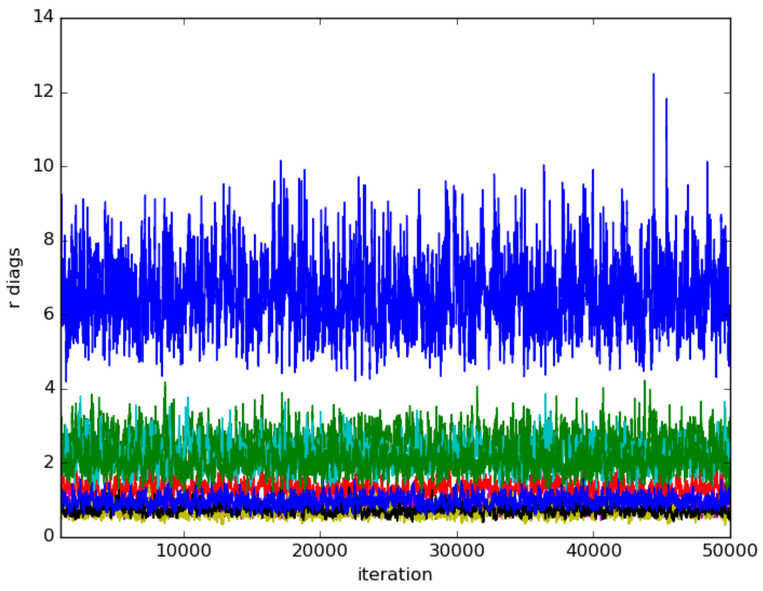


Figure A-2: PMMH samples ϕ, μ, σ^2 of Jacquier et al. model



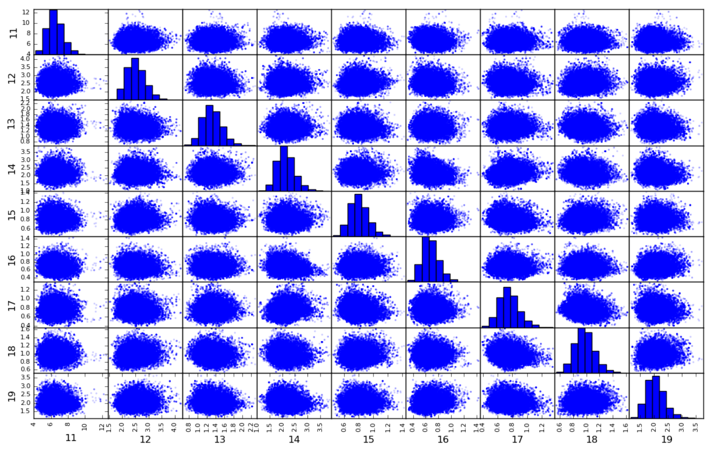


Figure A-3: PMMH samples of **R** of Jacquier et al. model

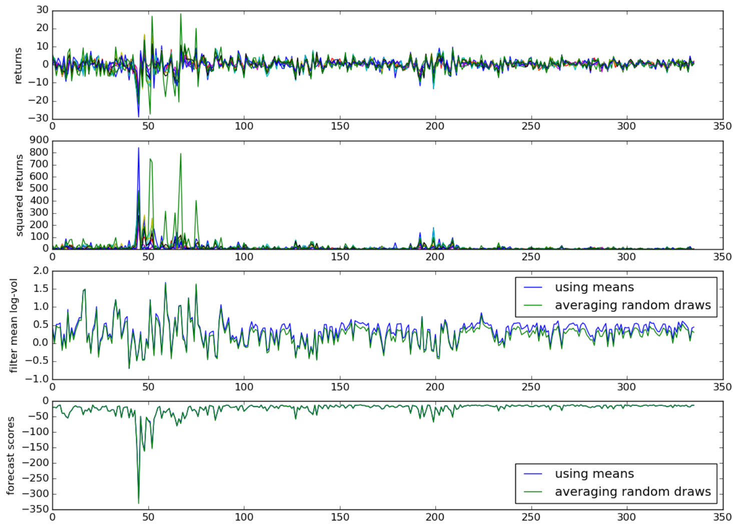


Figure A-4: Returns, squared returns, filtered log volatilities and log conditional likelihoods

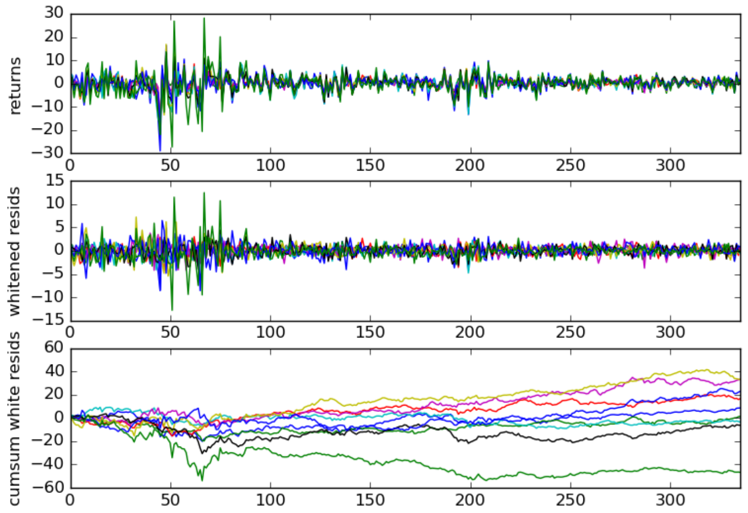


Figure A-5: Returns, whitened residuals, and their cumulative sums

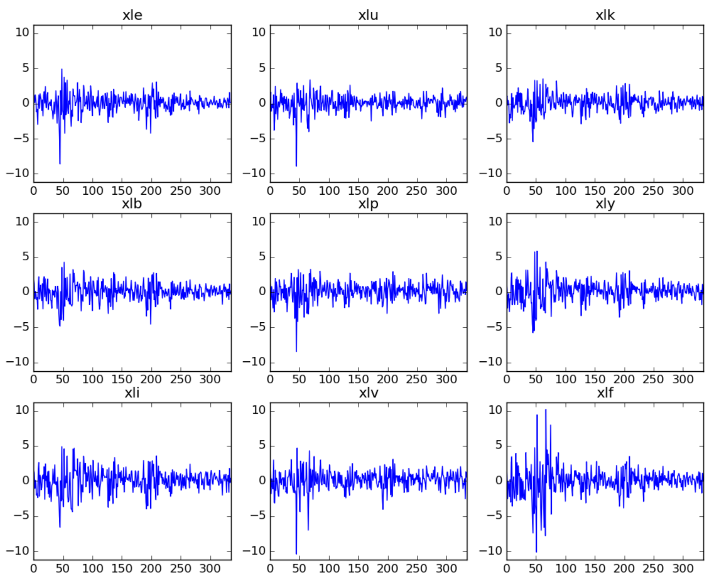


Figure A-6: prewhitened residuals

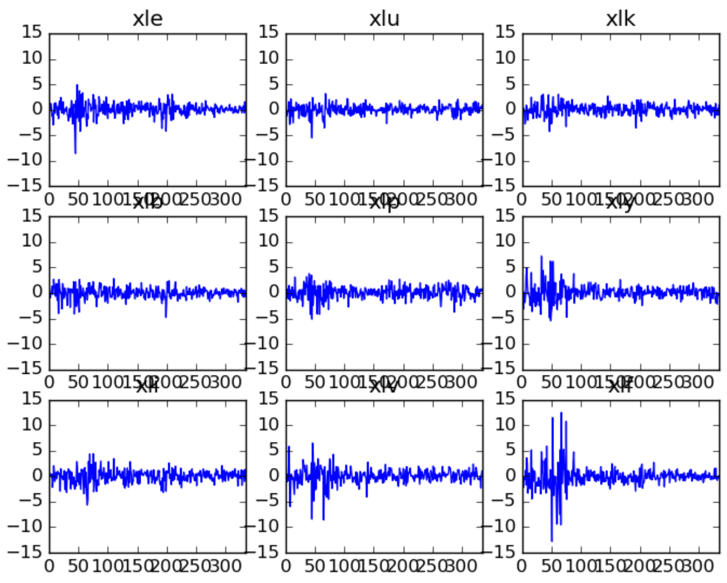


Figure A-7: Naively whitened residuals

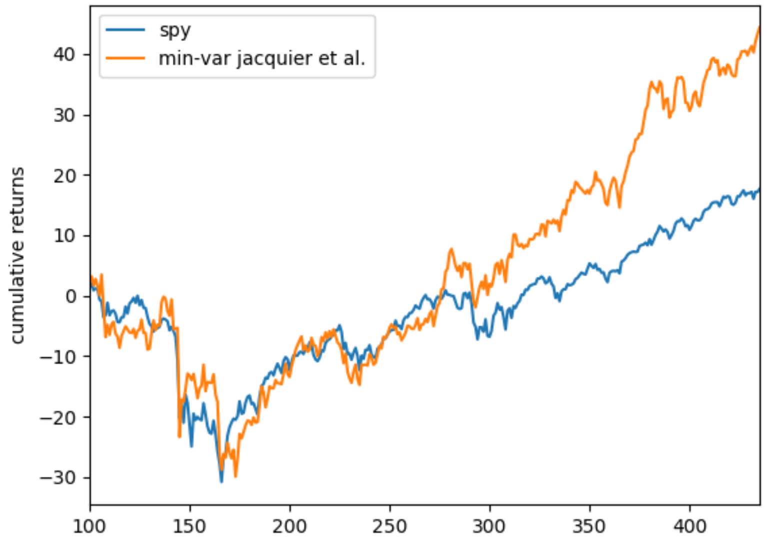


Figure A-8: Cumulative returns from minimum variance portfolio strategy using Jacquier et al. model

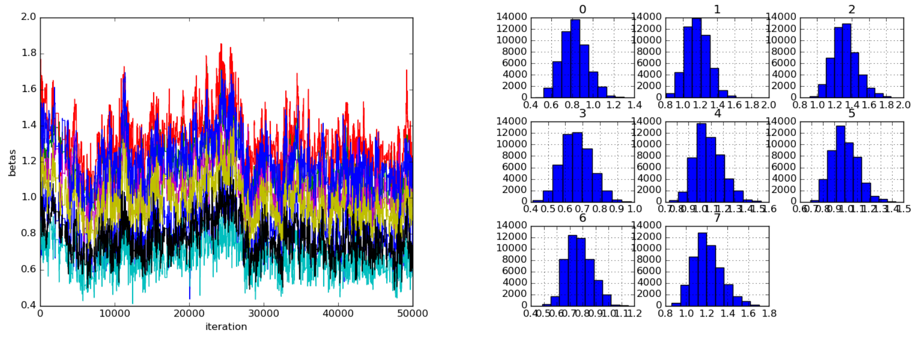


Figure A-9: PMMH **B** Samples of MSL1 model

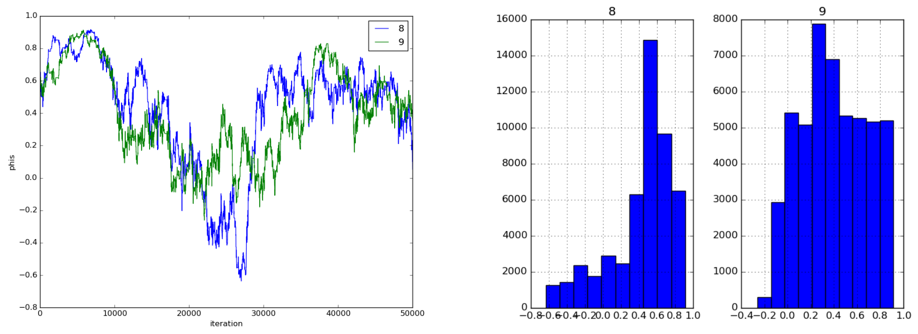


Figure A-10: PMMH ϕ_i Samples of MSL1 model

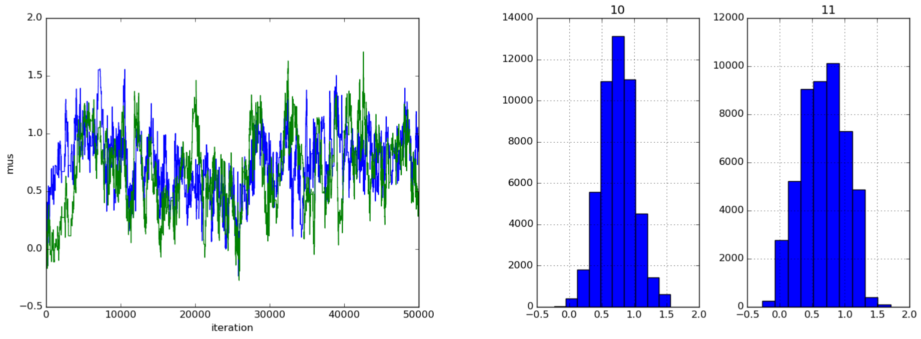


Figure A-11: PMMH μ_i Samples of MSL1 model

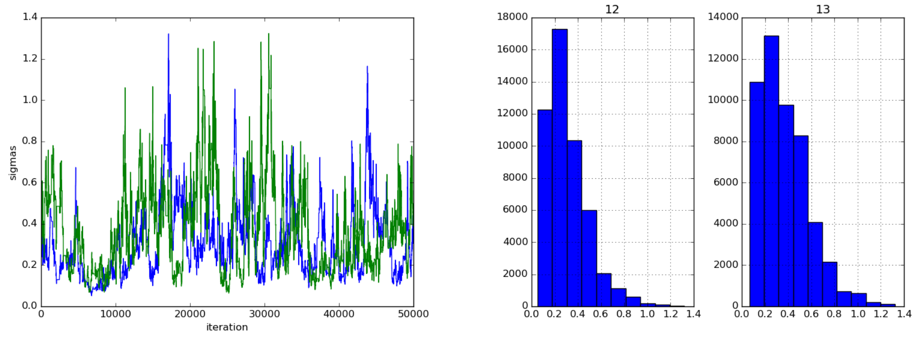


Figure A-12: PMMH σ_i^2 Samples of MSL1 model

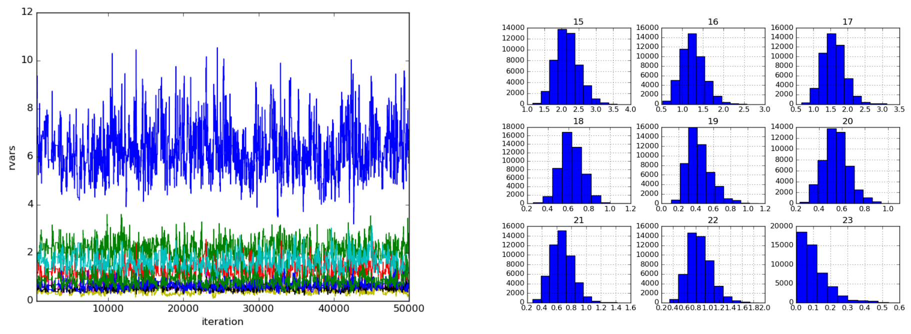


Figure A-13: PMMH $\mathbf{R}$ Samples of MSL1 model

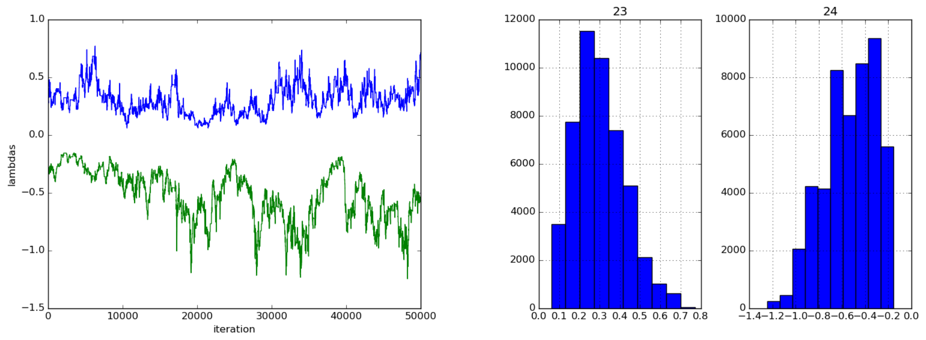


Figure A-14: PMMH λ_i Samples of MSL1 model

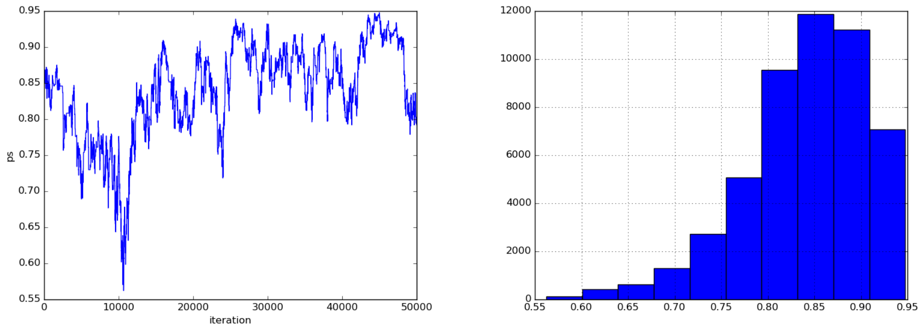


Figure A-15: PMMH p Samples of MSL1 model

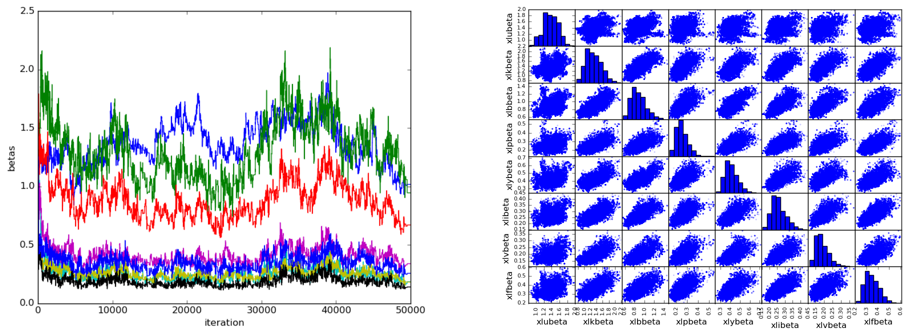


Figure A-16: PMMH **B** Samples of MSL2 model

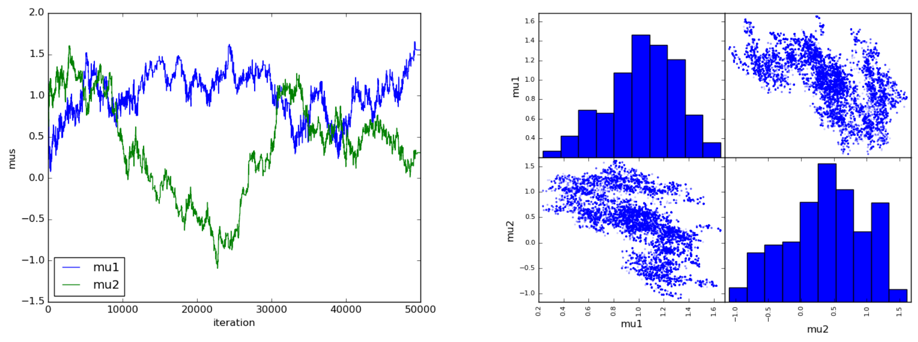


Figure A-17: PMMH μ_i Samples of MSL2 model

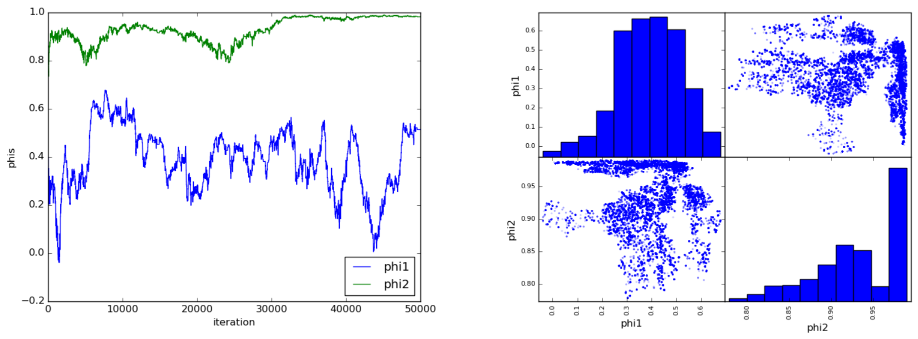


Figure A-18: PMMH ϕ_i Samples of MSL2 model

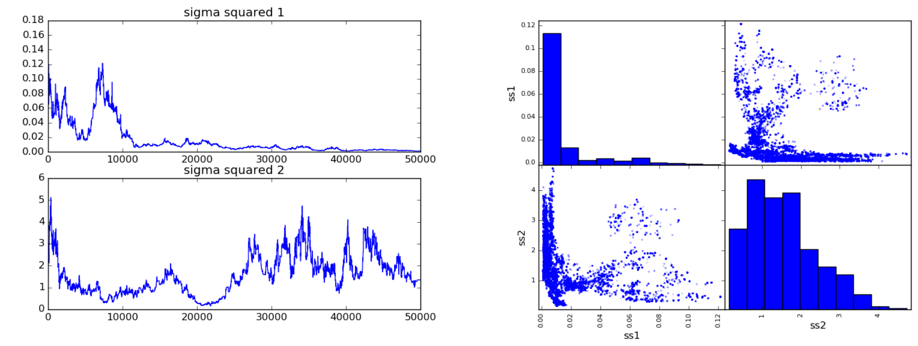


Figure A-19: PMMH σ_i^2 Samples of MSL2 model

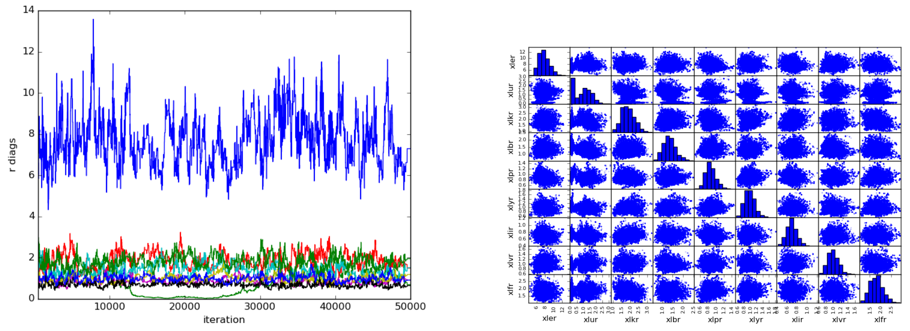


Figure A-20: PMMH **R** Samples of MSL2 model

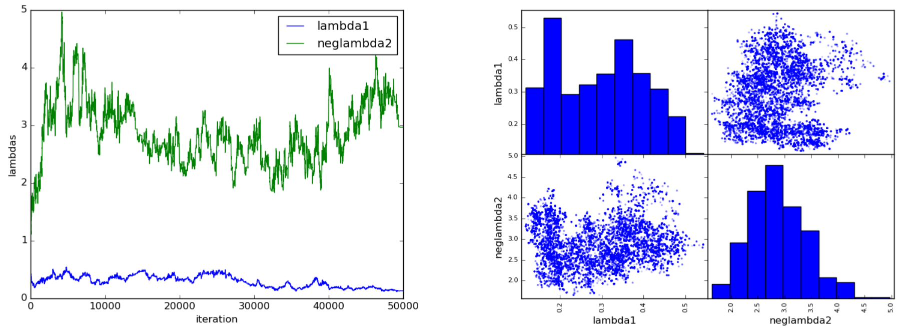


Figure A-21: PMMH λ_i Samples of MSL2 model

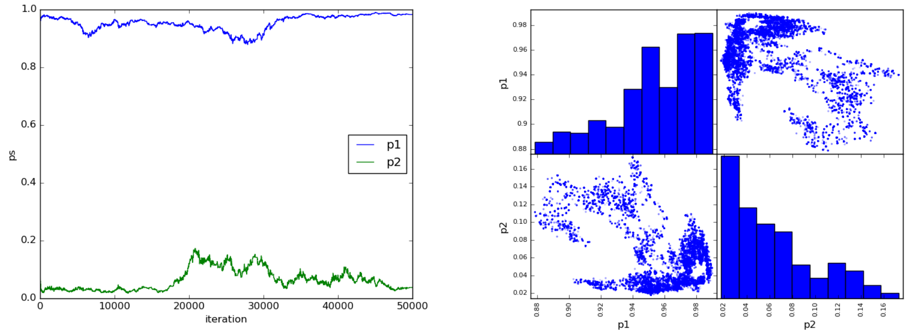


Figure A-22: PMMH p_i Samples of MSL2 model

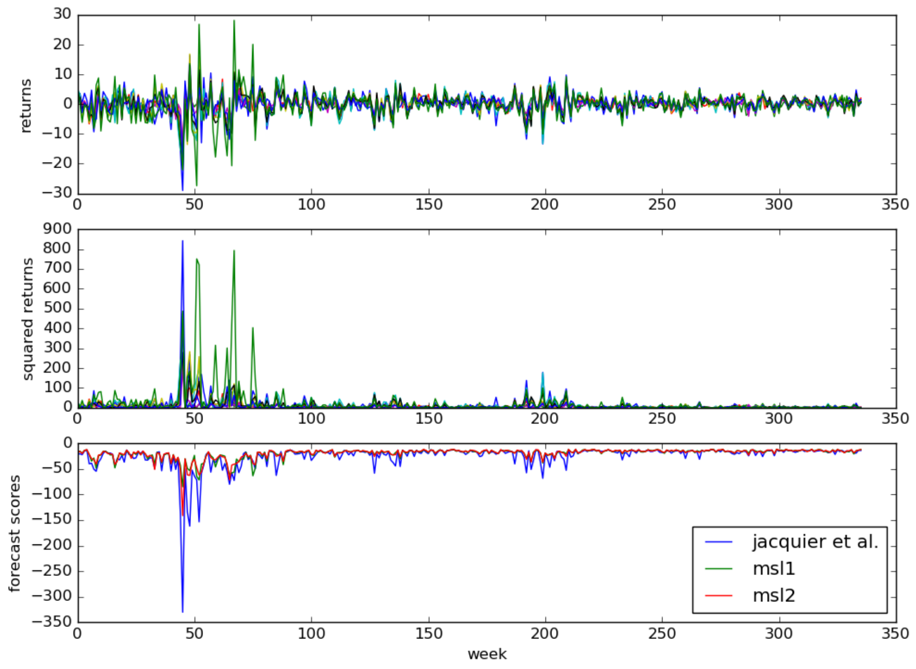
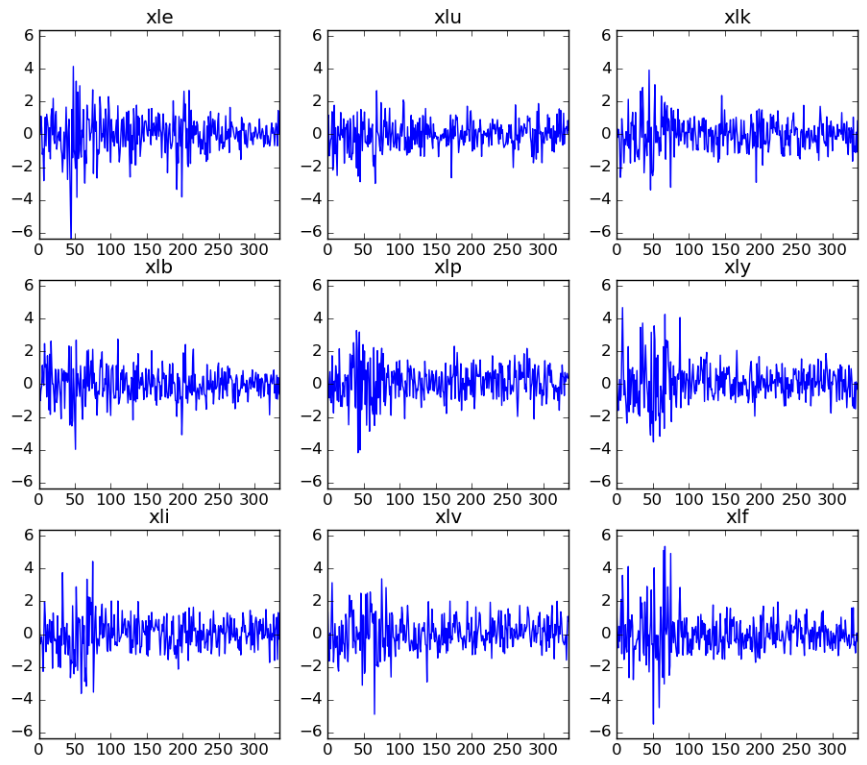


Figure A-23: Forecast Scores



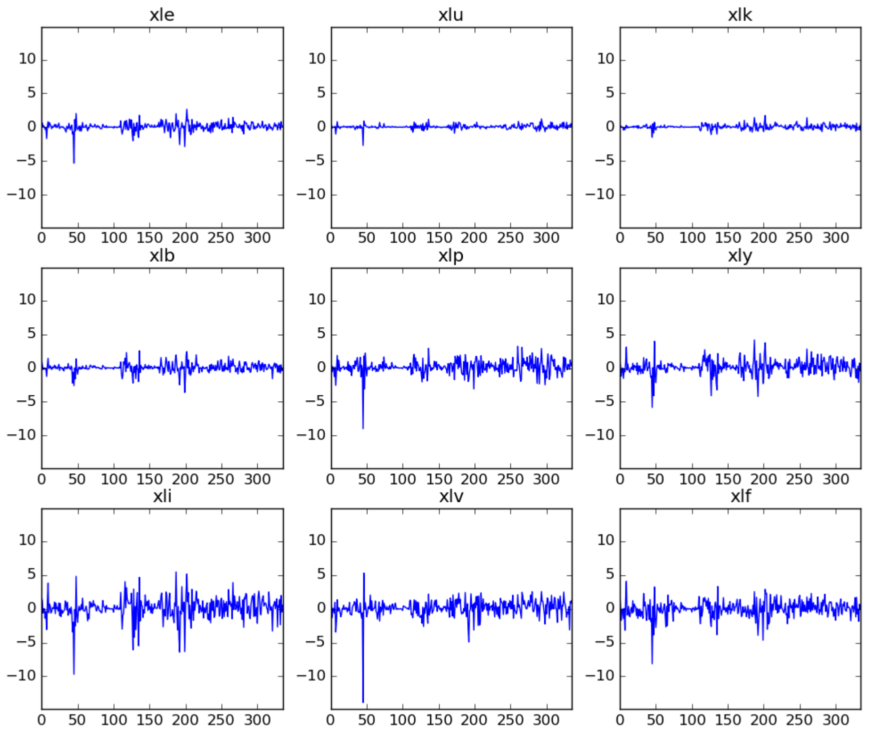


Figure A-24: Whitened Forecast Errors for MSL1 and MSL2, respectively

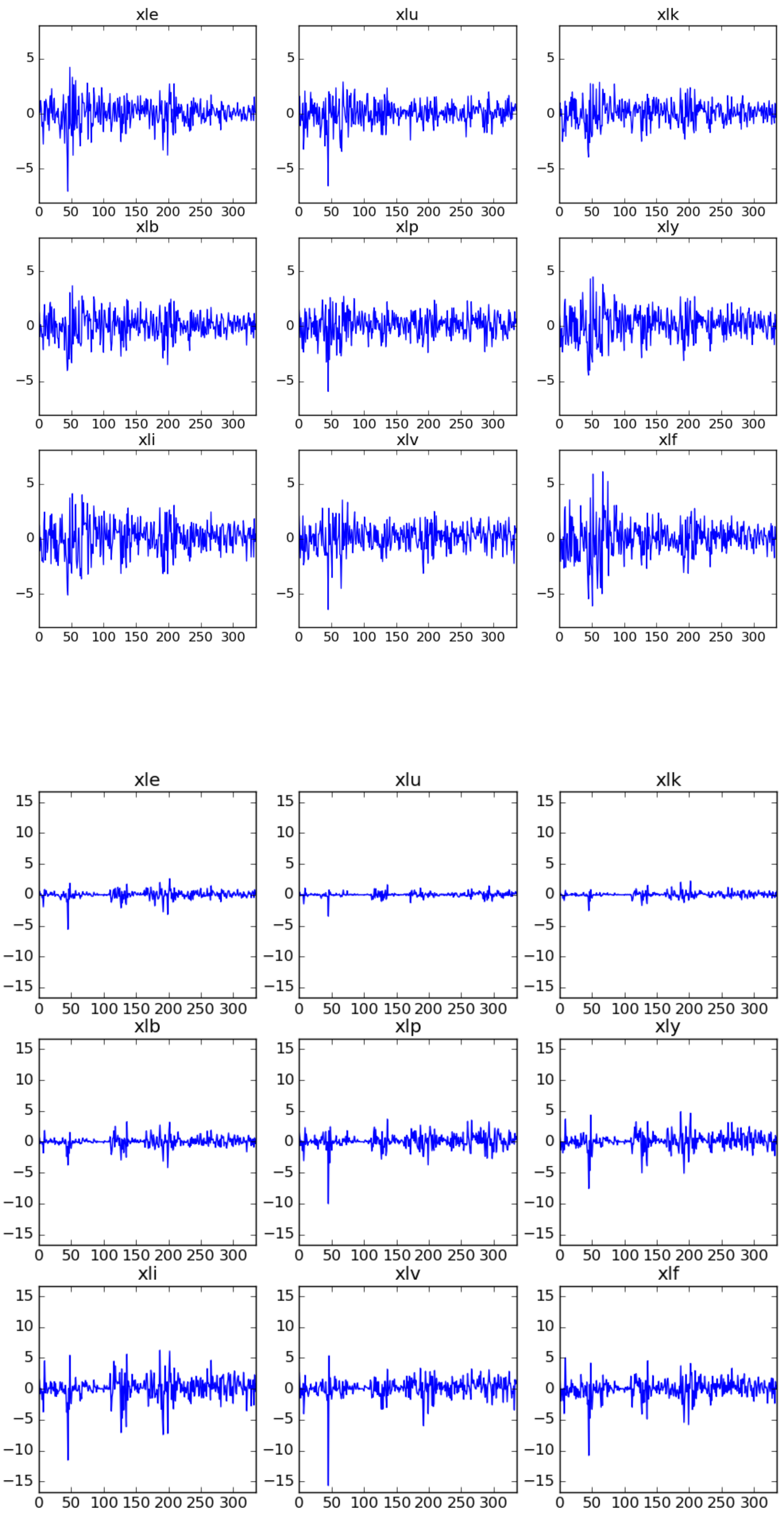


Figure A-25: Standardized Forecast Errors for MSL1 and MSL2, respectively

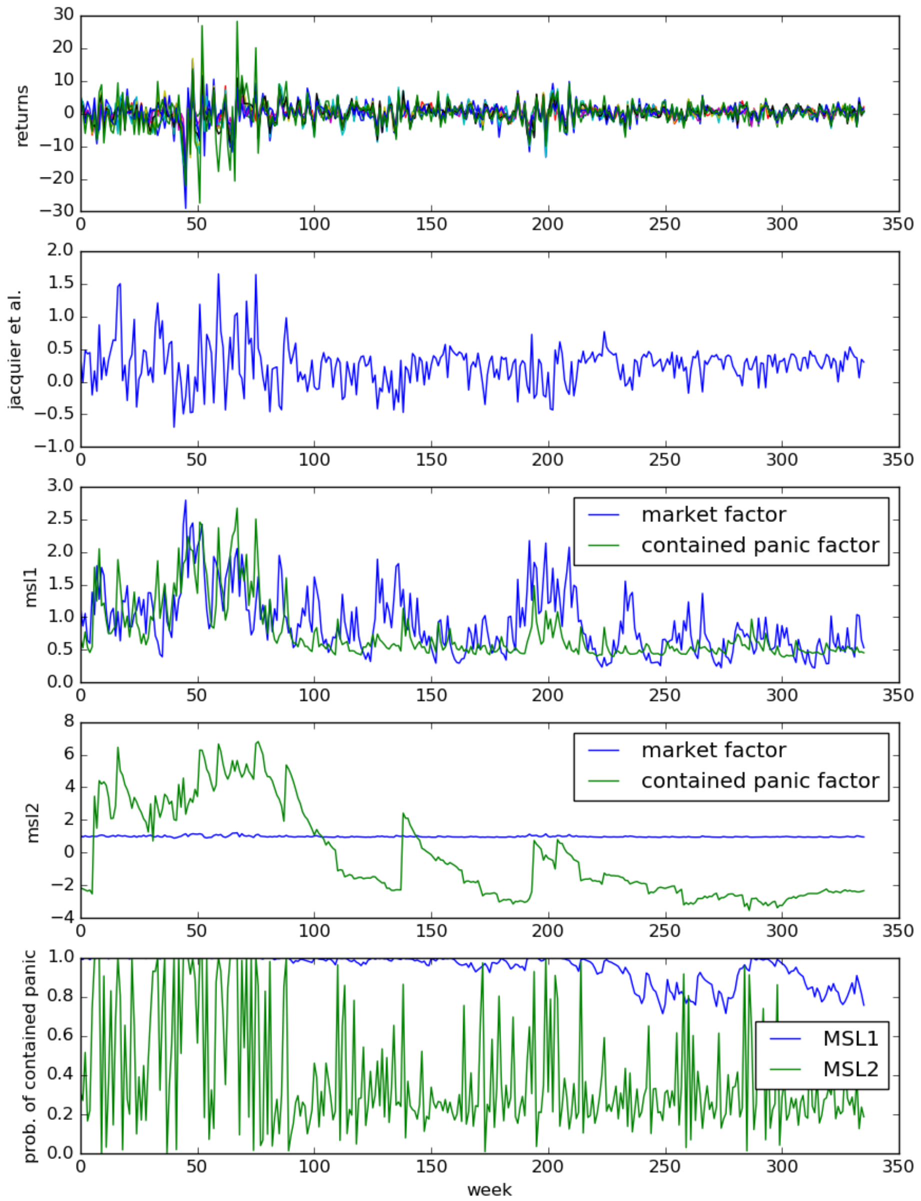


Figure A-26: Filtered Log Volatilities

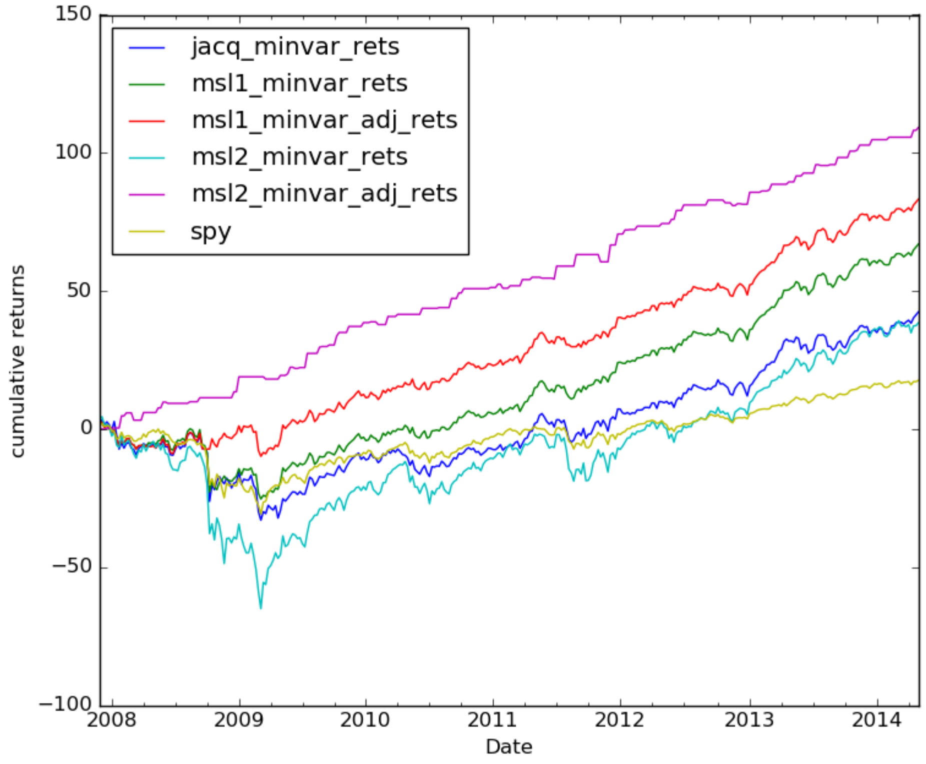


Figure A-27: Cumulative Returns from Minimum Variance Portfolio Strategy

Bibliography

[Aguilar and West, 2000] Aguilar, O. and West, M. (2000). Bayesian dynamic factor models and portfolio allocation. *Journal of Business & Economic Statistics*, 18(3):338–357.

[Amisano and Geweke, 2008] Amisano, G. and Geweke, J. (2008). Comparing and evaluating bayesian predictive distributions of assets returns. Working Paper Series 969, European Central Bank.

[Andrieu and Doucet, 2002] Andrieu, C. and Doucet, A. (2002). Particle filtering for partially observed gaussian state space models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(4):827–836.

[Andrieu et al., 2010] Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342.

[Andrieu and Roberts, 2009] Andrieu, C. and Roberts, G. O. (2009). The pseudo-marginal approach for efficient monte carlo computations. *Ann. Statist.*, 37(2):697–725.

[Cappé et al., 2005] Cappé, O., Moulines, E., and Rydén, T. (2005). *Inference in Hidden Markov Models*. Springer, New York, New York.

[Carter and Kohn, 1994] Carter, C. K. and Kohn, R. (1994). On gibbs sampling for state space models. *Biometrika*, 81(3):541–553.

[Chen and Liu, 2000] Chen, R. and Liu, J. S. (2000). Mixture kalman filters. *J. R. Statist. Soc. B*, 62:493–508.

[Chib et al., 2006] Chib, S., Nardari, F., and Shephard, N. (2006). Analysis of high dimensional multivariate stochastic volatility models. *Journal of Econometrics*, 134(2):341–371.

[Chib et al., 2009] Chib, S., Omori, Y., and Asai, M. (2009). *Multivariate Stochastic Volatility*, pages 365–400. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Chopin, 2004] Chopin, N. (2004). Central limit theorem for sequential monte carlo methods and its application to bayesian inference. *Ann. Statist.*, 32(6):2385–2411.

[Crisan and Doucet, 2002] Crisan, D. and Doucet, A. (2002). A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on Signal Processing*, 50(3):736–746.

[Douc, 2005] Douc, R. (2005). Comparison of resampling schemes for particle filtering. In *In 4th International Symposium on Image and Signal Processing and Analysis (ISPA*, pages 64–69.

[Doucet and Johansen, 2011] Doucet, A. and Johansen, A. M. (2011). A tutorial on particle filtering and smoothing: fifteen years later.

[Fearnhead, 2011] Fearnhead, P. (2011). Mcmc for state space models. In Steve Brooks, Andrew Gelman, G. L. J. and Meng, X.-L., editors, *Handbook of Markov Chain Monte Carlo*. Chapman & Hall/CRC.

[Flegal et al., 2017] Flegal, J. M., Hughes, J., Vats, D., and Dai, N. (2017). *mcmcse: Monte Carlo Standard Errors for MCMC*. Riverside, CA, Denver, CO, Coventry, UK, and Minneapolis, MN. R package version 1.3-2.

[Fritsche et al., 2009] Fritsche, C., Schon, T. B., and Klein, A. (2009). The marginalized auxiliary particle filter. *2009 3rd IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 289–292.

[Gelman et al., 2013] Gelman, A., Carlin, J., Stern, H., Dunson, D., Vehtari, A., and Rubin, D. (2013). *Bayesian Data Analysis, Third Edition*. Chapman & Hall/CRC Texts in Statistical Science. Taylor & Francis.

[Geyer, 2005] Geyer, C. J. (2005). Markov chain monte carlo lecture notes.

[Gneiting and Raftery, 2007] Gneiting, T. and Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378.

[Gordon et al., 1993] Gordon, N., Salmond, D., and Smith, A. (1993). Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEEE Proceedings F, Radar and Signal Processing*, 140(2):107–113.

[Grenander et al., 1991] Grenander, U., Chow, Y., and Keenan, D. (1991). *Hands: a pattern theoretic study of biological shapes*. Research notes in neural computing. Springer-Verlag.

[Haario et al., 2001] Haario, H., Saksman, E., and Tamminen, J. (2001). An adaptive metropolis algorithm. *Bernoulli*, 7(2):223–242.

[Jacob et al., 2009] Jacob, P., Chopin, N., Robert, C. P., and Rue, H. (2009). Comments on "Particle Markov chain Monte Carlo" by C. Andrieu, A. Doucet, and R. Hollenstein. *ArXiv e-prints*.

[Jacquier et al., 1999] Jacquier, E., Polson, N. G., and Rossi, P. (1999). Stochastic volatility: Univariate and multivariate extensions. Computing in Economics and Finance 1999 112, Society for Computational Economics.

[Jacquier et al., 1994] Jacquier, E., Polson, N. G., and Rossi, P. E. (1994). Bayesian analysis of stochastic volatility models. *Journal of Business & Economic Statistics*, 12(4):371–389.

[Jianjuna et al., 2007] Jianjuna, Y., Jianqiua, Z., and Klaasb, M. (2007). The marginal rao-blackwellized particle filter for mixed linear/nonlinear state space models.

[Kantas et al., 2014] Kantas, N., Doucet, A., Singh, S. S., Maciejowski, J., and Chopin, N. (2014). On Particle Methods for Parameter Estimation in State-Space Models. *ArXiv e-prints*.

[Kastner and Frühwirth-Schnatter, 2014] Kastner, G. and Frühwirth-Schnatter, S. (2014). Ancillarity-sufficiency interweaving strategy (asis) for boosting mcmc estimation of stochastic volatility models. *Comput. Stat. Data Anal.*, 76(C):408–423.

[Kim et al., 1998] Kim, S., Shephard, N., and Chib, S. (1998). Stochastic volatility: Likelihood inference and comparison with arch models. *Review of Economic Studies*, 65(3):361–393.

[Kipnis and Varadhan, 1986] Kipnis, C. and Varadhan, S. R. S. (1986). Central limit theorem for additive functionals of reversible markov processes and applications to simple exclusions. *Comm. Math. Phys.*, 104(1):1–19.

[Kitagawa, 1998] Kitagawa, G. (1998). A self-organizing state space model. 93(443):1203–1215.

[Kokkala and Särkkä, 2014] Kokkala, J. and Särkkä, S. (2014). Combining Particle MCMC with Rao-Blackwellized Monte Carlo Data Association for Parameter Estimation in Multiple Target Tracking. *ArXiv e-prints*.

[Kong et al., 1994] Kong, A., Liu, J. S., and Wong, W. H. (1994). Sequential imputations and bayesian missing data problems. *Journal of the American statistical association*, 89(425):278–288.

[Liu, 1996] Liu, J. S. (1996). Metropolized independent sampling with comparisons to rejection sampling and importance sampling.

[Liu et al., 2001] Liu, J. S., Chen, R., and Logvinenko, T. (2001). *Sequential Monte Carlo Methods in Practice*, chapter A Theoretical Framework for Sequential Importance Sampling with Resampling, pages 225–246. Springer New York, New York, NY.

[Lopes and Polson, 2010] Lopes, H. and Polson, N. (2010). Bayesian inference for stochastic volatility modeling. *Risk*, 1, 2:515–551.

[Lopes et al., 2006] Lopes, H. F., Carvalho, C. M., Bernardo, I., and Smith, A. P. (2006). Factor stochastic volatility with time varying loadings and markov switching regimes.

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

[Meyn and Tweedie, 2009] Meyn, S. and Tweedie, R. L. (2009). *Markov Chains and Stochastic Stability*. Cambridge University Press, New York, NY, USA, 2nd edition.

[Moral, 1996] Moral, P. D. (1996). Nonlinear filtering: Interacting particle resolution.

[Niemi and West, 2010] Niemi, J. and West, M. (2010). Adaptive mixture modeling metropolis methods for bayesian analysis of nonlinear state-space models. *Journal of Computational and Graphical Statistics*, 19(2):260–280.

[Nordh, 2014] Nordh, J. (2014). Rao-blackwellized auxiliary particle filters for mixed linear/nonlinear gaussian models. In *2014 12th International Conference on Signal Processing (ICSP)*, pages 1–6.

[Pitt et al., 2010] Pitt, M., Silva, R., Giordani, P., and Kohn, R. (2010). Auxiliary particle filtering within adaptive metropolis-hastings sampling.

[Pitt and Shephard, 1999a] Pitt, M. K. and Shephard, N. (1999a). Filtering via simulation: Auxiliary particle filters. 94(446):590–??

[Pitt and Shephard, 1999b] Pitt, M. K. and Shephard, N. (1999b). Time varying covariances: a factor stochastic volatility approach. In *Bayesian Statistics 6, Proceedings of the Sixth Valencia International Meeting,(edited by JM Bernardo, JO Berger, AP Dawid and AFM Smith)*, volume 547, pages 547–570.

[Roberts and Rosenthal, 2004] Roberts, G. O. and Rosenthal, J. S. (2004). General state space markov chains and mcmc algorithms. *Probab. Surveys*, 1:20–71.

[Shephard and Pitt, 1997] Shephard, N. and Pitt, M. K. (1997). Likelihood analysis of non-gaussian measurement time series. *Biometrika*, 84(3):653–667.

[Shumway and Stoffer, 2006] Shumway, R. H. and Stoffer, D. S. (2006). *Time series analysis and its applications : with R examples*. Springer texts in statistics. Springer, New York.

[Smith and Gelfand, 1992] Smith, A. F. M. and Gelfand, A. E. (1992). Bayesian statistics without tears: A sampling-resampling perspective. *The American Statistician*, pages 84–88.

[Taylor, 1982] Taylor, S. (1982). Financial returns modelled by the product of two stochastic processes, a study of daily sugar prices 1961-79. 1.

[Whiteley and Johansen, 2011] Whiteley, N. and Johansen, A. M. (2011). Recent developments in auxiliary particle filtering. In Barber, C. and Chiappa, editors, *Bayesian Time Series Models*. Cambridge University Press.